SIAM J. Sci. Comput.
Vol. 21, No. 2, pp. 691–710

A MULTIGRID PRECONDITIONED NEWTON–KRYLOV METHOD*

D. A. KNOLL AND W. J. RIDER†

Abstract. We study multigrid preconditioning of matrix-free Newton–Krylov methods as a means of developing more efficient nonlinear iterative methods for large scale simulation. Newton–Krylov methods have proven dependable in solving nonlinear systems while not requiring the explicit formation or storage of the complete Jacobian. However, the standard algorithmic scaling of Krylov methods is nonoptimal, with increasing linear system dimension. This motivates our use of multigrid-based preconditioning. It is demonstrated that a simple multigrid-based preconditioner can effectively limit the growth of Krylov iterations as the dimension of the linear system is increased. Different performance aspects of the proposed algorithm are investigated on three nonlinear, nonsymmetric, boundary value problems. Our goal is to develop a hybrid methodology which has Newton–Krylov nonlinear convergence properties and multigrid-like linear convergence scaling for large scale simulation.

Key words. Newton–Krylov methods, multigrid methods, nonlinear boundary value problems

AMS subject classifications. 65F10, 65H10, 65N55

PII. S1064827598332709

1. Introduction.

Matrix-free Newton–Krylov methods [4, 5] have been demonstrated to deliver inexact Newton [6] nonlinear convergence performance without requiring the formation or inversion of the complete Jacobian. By complete we mean a Jacobian matrix which contains all of its naturally occurring elements, irrespective of whether these elements are evaluated analytically or numerically. These methods have also been referred to as inexact Newton–finite-difference projection methods [3]. In [5] this capability was demonstrated with the two-dimensional Bratu problem and the two-dimensional driven cavity problem. The Bratu problem was solved on a 32×32 grid and only the Laplacian was used to construct a preconditioner. The driven cavity problem was solved on a 63×63 grid using only the biharmonic operator as a preconditioner for Reynolds numbers ranging from 500 to 5000. Some degradation was observed from this preconditioner with increasing Reynolds numbers. In both cases, the complete Jacobian was never formed, and the preconditioner was constructed from a simple linear part of the PDE (i.e., simple approximations to the Jacobian). Our goal is to efficiently extend this unique nonlinear iterative method capability to large scale simulation, i.e., finer grids. To achieve this goal we use a simple, linear, multigrid method applied to an approximate (not complete) Jacobian to precondition a matrix-free Newton–Krylov method.

We have employed matrix-free Newton–Krylov methods [5] on challenging problems, with single grid preconditioners, using a "mesh-sequencing" algorithm, where the solution on a coarse grid is interpolated to a grid of increased refinement and used as an initial guess on this fine grid [15, 14]. This greatly increases the radius of convergence of Newton's method and accelerates the nonlinear convergence (reduces the number of required Newton iterations). However, no coarse grid information was used

*Received by the editors January 19, 1998; accepted for publication (in revised form) September 9, 1998; published electronically October 14, 1999. This work was supported under the auspices of the U.S. Department of Energy under DOE contract W-7405-ENG-36 at Los Alamos National Laboratory.

http://www.siam.org/journals/sisc/21-2/33270.html

†M.S. D413, Hydrodynamic Methods Group, Applied Theoretical and Computational Physics Division, Los Alamos National Laboratory, Los Alamos, NM 87545 (nol@lanl.gov, wjr@lanl.gov).

in the preconditioner, and as the grid was refined the performance of a standard single grid ILU preconditioner was observed to degrade. As a result a significant growth in Krylov iterations per Newton iteration [14] was observed. This growth is of specific concern if one is using a Krylov algorithm such as GMRES [27] since the storage scales linearly and the work scales quadratically with the linear (Krylov) iteration number. In this study, we build upon the work in [5] and [15, 14, 17] by investigating preconditioners which enable efficient, more optimal, fine grid simulation.

Our proposed multigrid preconditioned, matrix-free, Newton–Krylov algorithm (NKMG) differs from previous nonlinear multilevel methods, such as the full approximation scheme (FAS) [2], the standard Newton's method with multigrid as the linear solver (referred to here as Newton-MG), or the two-level Newton–Krylov–Schwarz (NKS) [12]. In the FAS algorithm, multigrid is the outermost iteration, with local, nonlinear iterative methods as the smoother. In the Newton-MG algorithm, one is required to obtain a global linearization (a complete Jacobian) by evaluating and storing the elements of the Jacobian. Newton's method is the outermost iteration in Newton-MG; however, any approximations to the Jacobian will affect nonlinear convergence rates.

In our proposed algorithm, NKMG, Newton's method is the outermost iteration. However, the global linearization is accomplished through finite-difference approximations to the matrix-vector products in the Krylov method [5], and approximations to the Jacobian, which are required only for a preconditioner, do not affect nonlinear convergence rates. Additionally, in NKMG the linear iterative method is a combination of multigrid preconditioning of a Krylov method. Our proposed algorithm is the most similar to NKS, the main difference being that our preconditioner uses a more traditional multigrid coarsening, as compared to a two-level Schwarz method. We also mention the recent work of Washio and Oosterlee [29] which investigates using a GMRES algorithm on the fine grid to accelerate an FAS method.

We demonstrate the ability of a multigrid preconditioner to limit the growth of Krylov iterations per Newton iteration as the grid is refined. Some of the unique properties of a multigrid preconditioned matrix-free Newton–Krylov algorithm are demonstrated. Specifically, the ability to precondition a convection-diffusion system with only the diffusion operator, while maintaining inexact Newton nonlinear convergence characteristics, is demonstrated. Also, the ability of a low-order convective discretization to precondition a higher-order convective discretization, while maintaining inexact Newton nonlinear convergence characteristics, is demonstrated. The NKMG is compared to a Newton-MG algorithm, where the multigrid preconditioner from NKMG is used as a stand-alone solver for the linear problems at each Newton step. NKMG is also compared to a more standard Newton–Krylov method with ILU(0) [20] as the preconditioner.

We emphasize that the Newton-MG methods used in this study are not chosen because they have proven to be the most efficient. We compare Newton-MG methods for which the multigrid solver in Newton-MG is the multigrid preconditioner in NKMG. Our intent here is similar to the study in [1], except applied to nonlinear problems. We ask the question, *Can a simple multigrid method provide effective preconditioning for Newton–Krylov methods?*

Next we describe the matrix-free Newton–Krylov method and discuss some of its unique capabilities. Section 3 motivates multigrid preconditioning and describes our proposed multigrid preconditioned Newton–Krylov method. Section 4 presents algorithm performance results for one-dimensional model problems and two-dimensional model problems. A summary and conclusions are given in section 5.

2. Newton–Krylov method.

Application of Newton's method requires the solution of the linear system

$$\mathbf{J}^n \delta \mathbf{u^n} = -\mathbf{F}(\mathbf{u}^n), \tag{1}$$

where $\mathbf{J}$ is the Jacobian matrix, $\mathbf{F}(\mathbf{u})$ is the nonlinear system of equations, and $\mathbf{u}$ is the state vector. The new solution approximation at iteration $n+1$ is obtained from

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \delta \mathbf{u}^n. \tag{2}$$

We use the generalized minimal RESidual (GMRES) algorithm [27] to solve (1). Our choice of GMRES is based on comparing a variety of nonsymmetric Krylov subspace methods within the context of a matrix-free Newton–Krylov implementation and finding GMRES to be superior [18, 16]. Having chosen GMRES we are highly motivated to develop preconditioners which contain the growth of the Krylov iterations which typically accompany grid refinement since the work in GMRES scales quadratically with iteration count. In this study we consider single grid based preconditioners derived from simple matrix splittings such as weighted, or damped, Jacobi (DJ); symmetric Gauss–Seidel (SGS); and the well-known approximate factorization ILU(0) [20]. The matrix-splitting techniques (DJ and SGS) are also used as smoothers in multigrid-based preconditioners.

Since the use of an iterative technique to solve (1) does not require the exact solution of the linear system, the resulting algorithm is categorized as an "inexact" Newton's method [6]. We employ the following inexact convergence criteria on the linear iteration:

$$\parallel \mathbf{J}^n \delta \mathbf{u}^n + \mathbf{F}(\mathbf{u}^n) \parallel_2 < \gamma \parallel \mathbf{F}(\mathbf{u}^n) \parallel_2 . \tag{3}$$

Keeping γ small, which is required for Newton-like nonlinear convergence, can put a significant strain on the preconditioner, especially as the dimension of the linear system grows. There is a trade-off between the effort required to solve the linear system to a tight tolerance and the resulting required number of nonlinear iterations. A relatively large value for γ will result in less work for the Krylov method but more nonlinear iterations. Examples of this trade-off between total nonlinear iterations and CPU time are given in [24, 17]. Only constant values of γ are considered in this study, and these values will be such that true quadratic convergence of Newton's method is not observed. For further discussion on the subject of choosing γ, see [8], and for the effect of different choices of γ on the solution of the Navier–Stokes equations, see [24, 17].

The GMRES algorithm (as well as other Krylov algorithms) requires the action of the Jacobian only in the form of matrix-vector products, which may be approximated by [5]

$$\mathbf{J}\mathbf{P}^{-1}\mathbf{v} \approx \frac{\mathbf{F}(\mathbf{u} + \epsilon \mathbf{P}^{-1}\mathbf{v}) - \mathbf{F}(\mathbf{u})}{\epsilon}. \tag{4}$$

Here we employ right preconditioning, where $\mathbf{P}^{-1}$ is the inverse of the preconditioning matrix and $\mathbf{v}$ is a Krylov vector. To evaluate the small perturbation ϵ, we use

$$\epsilon = \frac{1}{N||\mathbf{v}||_2} \sum_{m=1}^{N} (a|u_m| + a) , \tag{5}$$

where N is the linear system dimension and a is a constant whose magnitude is approximately the square root of machine roundoff ($a = 10^{-6}$ in this study). Other options are discussed in [5]. The use of (4) enables the action of the Jacobian without explicitly forming or storing the matrix. In most instances, however, some approximation of the Jacobian is still required to generate an effective preconditioner.

This matrix-free approach, besides its obvious memory advantage, has many unique capabilities, many described in [5]. In instances where the Jacobian is expensive to form the same, "lagged," Jacobian can be used as a preconditioner for several Newton iterations while maintaining strong nonlinear convergence [15, 14, 16]. In systems of conservation laws where convection dominates, high-order convection schemes are desired for accuracy. Using the matrix-free method one can construct the preconditioner from a better conditioned low-order discretization, saving memory and often resulting in more effective preconditioning [12, 19, 13]. Consider the option of forming the preconditioner using a low-order convection scheme:

$$\mathbf{J}\mathbf{P}^{-1}\mathbf{v} \approx \frac{\mathbf{F}_{high}(\mathbf{u} + \epsilon \mathbf{P}_{low}^{-1}\mathbf{v}) - \mathbf{F}_{high}(\mathbf{u})}{\epsilon}. \tag{6}$$

Here, $\mathbf{F}_{high}(\mathbf{u})$ denotes the nonlinear function evaluated with a high-order discretization and $\mathbf{P}_{low}^{-1}$ denotes preconditioning formed with a low-order discretization. This option will result in less storage (a smaller discretization stencil for $\mathbf{P}_{low}$) and a potentially more effective preconditioning due to improved conditioning in $\mathbf{P}_{low}$ as compared to $\mathbf{P}_{high}$. For a convection-diffusion system, one can also consider splitting the Jacobian into its convection and diffusion parts, $\mathbf{J} = \mathbf{J}_{con} + \mathbf{J}_{diff}$, and then constructing the preconditioner from only the diffusion part:

$$\mathbf{J}\mathbf{P}^{-1}\mathbf{v} \approx \frac{\mathbf{F}(\mathbf{u} + \epsilon \mathbf{P}_{diff}^{-1}\mathbf{v}) - \mathbf{F}(\mathbf{u})}{\epsilon}. \tag{7}$$

This option was considered in [5, 24]. Both of these options, (6) and (7), will be demonstrated in section 4 in conjunction with multigrid preconditioning.

3. Preconditioning and proposed algorithm. Standard single grid preconditioners, whose memory scales linearly with problem size, will become less effective as the dimension of the linear system is increased as a result of an increased condition number:

$$\kappa(\mathbf{J}\mathbf{P}_{SG}^{-1})_{N_2} > \kappa(\mathbf{J}\mathbf{P}_{SG}^{-1})_{N_1}. \tag{8}$$

Here, $\kappa(\mathbf{A})$ is the condition number of matrix $\mathbf{A}$ and N is the dimension of the linear system, $N_2 > N_1$; SG denotes single grid. Thus, as N increases, not only will the cost of a GMRES iteration increase, but the number of GMRES iterations, for a given linear convergence tolerance, will also increase. We acknowledge that with an ILU(n) preconditioner one can consider increasing n, but of course this will result in a memory requirement which does not scale linearly.

The modern tool for defeating this nonoptimal scaling is the multigrid method [2, 30]. Multigrid methods have been used to precondition conjugate gradients (CG) [11, 25, 21, 1] for symmetric positive definite problems which were considered challenging for multigrid as a stand-alone solver. This work has shown that a multigrid preconditioner has the potential to render the number of required CG iterations nearly independent of grid dimension for problems which proved difficult for either stand-alone method. Recently, multigrid has been applied successfully as a preconditioner to

```
for $igrid = 1, ngrid$ do ("Mesh Sequencing" Loop)
    while (not converged) do (Newton Loop)
    form $\mathbf{F}_{igrid}(\mathbf{u}^n_{igrid})$ and $\mathbf{P}_{igrid}$
    if $igrid \neq 1$, form $\mathbf{P}_{igrid-1}, \ldots, \mathbf{P}_1$
        while (not converged) do (Krylov/GMRES Loop)
        solving $\mathbf{J}^n\mathbf{P}^{-1}\mathbf{z} = -\mathbf{F}(\mathbf{u}^n)$ for $\mathbf{z}$ (preconditioned solution)
        In Arnoldi process (build $\mathbf{v}_k = \mathbf{J}^n\mathbf{P}^{-1}\mathbf{v}_{k-1}$, $k$th Krylov vector )
            preconditioning operation $\Rightarrow MG(\mathbf{P}_{igrid}, \ldots, \mathbf{P}_1, \mathbf{v}, \mathbf{y}) \rightarrow \mathbf{y}$
            matvec operation $\Rightarrow \mathbf{J}^n\mathbf{P}^{-1}\mathbf{v} \approx [\mathbf{F}(\mathbf{u}^n + \epsilon\mathbf{y}) - \mathbf{F}(\mathbf{u}^n)] \;/\; \epsilon$
        Complete Arnoldi process
        Least Squares Minimization process
        Linear Convergence ?
        end
        Inverse preconditioning operation $\Rightarrow MG(\mathbf{P}_{igrid}, \ldots, \mathbf{P}_1, \mathbf{z}, \delta\mathbf{u}^\mathbf{n}) \rightarrow \delta\mathbf{u}^\mathbf{n}$
    $\mathbf{u}^{n+1}_{igrid} = \mathbf{u}^n_{igrid} + \delta\mathbf{u}^n_{igrid}$
    Nonlinear Convergence ?
    end
if $igrid \neq ngrid$, interpolate $\mathbf{u}_{igrid} \rightarrow \mathbf{u}_{igrid+1}$
end
```

FIG. 1. *NKMG algorithm for boundary value problems.*

nonsymmetric linear problems [23]. It is the goal of this study to investigate the performance of multigrid preconditioning in a Newton–Krylov algorithm for nonlinear, nonsymmetric, boundary value problems, knowing that

$$\kappa(\mathbf{JP}^{-1}_{MG})_{N_2} \approx \kappa(\mathbf{JP}^{-1}_{MG})_{N_1} \tag{9}$$

for a "properly" implemented multigrid algorithm (MG denotes multigrid).

Our Newton–Krylov algorithm already contains an element of multigrid by cycling through a series of increasingly refined grids to obtain an improved initial guess for each grid [15, 14]. This is precisely the FMG (full multigrid) of an FMG-FAS algorithm [2, 30]. While this mesh sequencing method significantly accelerates the nonlinear iteration, no effect is seen on the number of linear iterations. That is, no coarse grid information is used to accelerate the linear iterative method. If, however, we define a set of coarse grid preconditioners and restriction and prolongation operators to be used within a multigrid algorithm for the preconditioning of (4), then the preconditioning system $\mathbf{Py} = \mathbf{v}$, in (4), can be solved for $\mathbf{y}$ using a linear correction scheme multigrid algorithm [2]. This proposed (right) preconditioned algorithm for boundary value problems is given in Figure 1, which describes our mesh sequencing (FMG), multigrid preconditioned, Newton–Krylov algorithm. In this algorithm the operation MG is a standard V cycle with a fixed and equal number of pre- and post-smoothing steps, a $V(\nu, \nu)$ cycle.

To complete our algorithm definition, we give the multigrid specifics used in this study. This includes forming the coarse grid preconditioners as well as the restriction and prolongation operators which are considered. Restriction, $\mathcal{R}$, and prolongation, $\mathcal{P}$, operators are used to transfer linear residuals and linear corrections between grids. In the problems considered here we use cell-centered finite volume discretization, and coarsening involves combining two fine grid volumes into one coarse grid volume in one-dimension, and combining four fine grid volumes into one coarse grid volume in two-dimensions. In this study restriction is always piecewise constant. We will consider both a piecewise constant and a piecewise linear prolongation operator $\mathcal{P}$. The second option is required for the "order rule" $m_{\mathcal{P}} + m_{\mathcal{R}} > M$ [30]. Here m is the order of the interpolation plus one and M is the highest order of the PDE. Since our problems contain second-order operators, $M = 2$.

Defining coarse grid operators for a multigrid method inside of Newton's method involves defining coarse grid Jacobians. As stated previously, when using a matrix-free Newton–Krylov method only coarse grid representations of an approximate Jacobian are required. Two distinct options are available. The first option involves restricting the dependent variables down to the coarse grid and evaluating the linear operator on the coarse grid. This is what is done for most of this study. The second approach uses a Galerkin-like, or black-box multigrid [7], approach where the coarse grid operators are constructed from the fine grid operator in conjunction with the restriction and prolongation operators:

$$\mathbf{P}_{igrid-1} = \mathcal{R} * \mathbf{P}_{igrid} * \mathcal{P}. \tag{10}$$

The complexity of this approach is directly related to the complexity of the restriction and prolongation operators. This approach will be considered in the final model problem with piecewise constant restriction and prolongation operators.

4. Algorithm performance study. In this section we examine various aspects of the proposed algorithm performance. Three well-known nonlinear convection-diffusion problems are used to study performance. Most problems were run in double precision on a Hewlett–Packard 735 workstation, with the exception of some results on the final model problem which were run on a Sun ULTRA-1. All CPU times presented are for the complete nested solution process, starting on the coarsest grid.

4.1. The one-dimensional Burgers equation. The one-dimensional nonlinear Burgers equation, posed as a boundary value problem is

$$(c_1 + c_2 U)U_x - c_3 U_{xx} = 0. \tag{11}$$

The exact solution to this problem is

$$U_{exact}(x) = \frac{c_1}{c_2}\left[1 + \tanh\left(\frac{c_1(x - \frac{L}{2})}{2c_3}\right)\right], \tag{12}$$

with $U(0) = 0$ and $U(L) = 1$. This form of Burgers's equation can be put in conservation form, $\mathcal{F}_x = 0$, by choosing

$$\mathcal{F} = c_1 U + \frac{c_2}{2}U^2 - c_3 U_x. \tag{13}$$

We discretize the conservative form using the finite volume method with nx volumes between 0 and L. The left face of volume 1 is at $x = 0$, and the right face of volume nx is located at $x = L$. Dirichlet boundary conditions are imposed on these faces using (12). We consider a uniform grid with central differencing for the diffusion term and first-order upwind differencing of the convective derivative, resulting in a three-point stencil. We will also consider the application of a higher-order convection scheme which will result in a five-point stencil.

Consider the problem $c_1 = 0.5, c_2 = -1.0, c_3 = 0.25$. Table 1 presents performance results for a four grid mesh sequencing simulation, $nx(1) = 50$ to $nx(4) = 400$.

Table 1

Single grid (SG) vs. multigrid (MG) preconditioner performance on the 1*-D Burgers equation,* c_3=0.25 *(direct solve on grid* 1*), Newton tolerance* $=1.0 \times 10^{-8}$.

Method	Inexact	Grid 1		Grid 2		Grid 3		Grid 4		CPU
	Newton	nx = 50		nx = 100		nx =200		nx = 400		time
	tol. (γ)	N	K/N	N	K/N	N	K/N	N	K/N	HP735 (sec)
DJ(1)SG	0.1	4	0	6	26.6	6	61.5	6	157.5	17.7
DJ(5)SG	0.1	4	0	5	11.8	6	27.0	6	54.0	6.7
DJ(1)MG	0.1	4	0	4	1.0	4	1.0	4	1.0	0.5
DJ(1)MG	0.01	4	0	3	1.67	3	2.0	3	2.0	0.5

DJ = damped Jacobi; (1) = number of sweeps, i.e., 1; MG=multigrid; SG=single grid;
N = Newton iterations; K/N = Krylov iterations per Newton iteration.

The initial guess, on grid 1, is a linear profile in U, and linear interpolation is used to move the approximate solution up in the mesh sequencing algorithm. Data is presented for two runs with a single grid preconditioner and for two runs with a multigrid preconditioner. The two single grid preconditioner runs are for one and five passes of DJ ($\omega = 0.5$) as a preconditioner (DJ(1)SG) and (DJ(5)SG), respectively, both with γ = 0.1. Here SG refers to single grid. The two multigrid preconditioner runs are both with DJ(1)MG as a smoother, one with $\gamma = 0.1$ and the other with $\gamma = 0.01$. One V(1,1) cycle is used as the multigrid preconditioner, and the prolongation operator, $\mathcal{P}$, is piecewise linear. Using uppercase subscripts for the coarse grid and lowercase subscripts for the fine grid, the interpolation operators are

$$\mathcal{R} : U_I = U_i + U_{i+1}, \tag{14}$$

$$\mathcal{P} : U_i = 0.75U_I + 0.25U_{I-1}, \quad U_{i+1} = 0.75U_I + 0.25U_{I+1}. \tag{15}$$

Note that separate coding is required for $\mathcal{P}$ on the boundaries.

One hundred Krylov vectors are stored (GMRES(100)) with one restart allowed. We do not consider this an option for practical problems, and we will consider the effects of restart on the final model problem. On the coarsest grid a direct linear solution is used, and a complete Jacobian (evaluated numerically) is used in the preconditioner as a reference point. In Table 1, we can see that the average number of Krylov iterations per Newton iteration, K/N, increases rapidly with grid dimension for the single grid preconditioner, while it is nearly flat for the multigrid preconditioner. K/N is scaling like $nx^{1.0}$ (i.e., optimal) for $\gamma = 0.1$ and like $nx^{1.13}$ for $\gamma = 0.01$.

While very much a model problem, these results give a clear illustration of the shortcoming of single grid preconditioners. Here, with multigrid preconditioning, we observe a dramatic reduction in K/N on grid 4, with a concomitant reduction in CPU time. Additionally, there is potentially significant savings in memory as a result of the small number of GMRES iterations required. While the multigrid preconditioner used here requires roughly twice the storage as the fine grid preconditioner (for one-dimensional), it requires significantly fewer GMRES iterations. All of the multigrid preconditioned computations could have been done by storing only five GMRES vectors.

Given the excellent performance of the multigrid preconditioner, it is logical to ask how the multigrid method works as a linear solver for this problem. Table 2 presents results of using our multigrid preconditioner as a stand-alone linear solver. Here we use a standard Newton method with multigrid as the linear solver (Newton-MG). This method requires the complete Jacobian (global linearization) to achieve

Table 2
Multigrid stand-alone solver performance, $c_3 = 0.25$.

Method	Inexact Newton tol.	Grid 1 nx = 50		Grid 2 nx = 100		Grid 3 nx =200		Grid 4 nx = 400		CPU time HP735 (sec)
		N	V/N	N	V/N	N	V/N	N	V/N	
DJ(1)MG	0.1	4	0	5	2.0	5	2.0	5	2.0	0.5
DJ(1)MG	0.01	4	0	3	3.67	3	3.67	3	3.67	0.4

DJ = damped Jacobi; (1) = number of sweeps, i.e., 1; MG = multigrid;
N = Newton iterations; V/N = V cycles per Newton iteration.

Table 3
Performance of preconditioning with diffusion operator only, NKMG, and Newton-MG implementation, DJ(1)MG.

Method	Inexact Newton tol. (γ)	Grid 1 nx=50		Grid 2 nx=100		Grid 3 nx=200		Grid 4 nx=400		CPU time HP735 (sec)
		N	L/N	N	L/N	N	L/N	N	L/N	
NKMG, c_3=0.25	0.01	5	1.0	4	1.25	4	1.5	3	1.33	0.5
Newton-MG, c_3=0.25	0.01	8	2.75	5	2.75	4	2.75	4	2.75	0.4
NKMG, c_3=0.05	0.01	5	3.0	4	3.25	3	4.0	3	4.0	0.7
Newton-MG, c_3=0.05	0.01	37	2.0	28	2.0	26	2.0	23	2.0	1.6
NKMG, c_3=0.025	0.01	6	5.0	4	6.5	4	6.5	4	7.5	1.1
Newton-MG, c_3=0.025	-	-	-	-	-	-	-	-	-	-

DJ = damped Jacobi; 1 sweep, i.e., DJ(1); 1 V cycle;
N = Newton iterations; L/N = linear iterations per Newton iteration.

Newton-like nonlinear convergence. The same inexact Newton convergence tolerance, (3), is used. Note that this is not an FAS method, and multigrid is used only on the linear problem. As can be seen, the multigrid method performs very well as a solver for this problem, and it is not exceedingly sensitive to the choice of γ. Recall that with the Newton-MG algorithm we are not afforded some of the flexibility available to us when using the matrix-free, NKMG method.

Next we demonstrate the ability to precondition this problem with the diffusion operator alone. Table 3 shows results for three values of $c_3 = 0.25$, 0.05, and 0.025 ($c_1 = 0.5, c_2 = -1.0$). For the NKMG algorithm, (7) is used to approximate matrix-vector products, while for Newton-MG the Jacobian simply ignores the contribution for the convection term. Of course, this will affect nonlinear convergence in Newton-MG. Table 3 demonstrates that while the linear convergence in NKMG is effected as c_3 is decreased, nonlinear convergence is rather insensitive. Just the opposite is true for Newton-MG, and convergence is not obtained for $c_3 = 0.025$. This demonstrates the ability to use an approximate Jacobian (only the linear diffusion operator) in the preconditioner for NKMG, while maintaining inexact Newton nonlinear convergence. To isolate the effect of the approximate Jacobian in the preconditioner one should compare the last row of Table 3 (complete Jacobian used to build preconditioner) to the first row of Table 3 (only diffusion operator used to build the preconditioner). It can be seen that the performance is quite similar.

Finally, in this problem, we demonstrate the ability to implement a higher-order convection scheme in the matrix-free Newton–Krylov algorithm, maintaining Newton-like nonlinear convergence, while not storing the additional entries in the stencil. The higher-order convection scheme used here is an upwind-biased, quadratic interpolation resulting in a five-point stencil. Dependence on the higher-order terms is achieved by

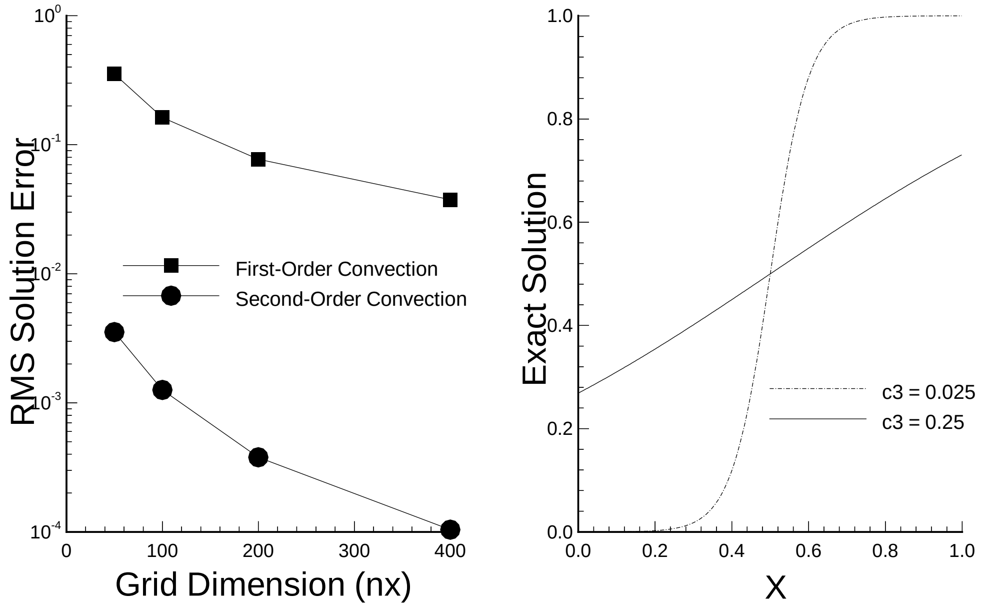

Fig. 2. (a) *Effect of higher-order differencing* (c_3=0.025) *on RMS error and* (b) *exact one-dimensional solution for* $c_3 = 0.25$ *and* 0.025 ($c_1 = 0.5, c_2 = -1.0$).

Table 4

Performance of higher-order differencing via matrix-free implementation, $c_3 = 0.025$, *DJ(1)MG.*

Method	Inexact Newton tol. (γ)	Grid 1 nx=50		Grid 2 nx=100		Grid 3 nx=200		Grid 4 nx=400		CPU time HP735 (sec)
		N	K/N	N	K/N	N	K/N	N	K/N	
$\mathbf{J}_{high}\mathbf{P}_{low}$	0.01	6	2.0	4	1.5	3	1.67	4	1.25	0.6
$\mathbf{J}_{low}\mathbf{P}_{low}$	0.01	5	0	4	1.5	4	1.75	3	1.67	0.5

damped Jacobi, 1 sweep, i.e., DJ(1), 1 V-cycle
N = Newton iterations, K/N = Krylov iterations per Newton iteration.

using (6) to approximate matrix-vector products in NKMG. An approximate Jacobian using the lower-order convection scheme (three-point stencil) is used to precondition this problem. Results are shown in Table 4 (for $c_3 = 0.025$) and in Figure 2. The results in Table 4 demonstrate that nonlinear convergence and CPU time are not affected by this approximation when compared to the nonlinear convergence and CPU time of the low-order solution. Thus, by using (6) we were able to obtain strong nonlinear convergence on this problem without forming a global linearization from the higher-order convective differencing. The root mean square (RMS) of the solution error, $\| (U - U_{exact})/U_{exact} \|_2$, is shown in Figure 2 for both discretizations. The advantage of the higher-order method is evident.

4.2. The two-dimensional Burgers equations. In this section we study algorithmic performance on the two-dimensional, two-component Burgers equations, where the dependent variables are U and V:

$$UU_x + VU_y - \frac{1}{Re}(U_{xx} + U_{yy}) = 0, \tag{16}$$

$$UV_x + VV_y - \frac{1}{Re}(V_{xx} + V_{yy}) = 0. \tag{17}$$

Analytical solutions exist for this problem [9] and are used to impose Dirichlet boundary conditions. Using the variables of [9] to describe the solution, our solution here corresponds to $a_0 = 110, a_1 = 110, a_2 = a_3 = 0, k = 5.0, \mathrm{Re} = 10, y_{max} = \pi/(6k)$. The dependent variables are

$$U = \frac{-2}{\mathrm{Re}}\frac{\phi_x}{\phi}, \quad V = \frac{-2}{\mathrm{Re}}\frac{\phi_y}{\phi} \tag{18}$$

and

$$\phi = a_0 + a_1 x + a_2 y + a_3 xy + \exp(k(x - x_0)) + \exp(-k(x - x_0))\cos(ky). \tag{19}$$

This is not convection dominated, but it is nonlinear and nonsymmetric. For simplicity, second-order central discretization is used for both convection and diffusion.

Figure 3 plots the RMS solution error as a function of grid dimension and Newton convergence on the 320 × 320 grid as a function of γ for this problem. It was found that $\parallel \mathbf{F}(\mathbf{u}^n) \parallel_2 < 1.0 \times 10^{-6}$ was required to observe second-order spatial convergence associated with central differencing. Thus, all simulations for this two-dimensional problem were run with this nonlinear tolerance.

First we compare the performance of single grid and multigrid preconditioned matrix-free Newton–Krylov with stand-alone multigrid as the linear solver for Newton-MG. In all cases SGS is used as the base smoother or preconditioner. While this is a two-component coupled system (U, V), only a point SGS smoother is used. A collective, or block, SGS smoother will be considered in the next problem. For the multigrid method the prolongation is piecewise linear, the restriction is piecewise constant, and a $V(2,2)$ cycle is used with 10 passes of the SGS smoother on the coarse grid (the 10 × 10 grid). When multigrid is used as a preconditioner (NKMG) one V cycle is used. Again, using uppercase subscripts for the coarse grid and lowercase subscripts for the fine grid, the interpolation operators are

$$\mathcal{R} : U_{I,J} = U_{i,j} + U_{i+1,j} + U_{i,j+1} + U_{i+1,j+1}, \tag{20}$$

$$\mathcal{P} : U_{i,j} = \frac{1}{16}[9U_{I,J} + 3U_{I+1,J} + 3U_{I,J+1} + U_{I+1,J+1}]. \tag{21}$$

Again, note that special coding is required for $\mathcal{P}$ on the boundaries.

Results are given in Table 5, where different algorithms are used with different inexact Newton convergence tolerances (γ). Again, as a reference point, we use the complete Jacobian to precondition the matrix-free methods, while it is required for Newton-MG. Here we see the expected growth in the number of Krylov iterations per Newton iteration with the single grid preconditioner (NKSG). For the NKMG we can see that the average number of Krylov iterations per Newton iteration (K/N) is held rather flat, scaling as $(2 \times nx \times nx)^{1.12}$ for $\gamma = 0.05$. It is also observed that K/N is not

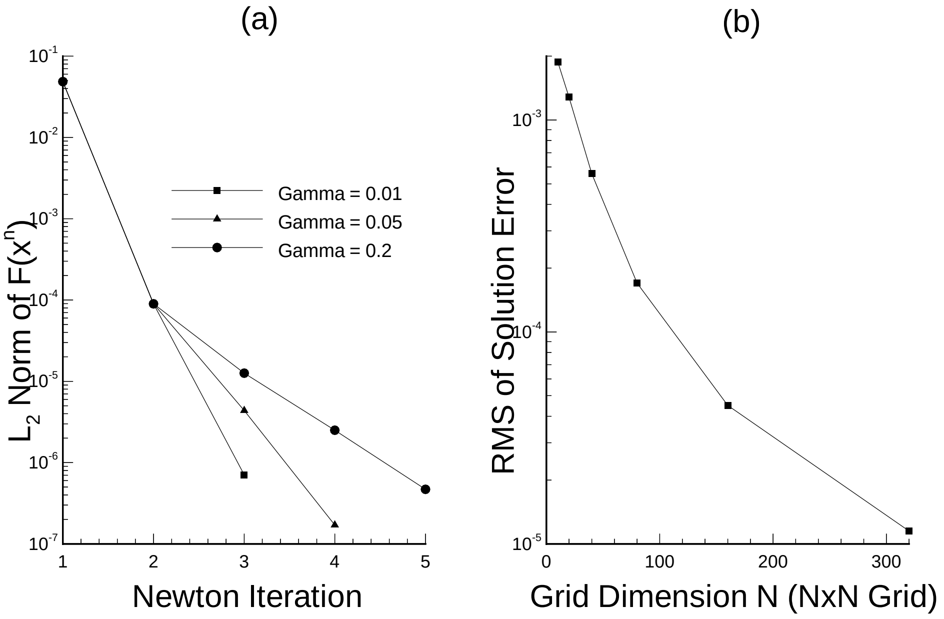

FIG. 3. (a) *Newton convergence on the* 320×320 *grid as a function of* γ*, the inexact Newton parameter and* (b) *RMS error as a function of grid.*

TABLE 5

Single grid (SG) vs. multigrid (MG) preconditioner performance for two-dimensional Burgers equations, $\mathcal{P}$ *is bi-linear, preconditioner is built from complete Jacobain.*

Method	Inexact Newton tol. (γ)	Grid 3 40×40		Grid 4 80×80		Grid 5 160×160		Grid 6 320 *times* 320		CPU time HP735 (min)
		N	L/N	N	L/N	N	L/N	N	L/N	
Newton-MG, SGS(2)	0.2	7	8.7	6	10.5	6	14.8	5	18	49
Newton-MG, SGS(2)	0.05	5	15.6	5	21.8	4	25.5	4	40.0	79
NKSG, SGS(2)	0.05	5	7	5	11.2	5	19.6	5	39.0	51
NKMG, SGS(2)	0.2	6	1.83	6	2.33	5	2.2	5	3.0	14
NKMG, SGS(2)	0.05	5	3.2	4	3.75	4	4.25	4	5.25	16
NKMG, SGS(2)	0.01	4	5.25	4	6.25	4	7.25	3	6.67	16

SGS = symmetric Gauss–Seidel; (2) = number of sweeps, i.e., 1; MG = multigrid;
N = Newton iterations; L/N = (Krylov iterations or V cycles) per Newton iteration.

very sensitive to γ. However, the stand-alone multigrid inside of Newton's method (Newton-MG) is sensitive to γ, and shows a steady increase in V cycles per Newton iteration as the grid dimension increases for $\gamma = 0.05$. For $\gamma = 0.2$, the Newton-MG algorithm shows less sensitivity as the grid dimension increases; however the total number of Newton iterations per grid has now increased, and thus the growth in the total number of V cycles to achieve the nonlinear convergence on each grid has remained fairly constant.

For the fine grid solution, 320×320 (204,800 unknowns), NKMG has provided a factor of 3 speed up in CPU time, compared to both the NKSG and Newton-MG, and almost an order of magnitude reduction in the required number of GMRES iterations per Newton iteration, as compared to NKSG. The Newton-MG method

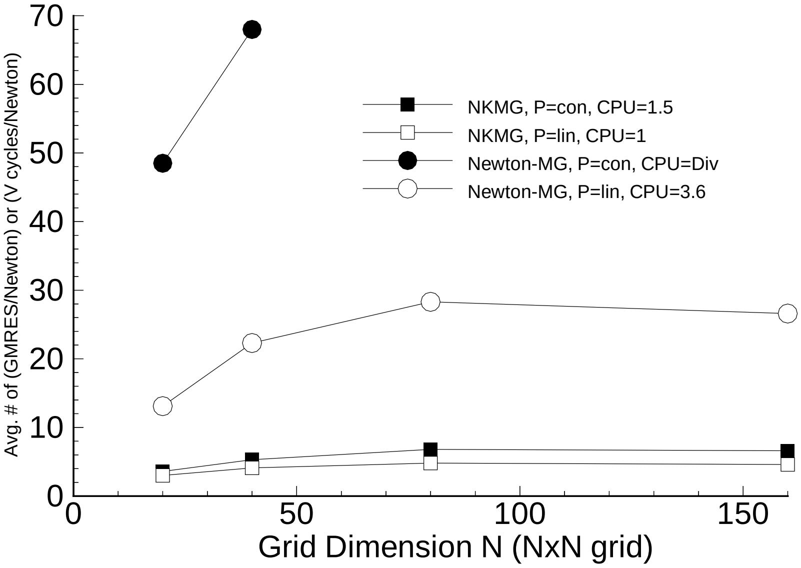

FIG. 4. *Comparison of NKMG and MG stand-alone on two-dimensional Burgers equations, using both piecewise constant, $\mathcal{P}_{con}$, and piecewise linear, $\mathcal{P}_{lin}$ prolongation operators* ($SGS(1)$).

also outperforms NKSG on this problem, in terms of CPU time, with $\gamma = 0.2$. Recall that for Newton-MG we require storage for the Jacobian and a complete Jacobian, while for NKSG and NKMG we require storage for the preconditioner and the Krylov vectors and only an approximate Jacobian is required to build the preconditioner.

Next, we demonstrate the effect of the accuracy used in prolongation $\mathcal{P}$. It is well known that to develop an optimal multigrid algorithm for a second-order equation requires either piecewise linear prolongation or piecewise linear restriction [30]. In the previous example we have used piecewise linear prolongation. Figure 4 demonstrates the result of using piecewise constant prolongation, which is given by

$$(22) \qquad \mathcal{P} : U_{i,j} = U_{I,J}, \;\; U_{i+1,j} = U_{I,J}, \;\; U_{i,j+1} = U_{I,J}, \;\; U_{i+1,j+1} = U_{I,J}.$$

Figure 4 shows that the NKMG is rather insensitive to the accuracy of the prolongation operator, while the Newton-MG (stand-alone multigrid) algorithm is sensitive to the accuracy of the prolongation operator. Thus it appears that the majority of the nonoptimality of the multigrid method with piecewise constant $\mathcal{P}$ has been removed via the outer GMRES iteration present in NKMG.

Finally, we demonstrate a capability of the matrix-free method on this two-dimensional problem. We will use only the diffusion operator in (16) and (17) to construct a preconditioning matrix. This is similar to the approach promoted in [5], but here we apply a multigrid-based preconditioner. This is a truly matrix-free method since the matrix from the diffusion operator is simple and does not need to be stored. Results are given in Table 6 for the linear and nonlinear iteration counts as a function of grid dimension. $\mathcal{P}$ is piecewise constant. Again, as was seen in one dimension, use of (7) to approximate matrix-vector products does not affect nonlinear convergence but can save significant memory. Comparing NKMG between Table 5

Table 6

Multigrid preconditioner performance for two-dimensional Burgers equations; $\mathcal{P}$ is piecewise constant. Preconditioner built from diffusion operator only.

Method	Inexact Newton tol. (γ)	Grid 3 40 × 40		Grid 4 80 × 80		Grid 5 160 × 160		Grid 6 320 × 320		CPU time HP735 (min)
		N	K/N	N	K/N	N	K/N	N	K/N	
NKMG, SGS(2)	0.1	5	3.4	6	3.67	5	3.4	5	4.6	17.5
NKMG, SGS(2)	0.05	5	4.4	4	5.5	4	6.0	4	6.0	16.5
NKMG, SGS(2)	0.01	4	6.25	4	7.75	4	8.75	3	8.3	19

SGS = symmetric Gauss–Seidel; (2) = number of sweeps, i.e., 2;
N = Newton iterations; K/N = Krylov iterations per Newton iteration.

(complete Jacobian in preconditioner) and Table 6 (only diffusion operator in preconditioner) for $\gamma = 0.05$ and 0.01, we see identical nonlinear iteration counts. There is a small increase in K/N when only the diffusion operator is used in the preconditioner. Also note that $\mathcal{P}$ is piecewise constant in Table 6. In this example, no matrix is stored, neither **J** or **P**, and thus the main storage requirement is the Krylov vectors of GMRES.

Certainly this problem should not be considered a challenge for FMG-FAS. However, it is another demonstration of how a simple multigrid method makes an effective preconditioner for the Newton–Krylov method. Using piecewise constant restriction and prolongation and only the diffusion terms in the preconditioner still results in an NKMG method which scales well under grid refinement.

4.3. Two-dimensional Navier–Stokes equations. As a final test of the performance of our proposed methods we will solve the two-dimensional incompressible Navier–Stokes equations in stream function-vorticity (ψ, ω) formulation. This is a coupled set of two PDEs, in Cartesian (x, y) coordinates:

$$\nabla^2 \psi = \omega, \tag{23}$$

$$\nabla \cdot (\vec{V}\omega) - \frac{1}{Re}\nabla^2 \omega = 0, \tag{24}$$

$$\vec{V} = V_1\hat{x} + V_2\hat{y}; \quad V_1 = \psi_y, \;\; V_2 = -\psi_x. \tag{25}$$

Note that we solve the vorticity equation in conservative form.

The standard driven cavity [10, 28] in the unit square is used as the model problem. The boundary conditions for this problem are $\psi = 0$ on all boundaries and the normal derivative of ψ is equal to zero on all boundaries except $y = 1$, where the normal derivative of ψ is set equal to one. This last boundary condition is a statement of the top wall moving at a velocity of one. This set of boundary conditions results in the boundary conditions on ω being a local function of ψ; see [10]. Cell-centered finite volume discretization is used with either second-order central differencing for the convection or with a second-order upwind biased, quadratic interpolation for convection. A few results will be given with first-order accurate upwind discretization of convection. When the second-order, upwind-biased, quadratic interpolation is used for convection it is used only to approximate the matrix-vector product in NKMG (i.e., (6)), and the first-order upwind scheme is used in the multigrid preconditioner.

All simulations are started on a 10×10 grid, and a mesh sequencing algorithm is used to move up to either a 160×160 grid or a 320×320 grid.

In keeping with the spirit of [5], the first results on this problem will use a simple linear (coupled) part of this system as the preconditioner, the discrete form of

$$\nabla^2 \psi = \omega, \tag{26}$$

$$-\frac{1}{Re}\nabla^2 \omega = 0. \tag{27}$$

Because of its simplicity, no storage is required for our multigrid preconditioner. However, storage will be required for a matrix factorization-based preconditioner, ILU(0). We compare the performance of NKMG to an NKSG method with a standard ILU(0) preconditioner [20]. Our ILU(0) is not block-based. The multigrid preconditioner applied here is over-simplified and does not work as a stand-alone solver for this problem; thus the Newton-MG algorithm is not included in the comparison. The multigrid smoother is a block SGS smoother, where the block is a 2×2 coupling (ψ, ω) in each finite volume. This is also referred to as a collective SGS method [30]. Piecewise constant restriction and prolongation are used, along with a standard V cycle with a fixed number of pre- and post-smoothing operations. With improvements to the above smoother, the cycle and the intergrid transfer operators multigrid has been highly successful in solving this problem [30, 10, 28, 22].

First we wish to investigate if the above simple multigrid method, without improvements, is a more effective preconditioner than ILU(0). This question is asked on a relatively low Reynolds number problem where ignoring the convection terms in the preconditioner is inconsequential. We expect this preconditioner to degrade with an increasing Reynolds number. For $Re = 100$ we compare the performance of our multigrid preconditioner to ILU(0). We use a nonlinear convergence tolerance of $\| \mathbf{F}(\mathbf{u}^n) \|_2 < 1.0 \times 10^{-6}$ and an inexact Newton coefficient of $\gamma = 0.01$. For this problem the multigrid preconditioner is one $V(3,3)$ cycle. On the coarse grid (10×10) 15 passes of the block SGS smoother are used.

Figure 5 plots the average number of GMRES iterations per Newton iteration as a function of grid dimension for five different solution methods. It can be seen that the simple multigrid-based preconditioner significantly outperforms ILU(0) as the grid is refined. Furthermore, this allows us to store less GMRES vectors. While restart is employed on this problem, allowing 200 total GMRES iterations, its success is limited. We employ the standard restarting algorithm referred to as Algorithm 6.11 in [26]. In terms of normalized CPU time for a converged solution on the 160×160 grid we have NKMG, GMRES(20) = 1.0 (17 min 34 sec), NKMG, GMRES(10) = 2.6, NK-ILU(0), GMRES(100) = 3.1, NK-ILU(0), GMRES(50) = 2.7, NK-BSGS(3), GMRES(50) = 5.3. Only NKMG, GMRES(20) converged in a reasonable time on the 320×320 grid, CPU = 5.6. On the 160×160 grid, NKMG would not converge with less than 10 GMRES vectors stored while NK-ILU(0) would not converge with less than 50 GMRES vectors stored. We would expect a block-based ILU(0) to outperform our pointwise ILU(0), but we would not expect it to scale with grid refinement as the multigrid preconditioner has done. It is worth noting here that the memory saved by not storing the matrix is approximately equal to 10 GMRES vectors. Storing the full preconditioner for this problem reduces the average number of GMRES iterations per Newton iteration by only 1 or 2.

As a final study we will simulate a higher Reynolds number and attempt to make some quantitative comparisons to the results for more standard nonlinear multigrid

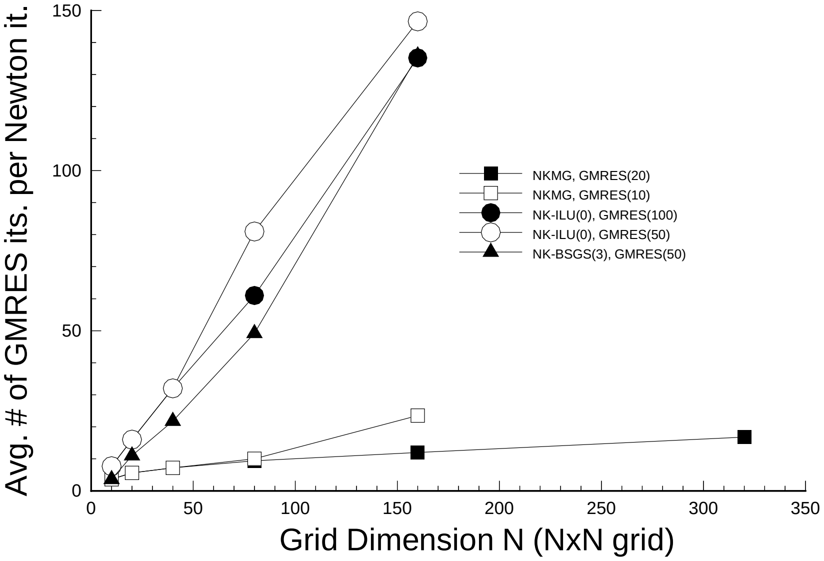

FIG. 5. *Comparison of NKMG and NK-ILU*(0) *on two-dimensional Navier–Stokes equations; varying number of GMRES vectors stored.*

approaches in [10, 28, 22]. For these results the full discrete equations will be used in the preconditioner with first-order upwind discretization used for the convection operator in nonconservative form. In conservative form, and with the higher-order upwind convection, the term $\nabla \cdot (\vec{V}\omega)$ in (24) requires a discretization stencil with nine values of ω and nine values of ψ. However, in the preconditioner we use the nonconservative form, $\vec{V} \cdot \nabla\omega$, and first-order upwinding which requires a discretization stencil with only five values of ω and five values of ψ. The resulting memory savings is approximately equal to storing 8 GMRES vectors; all of the following problems were run with GMRES(10). We retain the simple piecewise constant restriction and prolongation operators, along with a fixed $V(10,10)$ cycle and no special treatment on the coarse grid. In this final example the coarse grid operators are formed using the Galerkin approach, eliminating the need to restrict dependent variables down to coarse grids. This preconditioner could be added quickly to an existing flow solver based on a matrix-free Newton–Krylov method.

We review the basics of the methods in [10, 28, 22] so as to make our approximate comparisons somewhat more meaningful. In [10] the stream function-vorticity equations are solved using the FMG-FAS nonlinear multigrid method. Adaptive V cycles are used and the smoother is an ILU(0)-type method. By adaptive it is meant that switching from one grid to another is controlled by residual reduction; it is not done after a fixed number of smoothing steps. Thus the work per V cycle is not constant. Second-order accuracy of the convection operator is obtained via defect correction. Fullweighting is used for a restriction operator and bilinear interpolation is used for the prolongation operator. A false time step is used to aid in convergence. The solution convergence plot, Figure 6 of [10], will be our main point of comparison.

In [28] the driven cavity problem is solved in primitive variables on a staggered mesh using the FMG-FAS nonlinear multigrid method. Adaptive V cycles are used

Table 7
First-order convective discretization, nonlinear convergence tolerance of 1.0×10^{-3}.

Problem	Inexact Newton tol. (γ)	Grid 2 20 × 20		Grid 3 40 × 40		Grid 4 80 × 80		Grid 5 160 × 160		CPU time SUN Ultra-1 (sec)
		N	T.V.	N	T.V	N	T.V.	N	T.V.	
$Re = 100$	0.05	2	6	2	6	3	9	3	9	57
$Re = 1000$	0.05	3	13	2	6	2	6	2	6	54
$Re = 5000$	0.05	4	20	3	15	2	9	2	8	54

N = Newton iterations; T.V. = total V cycles.

Table 8
First-order convective discretization, nonlinear convergence tolerance of 1.0×10^{-5}.

Problem	Inexact Newton tol. (γ)	Grid 2 20 × 20		Grid 3 40 × 40		Grid 4 80 × 80		Grid 5 160 × 160		CPU time SUN Ultra-1 (sec)
		N	T.V.	N	T.V	N	T.V.	N	T.V.	
$Re = 100$	0.05	3	11	3	11	4	17	4	22	130
$Re = 1000$	0.05	4	18	4	22	4	29	4	34	183
$Re = 5000$	0.05	5	25	4	22	4	31	4	31	180

N = Newton iterations; T.V. = total V cycles.

along with a hybrid convection discretization which is first-order upwinding for grid Reynolds numbers greater than 2. The smoother is a symmetric block Gauss–Seidel. Averaging is used for the restriction operator and bilinear interpolation is used for the prolongation operator. Underrelaxation is used to aid convergence. The iteration count table, Table II in [28], will be our main point of comparison. It should be noted that convergence in both [10] and [28] was declared much earlier than our previous example and with a slightly different residual measure. Therefore we will look at the convergence of our method for a tolerance of $\sqrt{\sum(F_\psi)^2 + \sum(F_\omega)^2} < 1.0 \times 10^{-3}$, which is comparable to [10, 28], as well as a tolerance of $\sqrt{\sum(F_\psi)^2 + \sum(F_\omega)^2} < 1.0 \times 10^{-5}$.

Tables 7 and 8 present results using first-order upwind discretization in both the preconditioner and the matrix-vector multiply. We have presented this data so they can be more easily compared to those of [10, 28]. N is the number of Newton iterations required on that grid and T.V. is defined as the total V cycles required on a given grid (fine grid iterations). From Table II in [28] the reported number of fine grid iterations to reach the same level of convergence is 15, 19, and 52, for $\mathrm{Re} = 100, 1000, 5000$, respectively, while from our Table 7 the fine grid iterations are 9, 6 and 8, respectively. More insightful is the manner in which the NKMG CPU time scales with the Reynolds number for this convergence tolerance. Recall that in [28] an adaptive V cycle is used while we are using a fixed V cycle. As can be seen in Table II of [28] the work per V cycle increases with Reynolds number. In [28] the ratio between the CPU time to solve $Re = 5000$ and $Re = 100$ is 13.3, while the same ratio from our Table 7 is nearly 1. The results in Table 8 show that considerably more effort is required for the tighter convergence tolerance.

Figure 6 is a plot of the nonlinear residual on the 160×160 grid for $\mathrm{Re} = 100, 1000$, and 5000 using the higher-order convective discretization in the matrix-vector multiply (i.e., (6)). This figure should be compared to the convergence plot in [10], Figure 6, which is from a 128×128 grid. First, comparing back to our tables, we can see that for $\mathrm{Re} = 100$ and 1000 there is no noticeable effect in using the higher-order discretization in the matrix-vector multiply, the effect is more noticeable at $\mathrm{Re} = 5000$. We will discuss this shortly. Focusing on $\mathrm{Re} = 1000$ it can be seen that in six fine grid

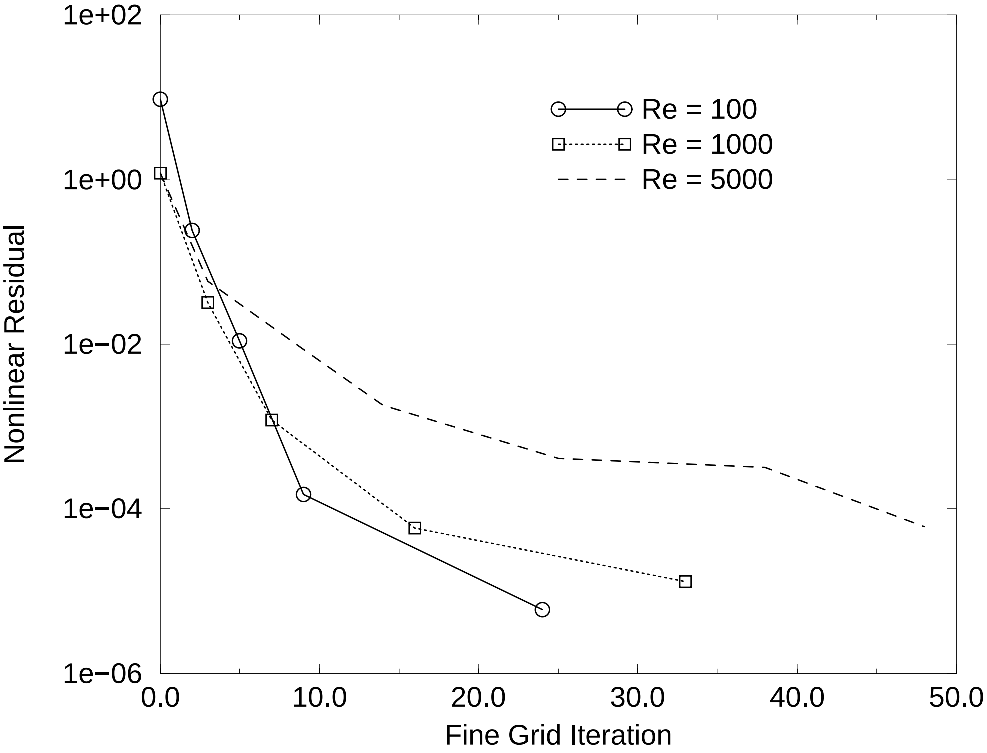

FIG. 6. *Convergence on* 160×160 *grid with second-order convection discretization in the matrix-vector multiply for* Re $= 100$*,* 1000*, and* 5000.

iterations (two Newton iterations) NKMG has surpassed the convergence level which required 30 fine grid iterations from the solver in [10]. With an equal number of fine grid iterations (30) NKMG has surpassed their convergence tolerance by 3 orders of magnitude. Recall that the fine grid iteration in [10] is an adaptive V cycle with a coupled ILU(0)-type smoother, while our fine grid iteration equates to a fixed $V(10,10)$ cycle with a block SGS smoother and one GMRES iteration. As for the performance of the Re = 5000 simulation in Figure 6 this can possibly be explained by (1) the simple multigrid preconditioner, (2) the use of (6), and (3) the use of a uniform grid instead of a stretched grid. Investigating this will be part of a future study.

Finally we attempt a comparison with the results from a more modern FMG-FAS solution algorithm for the driven cavity problem [22], which uses primitive variables on a colocated grid. In [22] a coupled line smoother is developed which works on the higher order convective discretization directly, avoiding the need for defect correction. Also in [22] the F cycle is used, fullweighting is used for restriction, and bilinear interpolation is used for prolongation [30]. The F cycle with a coupled line smoother will be more expensive then our fixed V cycle and block SGS smoothing. Thus we will attempt to make a CPU comparison, trying only to determine if our method is at least competitive with [22]. Table 7 in [22] lists wall clock times for their method on a uniform grid with Re = 1000 and an RS6000 workstation. Since they use primitive variables they have 3 dependent variables per grid point where we have only 2. Additionally, [22] monitors more of an L_{inf} norm of the residual which relates

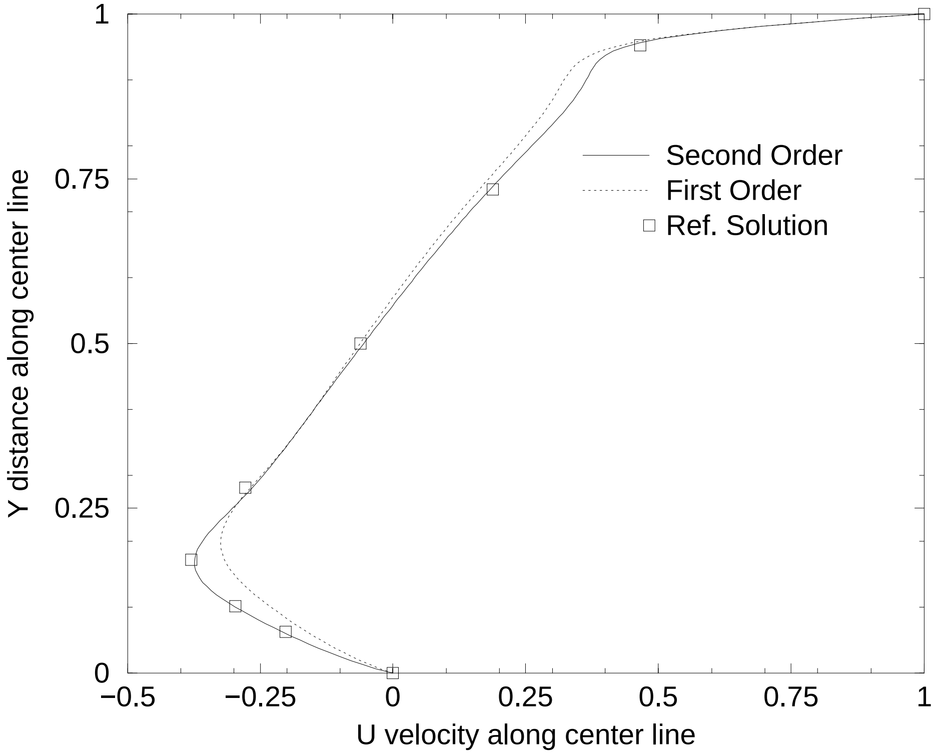

FIG. 7. *Comparison of first- and second-order convection discretization for* $Re = 1000$.

to $|F_\psi|_{max} + |F_\omega|_{max}$. For our problems, when $\sqrt{\sum(F_\psi)^2 + \sum(F_\omega)^2} < 1.0 \times 10^{-5}$ we have $|F_\psi|_{max} + |F_\omega|_{max} < 1.0 \times 10^{-6}$. The CPU times from Table 7 in [22] are 58, 156, and 284 seconds for 64^2, 128^2, and 192^2 grids, respectively. For similar convergence requirements our CPU times are 10, 50, and 195 seconds for 40^2, 80^2, and 160^2 grids, respectively (on the Sun ULTRA-1). We draw only one conclusion from this comparison, and that is that NKMG, with simple multigrid preconditioning, appears to be competitive with a modern FMG-FAS method. The center line velocity, V_1, for the first- and second-order accurate convection discretizations on the 160×160 uniform grid are given in Figure 7, along with the reference solution from [10].

This model problem has demonstrated the ability of a simple multigrid-based preconditioner to limit the growth in Krylov iterations per Newton iteration as compared to a standard ILU(0) preconditioner. This has provided both a savings in CPU time and in memory requirements. Additionally, NKMG has been demonstrated to be competitive with FMG-FAS solution methods for this problem.

5. Conclusions.

We have presented the details of a proposed new nonlinear multilevel iterative algorithm. Some of the unique capabilities of this method have been highlighted. Performance results of the algorithm have been presented on three nonlinear, nonsymmetric, boundary value problems. It has been our goal to answer the question, *Can a simple multigrid method provide effective preconditioning for Newton–Krylov methods?*

Initial results with nonlinear convection-diffusion boundary value problems are promising. The combination of GMRES and multigrid preconditioning (NKMG) outperformed single grid preconditioned Newton–Krylov (NKSG) and standard Newton's method with multigrid as the linear solver (Newton-MG) on most problems considered. The Newton-MG methods compared against were not based on the best known choices for smoothers and intergrid transfer operators. NKMG performance was shown to be less sensitive to the choice of prolongation operator, as compared to the stand-alone multigrid considered. It was also shown to provide a significant level of flexibility in defining the relaxation matrix (preconditioner) while maintaining strong nonlinear convergence. It was clearly demonstrated that our simple multigrid preconditioner outperformed ILU(0) in terms of memory requirements, iteration count, and CPU time on the two-dimensional Navier–Stokes problem. Additionally, comparisons with published results on the driven cavity problem have shown NKMG with simple multigrid preconditioning to be competitive with FMG-FAS solution methods.

Acknowledgment. The authors wish to acknowledge many useful discussions with Joel Dendy as well as his suggestions on the presentation of this material.

REFERENCES

[1] S. Ashby and R. Falgout, *A parallel multigrid preconditioned conjugate gradient algorithm for groundwater flow systems*, Nuclear Sci. and Engrg., 124 (1996), p. 145.

[2] A. Brandt, *Multi-level adaptive solutions to boundary value problems*, Math. Comp., 31 (1977), p. 333.

[3] P. N. Brown, *A local convergence theory for combined inexact Newton/finite-difference projection methods*, SIAM J. Numer. Anal., 24 (1987), pp. 407–434.

[4] P. N. Brown and A. C. Hindmarsh, *Matrix-free methods for stiff systems of ODEs*, SIAM J. Numer. Anal., 23 (1986), pp. 610–638.

[5] P. N. Brown and Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 450–481.

[6] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[7] J. Dendy, *Black box multigrid*, J. Comput. Phys., 48 (1982), p. 366.

[8] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in a inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.

[9] C. Fletcher, *Generating exact solutions of the two-dimensional Burgers' equations*, Internat. J. Numer. Methods Fluids, 3 (1983), pp. 213–216.

[10] U. Ghia, K. N. Ghia, and C. T. Shin, *High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method*, J. Comput. Phys., 48 (1982), pp. 387–411.

[11] R. Kettler, Ph.D. thesis, University of Technology, Delft, The Netherlands, 1987.

[12] D. Keyes, *Aerodynamic applications of Newton-Krylov-Schwarz solvers*, in Proceedings of the 14th Conference on Numerical Methods in Fluid Dynamics, Springer, Berlin, 1995, pp. 1–20.

[13] D. Knoll, *An improved convection scheme applied to recombining divertor plasma flows*, J. Comput. Phys., 142 (1998), pp. 473–488.

[14] D. Knoll and P. McHugh, *Enhanced nonlinear iterative techniques applied to a nonequilibrium plasma flow*, SIAM J. Sci. Comput., 19 (1998), pp. 291–301.

[15] D. Knoll, P. McHugh, and D. Keyes, *Newton-Krylov methods for low Mach number compressible combustion*, AIAA J., 34 (1996), pp. 961–967.

[16] D. Knoll, P. McHugh, and V. Mousseau, *Newton-Krylov-Schwarz methods applied to the tokamak edge plasma fluid equations*, in Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering, D. Keyes, Y. Saad, and D. Truhlar, eds., SIAM, Philadelphia, PA, 1995.

[17] P. McHugh and D. Knoll, *Fully implicit finite volume solutions of the incompressible Navier-Stokes and energy equations using inexact Newton's method*, Internat. J. Numer. Methods Fluids, 18 (1994), pp. 439–455.

[18] P. McHugh and D. Knoll, *Inexact Newton's method solution to the incompressible Navier-Stokes and energy equations using standard and matrix-free implementations*, AIAA J., 32 (1994), pp. 2394–2400.

[19] P. McHugh, D. Knoll, and D. Keyes, *Application of a Newton-Krylov-Schwarz algorithm to low Mach number combustion*, AIAA J., 36 (1998), pp. 290–292.

[20] J. Meijerink and H. van der Vorst, *An iterative solution method for linear linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), p. 148.

[21] J. C. Meza and R. S. Tuminaro, *A multigrid preconditioner for the semiconductor equations*, SIAM J. Sci. Comput., 17 (1996), p. 118–132.

[22] C. Oosterlee, F. J. Gaspar, T. Washio, and R. Wienands, *Multigrid line smoothers for high-order upwind discretizations of convection dominated problems*, J. Comput. Phys., 139 (1998), pp. 274 –307.

[23] C. W. Oosterlee and T. Washio, *An evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems*, SIAM J. Sci. Comput., 19 (1998), pp. 87–110.

[24] M. Pernice and H. F. Walker, *NITSOL: A Newton iterative solver for nonlinear systems*, SIAM J. Sci. Comput., 19 (1998), pp. 302–318.

[25] W. J. Rider, D. B. Kothe, S. J. Mosso, J. H. Cerruti, and J. I. Hochstein, *Accurate solution algorithms for incompressible multiphase fluid flows*, Tech. report AIAA 95–0699, AIAA, 1995; presented at the 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, 1995.

[26] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.

[27] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), p. 856–869.

[28] S. Vanka, *Block-implicit multigrid solution of Navier-Stokes equations in primitive variables*, J. Comput. Phys., 65 (1986), pp. 138–158.

[29] T. Washio and C. Oosterlee, *Krylov subspace acceleration for nonlinear multigrid schemes*, Electronic Trans. Numer. Anal., 6 (1997), pp. 271–290.

[30] P. Wesseling, *An Introduction to Multigrid Methods*, John Wiley, Chichester, UK, 1992.