Journal of Computational Physics **151,** 728–756 (1999)
Article ID jcph.1999.6216, available online at http://www.idealibrary.com on IDEAL®

A Robust Multigrid Algorithm for the Euler Equations with Local Preconditioning and Semi-coarsening

D. L. Darmofal* and K. Siu†

*Aeronautics & Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139; and †Department of Aerospace Engineering, Texas A & M University, College Station, Texas 77843
E-mail: darmofal@mit.edu and ksiu@aero.tamu.edu

Received September 14, 1998; revised January 21, 1999

A semi-coarsened multigrid algorithm with a point block Jacobi, multi-stage smoother for second-order upwind discretizations of the two-dimensional Euler equations which produces convergence rates independent of grid size for moderate subsonic Mach numbers is presented. By modification of this base algorithm to include local preconditioning for low Mach number flows, the convergence becomes largely independent of grid size and Mach number over a range of flow conditions from nearly incompressible to transonic flows, including internal and external flows. A local limiting technique is introduced to increase the robustness of preconditioning in the presence of stagnation points. Computational timings are made showing that the semi-coarsening algorithm requires $O(N)$ time to lower the fine grid residual six orders of magnitude, where N is the number of cells. By comparison, the same algorithm applied to a full-coarsening approach requires $O(N^{3/2})$ time, and, in nearly all cases, the semi-coarsening algorithm is faster than full coarsening with the computational savings being greatest on the finest grids.

Key Words: convergence acceleration; multigrid; preconditioning; Euler equations.

1. INTRODUCTION

An important aspect of any computational method is robustness. A robust computational method not only solves a given class of problems but does so in a reliable, predictable manner from case to case. In practice, robustness issues can arise in many different ways. For example, an algorithm which provides accurate answers in a reasonable amount of time for one case may suddenly require a significant amount of time to arrive at the same level of accuracy for a different case. Another common (and perhaps worse) difficulty is a typically reliable algorithm which simply diverges and fails to return any useful information for a particular problem. In either situation, these algorithms exhibit non-robust behavior.

0021-9991/99 $30.00

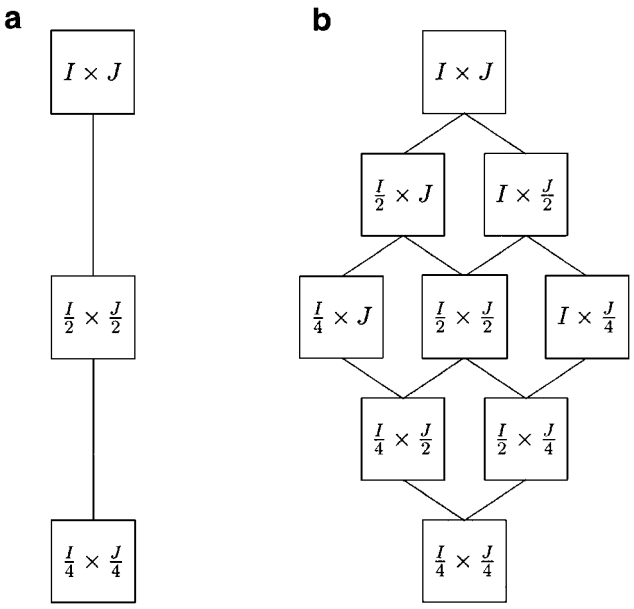

FIG. 1. Multigrid coarsening strategies. (a) Full coarsening and (b) semi-coarsening.

In this paper, we present a robust method for solving second-order upwind discretizations of the steady, two-dimensional Euler equations based on semi-coarsening multigrid, point block Jacobi multi-stage relaxation, and low Mach number local preconditioning. As we will show, the proposed algorithm is very robust, with its convergence rate being nearly independent of grid size and Mach number for both internal and external flows.

A robust multigrid algorithm requires the careful matching of the coarsening strategy with the iterative scheme or smoother. In particular, the smoother must effectively damp any modes which cannot be represented on coarser grids (without aliasing) [1, 2]. The most common coarsening strategy for multigrid on structured grids is full coarsening, in which every other point is removed in both directions as illustrated in Fig. 1. For typical upwind discretizations, the use of full coarsening generally requires an implicit relaxation in order to produce grid-independent convergence rates. This approach has been followed by several researchers, including Euler [3–7] and Navier–Stokes [8, 9] applications.

The necessity for implicit relaxation with full-coarsening multigrid arises from the existence of grid-aligned modes, i.e., error modes with long streamwise wavelengths but short cross-stream wavelengths. For upwind discretizations which do not introduce numerical dissipation normal to characteristics, these grid-aligned modes cannot be easily damped by simple point relaxations such as the multi-stage point block Jacobi relaxation algorithm employed in this work. The difficulty with damping grid-aligned error modes is a direct consequence of the upwind discretization; thus, schemes with more isotropic numerical dissipation [10] may not suffer these problems. However, schemes with more isotropic numerical dissipation are also likely to be less accurate [11–13]. In contrast to convection-dominated problems, point relaxation in combination with full-coarsening multigrid generally works well for elliptic problems with isotropic physical dissipation [1].

In this paper, we explore the use of semi-coarsening which is a more complex coarsening strategy devised by Mulder [14, 15] specifically for the Euler equations and other

convection-dominated flows. In semi-coarsening, a fine grid is associated with two coarser grids, each independently coarsened in a single grid direction. A typical family of semi-coarsened grids is shown in Fig. 1. While the semi-coarsening algorithm is more complex than full coarsening, the smoothing requirements are significantly reduced. In particular, a simple, point smoother is now sufficient for achieving smoothing of all fine grid error modes [14, 15, 2].

We select a point block Jacobi, multi-stage relaxation method as our smoother for a second-order upwind discretization. An advantage of point block Jacobi is its simplicity as it requires the inversion and storage of only the local block matrix arising from a linearization of the discrete equations. Also, point block Jacobi relaxation has been shown to be an effective smoother of high-high frequency errors for the 2-D discrete Euler [14, 15] and Navier–Stokes equations [2]. A multi-stage implementation of point block Jacobi is required for stability as a single-stage, damped Jacobi method is not stable for second-order discretizations. For the proposed semi-coarsening algorithm, convergence rates for moderate Mach numbers are nearly grid independent, implying that the total work for this algorithm is $O(N)$, where N is the number of cells. We note that similar $O(N)$ convergence for semi-coarsening and point block Jacobi has been previously observed by Mulder [15].

While semi-coarsening with point block Jacobi smoothing gives robust convergence rates for moderate Mach numbers, the performance severely degrades at lower Mach numbers. To alleviate this problem, we introduce the low Mach number preconditioning of Turkel [16] by modifying the upwind flux function [17] while retaining the point block Jacobi relaxation based on the modified fluxes. We refer to this approach as the preconditioned Jacobi. Mavriplis [18] and Turkel [19] have previously studied this method for incorporating low Mach number preconditioning.

Unfortunately, the full benefits of local preconditioning, especially at low Mach numbers, have been difficult to achieve because local preconditioners designed for good low Mach number performance inevitably have poor robustness at stagnation points [20, 21]. Darmofal and Schmid [21] have shown that this lack of robustness is due to unlimited transient amplification of perturbations stemming from a highly non-orthogonal (in fact, degenerate) eigenvector structure of the preconditioned equations as $M \to 0$. The most common technique for avoiding this robustness problem is based on limiting the effect of preconditioning below a multiple of the freestream Mach number. Since this multiple is typically greater than one, the limit often acts globally and thus destroys the locality of the preconditioning. Furthermore, for problems in which a reference Mach number is inappropriate or non-existent, this type of limiting will be difficult to realize. Examples of these types of flows would be a hypersonic flow about a blunt body (which would contain regions of subsonic flow) or flow of a high-speed jet into a stationary fluid.

In this paper, we develop of a new method for improving the robustness of local preconditioners. This new method relies on strictly local information and, as a result, acts locally to avoid transient amplification of perturbations. In addition, we show through numerical tests that the preconditioned block Jacobi algorithm is quite robust even without resorting to local preconditioning limiting. Thus, the overall robustness of the algorithm is a result of not only the preconditioner limiting technique but also the point block Jacobi smoothing. Utilizing the preconditioned Jacobi relaxation with semi-coarsening, convergence rates have only a small dependence on Mach number. Furthermore, $O(N)$ convergence is demonstrated over a wide range of Mach numbers, including $M_\infty \to 0$ for both internal and external flows.

2. NUMERICAL METHOD

2.1. *Discretization*

Employing a cell-centered, finite volume algorithm on a structured grid composed of quadrilateral cells, the two-dimensional Euler equations in semi-discrete form can be written as

$$A\frac{dU}{dt} + R = 0, \tag{1}$$

where A is the cell area and U is the cell-average conservative state vector defined as $U = (\rho, \rho u, \rho v, \rho E)$. The cell residual, R, is defined as

$$R = \sum_{k=1}^{4} \hat{H}_{n_k} \Delta s_k,$$

where Δs_k is the length and $\hat{H}_{n_k}$ is the numerical flux approximation for face k. Using an approximate Riemann solver (and dropping the subscript, k), $\hat{H}_n$ is given by

$$\hat{H}_n = \tfrac{1}{2}[H(U_\mathrm{L}) + H(U_\mathrm{R})] - \tfrac{1}{2}|\hat{\mathbf{A}}|(U_\mathrm{R} - U_\mathrm{L}),$$

where $\hat{\mathbf{A}} = \partial H_n / \partial U$ is the flux Jacobian evaluated using a Roe average [22]. For our second-order scheme, we approximate the left and right states, U_L and U_R, using van Leer's κ scheme [23]. This reconstruction is actually performed on the primitive variables, ρ, u, v, and p, instead of the conserved variables. For the results in this paper, we use $\kappa = 0$. Also, we did not limit the reconstruction; thus, near shocks, the solutions may not be monotonic.

Low Mach number preconditioning is incorporated into the algorithm by modifying the flux function,

$$\hat{H}_n = \tfrac{1}{2}[H(U_\mathrm{L}) + H(U_\mathrm{R})] - \tfrac{1}{2}\hat{\mathbf{P}}^{-1}|\hat{\mathbf{P}}\hat{\mathbf{A}}|(U_\mathrm{R} - U_\mathrm{L}), \tag{2}$$

where $\hat{\mathbf{P}}$ is a local flux preconditioner. This modification is required for stability and improves the accuracy for low Mach number flows [17]. Details for the implementation of the modified flux function are given in the Appendix.

For the flux preconditioner, we have chosen a form of Turkel's preconditioning [16]. This preconditioner takes on a particularly simple form when expressed in the symmetrizing variables, $d\tilde{U}^\mathrm{T} = [dp/\rho c, du, dv, dp - c^2 d\rho]$. In symmetrizing variables, the specific form of Turkel's preconditioner we use is a diagonal matrix,

$$\tilde{\mathbf{P}} = \begin{bmatrix} \epsilon & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Although not obvious, this preconditioner is identical to the preconditioner of Weiss and Smith [24]. $\tilde{\mathbf{P}}$ can be related to the preconditioner, $\hat{\mathbf{P}}$, appearing in the flux function through the similarity transformation

$$\tilde{\mathbf{P}} = \mathbf{M}\hat{\mathbf{P}}\mathbf{M}^{-1},$$

where $\mathbf{M}$ is the transformation matrix from the conserved variables to the symmetrizing variables, i.e., $d\tilde{U} = \mathbf{M}\, dU$. In Section 3, we discuss rationale for choosing ϵ, but we note here that we can return to the unpreconditioned flux by setting $\epsilon = 1$.

Boundary conditions have been implemented using a ghost cell approach. In particular, at solid surfaces, the normal velocity component is reflected while the density and pressure are taken directly from the first interior cell at the boundary. This boundary condition, while being quite robust, is lower order. Future research should investigate the effects of higher order boundary conditions on the convergence and robustness of the multigrid algorithm.

2.2. *Smoothing*

The smoothing algorithm is based on point block Jacobi relaxation and multi-stage integration. Specifically, we modify the semi-discrete Euler equation from Eq. (1) to include a cell residual preconditioner, $\mathbf{P}_c$, such that

$$A\frac{dU}{dt} + \mathbf{P}_c R = 0. \tag{3}$$

The role of the residual preconditioner is to guarantee good smoothing of high-high frequency error modes for the semi-coarsening multigrid algorithm. Good smoothing of high-high modes can be accomplished using a point block Jacobi preconditioner [2] which we approximate as

$$\mathbf{P}_c^{-1} = (1-\kappa)\frac{\Delta t_{\max}}{2A}\sum_{k=1}^{4} \hat{\mathbf{P}}_k^{-1}|\hat{\mathbf{P}}_k\hat{\mathbf{A}}_k|\Delta s_k.$$

For the case without low Mach number flux preconditioning, the preconditioner $\hat{\mathbf{P}}_k$ is eliminated from the expression for $\mathbf{P}_c$. Finally, a four-stage integration of Eq. (3) gives

$$\begin{aligned}
U^{(1)} &= U^n - \alpha^{(1)}\frac{\Delta t_{\max}}{A}\mathbf{P}_c^n R^n,\\
U^{(2)} &= U^n - \alpha^{(2)}\frac{\Delta t_{\max}}{A}\mathbf{P}_c^n R^{(1)},\\
U^{(3)} &= U^n - \alpha^{(3)}\frac{\Delta t_{\max}}{A}\mathbf{P}_c^n R^{(2)},\\
U^{n+1} &= U^n - \alpha^{(4)}\frac{\Delta t_{\max}}{A}\mathbf{P}_c^n R^{(3)}.
\end{aligned}$$

For the coefficients α_i, we use the optimal damping schemes of Lynn and Van Leer [25, 26], which are (0.203, 0.451, 0.906, 1.466) for full coarsening and (0.182, 0.412, 0.785, 1.401) for semi-coarsening. The preconditioner $\mathbf{P}_c$ is frozen at the first stage of an iteration, allowing an LU factorization and efficient back-solves to be performed on all subsequent stages. We note that the inclusion of $\Delta t_{\max}/A$ in $\mathbf{P}_c$ cancels with the same term in the multi-stage update. Thus, in practice, $\Delta t_{\max}/A$ is never calculated when using the block Jacobi preconditioner.

2.3. *Multigrid*

The semi-coarsening multigrid method was implemented using a full approximation storage (FAS) scheme [1] and follows the algorithm described by Mulder [14, 15] with a few exceptions. First, we solve the second-order discretization on all grids unlike Mulder [15],

who employs a defect correction strategy in which a first-order discretization is used on coarse grids. Use of defect correction may increase the efficiency of the method but was beyond the scope of the current work. Also, Mulder developed two anti-symmetric prolongation operators which he alternates on successive multigrid cycles. We have found this approach problematic, often producing non-monotonic convergence histories in conjunction with the switching between prolongation operators. Instead, we use a symmetric prolongation operator [27] such that the correction, ΔU, from two coarse grids to a fine grid is defined by

$$\Delta U = \mathcal{P}^I \Delta U_I + \mathcal{P}^J \Delta U_J - \tfrac{1}{2}\left(\mathcal{P}^J \mathcal{R}_J \mathcal{P}^I \Delta U_I + \mathcal{P}^I \mathcal{R}_I \mathcal{P}^J \Delta U_J\right),$$

where $\mathcal{P}^{I/J}$ are the I/J prolongation operators, $\mathcal{R}_{I/J}$ are the I/J restriction operators, and $\Delta U_{I/J}$ is the change on the I/J-coarsened grid. Linear interpolation is used for prolongation and area-weighted averaging for restriction. A V-cycle is used with two pre-smoothing and two post-smoothing iterations. The current algorithm starts from the finest grid with an initially uniform flow (i.e., impulsive start). We have also implemented a full-coarsening multigrid method for comparisons with semi-coarsening. A detailed description of the algorithms is given by Siu [28].

3. DEFINITION OF ϵ

As shown by Turkel [16], for good low Mach number preconditioning, the value of ϵ should be proportional to M^2; in this case, as $M \to 0$, the preconditioned eigenvalues all remain proportional to the flow speed. A useful measure of the effectiveness of preconditioning is the characteristic condition number, κ_g, defined as the ratio of largest to smallest propagation speeds for an isolated point disturbance [17] which is equivalent to the ratio of largest to smallest group velocities [29]. The optimal variation of ϵ which minimizes κ_g over all Mach numbers [28] is given by

$$\epsilon_{\text{opt}} = \begin{cases} 2M^2/(1-2M^2) & \text{for } M < 0.5, \\ 1 & \text{for } M \geq 0.5. \end{cases} \tag{4}$$

With a block Jacobi iterative scheme, we have found that slightly better convergence is given by ϵ of the form

$$\epsilon_{\text{cut}} = \begin{cases} M^2/\left(1-\alpha_{\text{cut}}^2 M^2\right) & \text{for } M < M_{\text{cut}}, \\ 1 & \text{for } M \geq M_{\text{cut}}, \end{cases} \tag{5}$$

where $\alpha_{\text{cut}}^2 = (1 - M_{\text{cut}}^2)/M_{\text{cut}}^2$ and M_{cut} is the user-defined Mach number above which no preconditioning is used. Specifically, for all of the preconditioned block Jacobi results in this paper, we use $M_{\text{cut}} = 0.5$. Figure 2 contains plots of κ_g for the Euler equations (without preconditioning), the Euler equations with block Jacobi (no flux preconditioning), and the Euler equations with preconditioned block Jacobi (Turkel flux preconditioning with $\epsilon = \epsilon_{\text{cut}}$ and $M_{\text{cut}} = 0.5$). The poor conditioning of the Euler equations is clearly seen at $M = 0$ and $M = 1$. Without any low Mach number preconditioning, block Jacobi has conditioning identical to that of the Euler equations until $M = 0.5$, above which block Jacobi is an improvement. The combination of block Jacobi preconditioning with flux preconditioning proves effective at removing low Mach number stiffness.

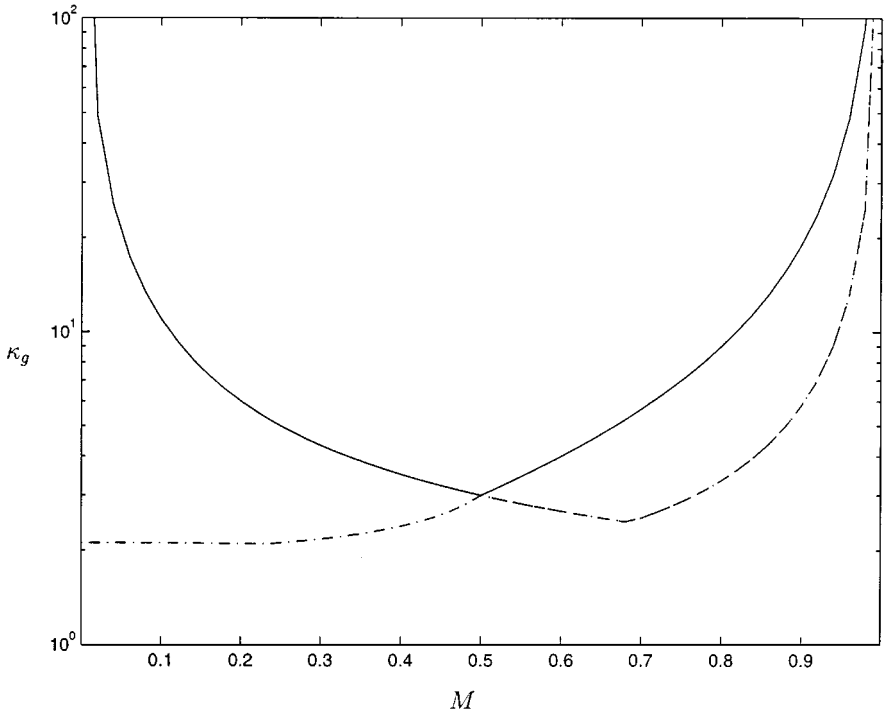

FIG. 2. Characteristic condition number, κ_g, for Euler (solid line), block Jacobi ($\epsilon = 1$, dashed line), and preconditioned block Jacobi ($\epsilon = \epsilon_{\text{cut}}$ and $M_{\text{cut}} = 0.5$, dash-dotted line).

Although the wave propagation stiffness is substantially improved at low Mach numbers using the flux preconditioning, a major source of trouble lies in the lack of robustness as $M \to 0$. Darmofal and Schmid [21] have shown that many local preconditioners can transiently amplify perturbations by a factor of $1/M$ as $M \to 0$. To demonstrate this, we consider the linearized, one-dimensional Euler equations in symmetrizing variables preconditioned by the Turkel preconditioner

$$\frac{\partial \tilde{U}}{\partial t} + \tilde{\mathbf{P}}\tilde{\mathbf{A}}\frac{\partial \tilde{U}}{\partial x} = 0,$$

where

$$d\tilde{U} = \begin{pmatrix} \frac{dp}{\rho c} \\ du \end{pmatrix}, \qquad \tilde{\mathbf{P}} = \begin{bmatrix} \epsilon & 0 \\ 0 & 1 \end{bmatrix}, \qquad \tilde{\mathbf{A}} = \begin{bmatrix} \bar{u} & \bar{c} \\ \bar{c} & \bar{u} \end{bmatrix}.$$

This can be Fourier transformed and solved for all wave numbers k,

$$\hat{U}(k, t) = G(k, t)\hat{U}_0(k),$$

where

$$G(k, t) = \exp(-ik\tilde{\mathbf{P}}\tilde{\mathbf{A}}t),$$

with $\hat{U}(k, t)$ being the Fourier-transformed state vector with wave number k and $\hat{U}_0(k)$ being the corresponding initial condition. Figure 3 is a plot of $\|G(k, t)\|$ versus time for $M = 0$ using different values of ϵ. As can be clearly seen, the potential for significant transient

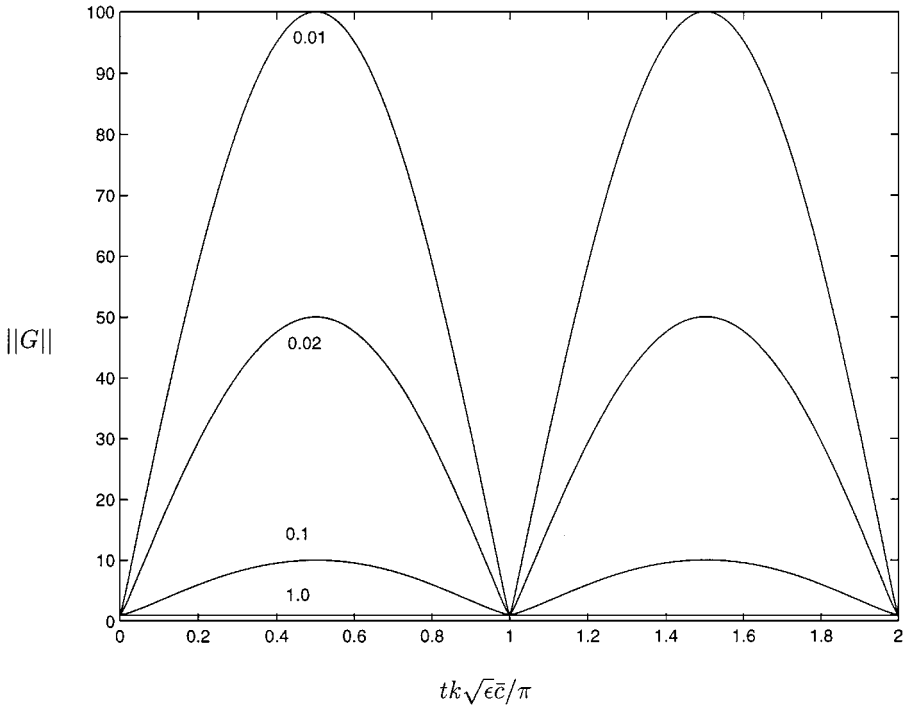

FIG. 3. Maximum amplification, $\|G\|$, versus time for $M = 0$ with varying $\sqrt{\epsilon}$.

growth exists at $M = 0$ as $\epsilon \to 0$. Specifically, the maximum growth over all time behaves as $G_{\max} = 1/\sqrt{\epsilon}$ at $M = 0$. Also, as shown in Fig. 3, the temporal evolution scales with wave number such that higher wave number modes will achieve maximum amplification in shorter times. These results suggest the need to limit ϵ such that it does not approach zero at stagnation points.

We can gain some additional insight into the transient growth for this preconditioner by calculating the initial perturbations which lead to the largest possible transient amplification $G_{\max}$. In fact, this worst-case disturbance is a mode with all of its initial energy in the pressure perturbation, i.e., $\hat{U}_0 \approx (1, 0)^{\mathrm{T}}$. Using this as the initial condition, the transient response of the Fourier-transformed pressure, $\hat{p}/\overline{\rho c}$, and velocity, $\hat{u}$, is calculated and shown in Fig. 4 for $M = 0$ and $\epsilon = 0.01$. As can be clearly seen, a unit-magnitude pressure disturbance creates an amplified velocity perturbation with magnitude $1/\sqrt{\epsilon} = 10$.

The typical approach for constructing a limit for ϵ is to require ϵ to be greater than some multiple of the freestream Mach number [21, 30, 31]. For example, if the desired value of $\epsilon = M^2$, the limited value would be

$$\epsilon = \max\left(M^2, \eta M_\infty^2\right).$$

With this limit, the amplification can still reach $1/(\sqrt{\eta} M_\infty)$, which could be significant at lower freestream Mach numbers. A typical value for $\eta = 3.0$ [31]. This approach works reasonably well for airfoil flows but the high value of η means that the limit is often active throughout the computational domain.

In developing a new ϵ limit, the key idea is to recognize that when perturbations are small, the maximum amplification can be large; however, for large perturbations, the maximum

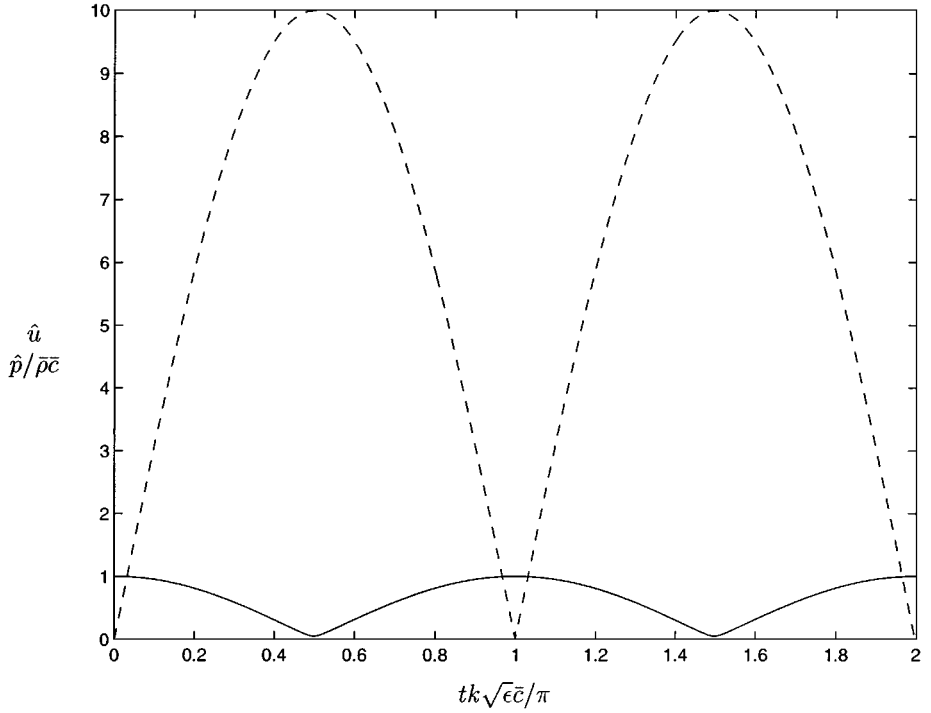

FIG. 4. Behavior of $\hat{p}/\overline{\rho c}$ (solid line) and $\hat{u}$ (dashed line) versus time for the most amplified initial condition with $M = 0$ and $\epsilon = 0.01$.

amplification must be small. This suggests making the limit a function of the local flow perturbations. The specific limit[1] which we have found to work successfully for a variety of flows is

$$\epsilon_{\text{lim}_{\text{new}}} = \frac{|\hat{p}_0(k)|}{\overline{\rho c^2}}. \tag{6}$$

As shown in the preceding analysis of the preconditioned system's transient growth, pressure perturbations are the most dangerous disturbances and result in velocity perturbations bounded by

$$|\hat{u}(k,t)| \leq \frac{1}{\sqrt{\epsilon}} \frac{|\hat{p}_0(k)|}{\overline{\rho c}}. \tag{7}$$

Thus, substituting $\epsilon_{\text{lim}_{\text{new}}}$ from Eq. (6) into Eq. (7), we may show that the velocity perturbation squared is bounded by

$$|\hat{u}(k,t)|^2 \leq \frac{|\hat{p}_0(k)|}{\bar{\rho}}.$$

This is reminiscent of the incompressible Bernoulli equation and suggests that the magnitude of velocity perturbations will correctly scale with pressure perturbations when ϵ is required to be greater than $\epsilon_{\text{lim}_{\text{new}}}$. This limit can also be interpreted as a linearity condition since it guarantees that the square of the velocity perturbation is less than the pressure perturbation.

[1] This limit came from a suggestion by Jonathon Weiss of Fluent to include pressure variations in the ϵ cutoff. The authors acknowledge his contribution.

In our preconditioning strategy, the values of ϵ are needed only during the flux calculation. These flux values of ϵ are called ϵ_{flux}. To determine ϵ_{flux}, a new value of ϵ is calculated for every face,

$$\epsilon_{\text{face}} = \min[1, \max(\bar{\epsilon}_{\text{L}}, \bar{\epsilon}_{\text{R}}, \epsilon_{\text{lim}})], \tag{8}$$

where $\bar{\epsilon}_{\text{L,R}}$ are the values of ϵ from Eq. (5) using the left and right cell-average states. When using the old limit,

$$\epsilon_{\text{lim}_{\text{old}}} = \eta\epsilon_{\infty},$$

where ϵ_{∞} is the value of ϵ evaluated at M_{∞} using Eq. (5). For the new limit, we approximate $|\hat{p}|$ by the difference in cell-average pressures, $|\bar{p}_{\text{R}} - \bar{p}_{\text{L}}|$, giving

$$\epsilon_{\text{lim}_{\text{new}}} = \frac{|\bar{p}_{\text{R}} - \bar{p}_{\text{L}}|}{\hat{\rho}\hat{c}^2}. \tag{9}$$

The new ϵ_{face} values are limited only with respect to pressure variations across the face; however, pressure gradients in all directions should be accounted for when limiting ϵ. Thus, the values of ϵ_{face} are sent to the cells with the cell values of ϵ being the maximum of the four face values which surround it,

$$\epsilon_{\text{cell}} = \max_{k=1}^{4} \epsilon_{\text{face}_k}. \tag{10}$$

Finally, the value ϵ_{flux} used in the flux calculations is the maximum value of ϵ_{cell} from the two cells surrounding a face,

$$\epsilon_{\text{flux}} = \max\left(\epsilon_{\text{cell}_{\text{L}}}, \epsilon_{\text{cell}_{\text{R}}}\right).$$

We note that the process of maximizing the ϵ over faces and cells as described above naturally raises the value of ϵ even without recourse to the new limit.

4. RESULTS

The convergence rates presented in the following results include cycle counts, work units, and CPU timings required to converge the solution six orders of magnitude from the initial residual. Convergence is measured using the RMS residual of all components of the residual vector (i.e., mass, momentum, and energy). A single work unit is equal to the amount of work required to evaluate the residual on the finest grid. Also, the total amount of work includes only the work required to perform smoothing passes on all of the grids but not any intergrid transfers.

4.1. *Bump Flow Results*

The first set of tests simulates flow over a solid bump between $0 \le x \le 1$ described by $y = 0.042 \sin^2(\pi x)$. The domain is 5 unit lengths long and 2 lengths high. The grid is structured with clustering toward the wall boundary. A sample grid and flow solution are shown in Fig. 5. Grid sizes range from 32×16 to 256×128.

The results for the Jacobi algorithm in Table I show a pronounced dependence on Mach number. In particular, at low Mach numbers, the convergence rates significantly degrade

TABLE I
Bump Flow with Jacobi Results

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
32×16	67	1492	46	2393
64×32	113	2551	64	3825
128×64	241	5457	73	4659
256×128	DNC	DNC	81	5338
		$M_\infty = 0.3$		
32×16	37	824	14	729
64×32	63	1422	14	838
128×64	102	2310	14	894
256×128	168	3808	14	923
		$M_\infty = 0.5$		
32×16	23	513	8	417
64×32	35	791	8	479
128×64	52	1178	8	512
256×128	101	2290	8	528
		$M_\infty = 0.8$		
32×16	28	624	10	521
64×32	42	949	10	599
128×64	48	1088	15	958
256×128	80	1814	16	1055

Note. Six orders drop in residual. DNC, did not converge in 300 cycles.

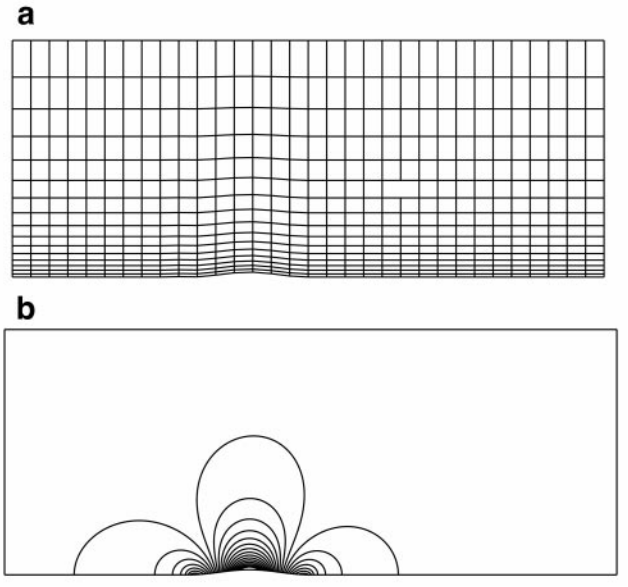

FIG. 5. Sample bump grid and C_p data. (a) 32×16 grid and (b) C_p contours, $M_\infty = 0.1$, preconditioned Jacobi.

TABLE II
Bump Flow with Preconditioned Jacobi Results

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
32×16	20	446	7	365
64×32	21	475	7	419
128×64	33	748	7	448
256×128	65	1474	7	462
		$M_\infty = 0.3$		
32×16	19	423	7	365
64×32	23	520	7	419
128×64	36	816	7	448
256×128	71	1610	7	462
		$M_\infty = 0.5$		
32×16	23	513	8	417
64×32	35	791	8	479
128×64	52	1178	8	511
256×128	101	2290	8	528
		$M_\infty = 0.8$		
32×16	28	624	10	521
64×32	42	949	10	599
128×64	48	1088	15	958
256×128	80	1814	16	1055

Note. Six orders of magnitude drop in residual.

from similar grid sizes at higher Mach numbers. For example, for a grid of 128×64 cells, the $M_\infty = 0.1$ case converges (i.e., the residual drops six orders of magnitude) in 252 full-coarsening cycles while the $M_\infty = 0.5$ case converges almost five times faster, needing only 52 full-coarsening cycles. For semi-coarsening, the results are even more dramatic with $M_\infty = 0.1$ and $M_\infty = 0.5$ converging in 73 and 8 cycles, respectively. While this degradation in convergence rate is observed most significantly at $M_\infty = 0.1$, the effect is also evident at $M_\infty = 0.3$.

By comparison, the results for the preconditioned Jacobi algorithm in Table II show very little dependence on Mach number for full and semi-coarsening with the lowest Mach number cases converging fastest. The semi-coarsening performance is particularly impressive, with the total range of cycles for all Mach numbers and all grids being only from 7 cycles for the $M_\infty = 0.1$ cases to 16 cycles for the finest grid, $M_\infty = 0.8$ case. We note also that for $M_\infty \geq 0.5$, the Jacobi and preconditioned Jacobi converge in almost exactly the same amount of cycles (or work). This result is expected since the preconditioning is turned off for $M \geq 0.5$ by the definition of ϵ in Eq. (5).

Another interesting aspect of the bump flow convergence results is the dependence of convergence rate on grid size for full and semi-coarsening. Tables I and II clearly show that full coarsening requires an increasing number of cycles (or work units) to converge with an increasing grid size for Jacobi and preconditioned Jacobi. In fact, for the largest grids, the total cycles or work units for convergence are increasing by almost exactly a factor of 2 for a factor of 4 increase in grid size. This suggests that the full-coarsening algorithm requires $O(N^{3/2})$

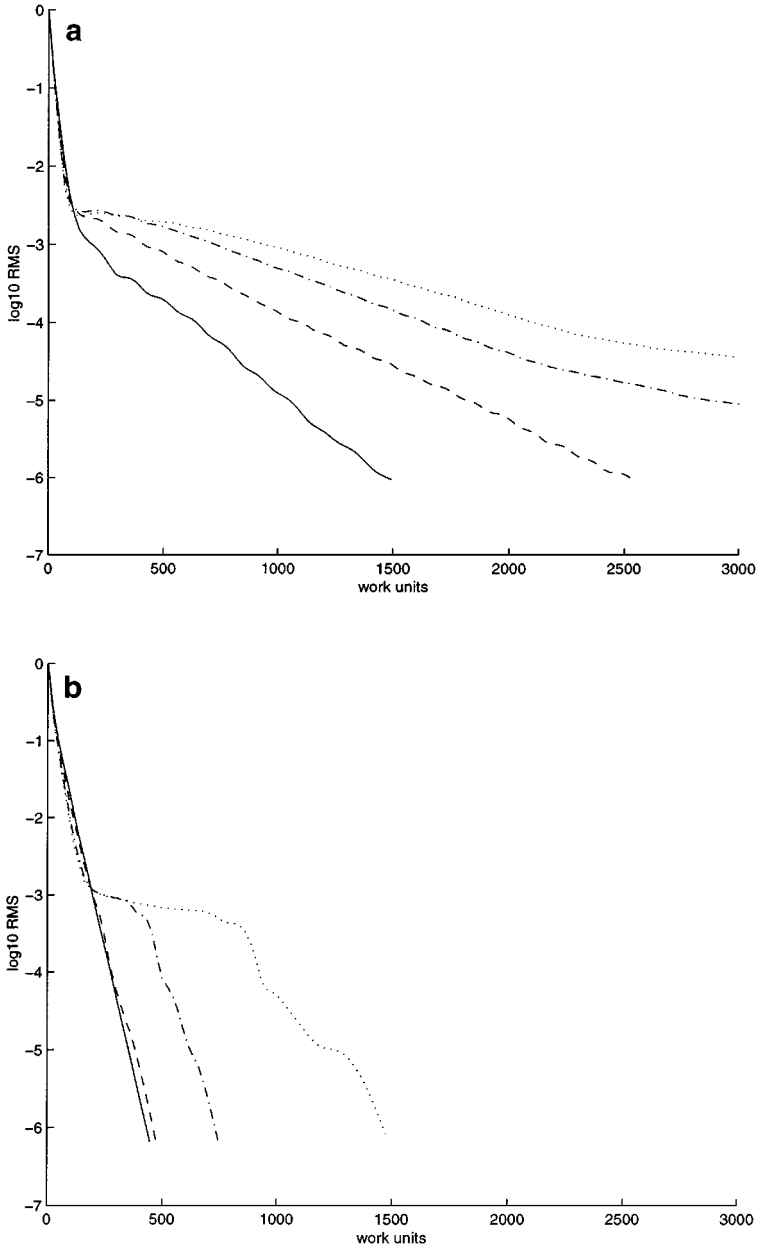

FIG. 6. Variation of convergence with grid size. Bump flow with full coarsening for $M_\infty = 0.1$. Solid, 32×16; dashed, 64×32; dash-dotted, 128×64; dotted, 256×128. (a) Jacobi and (b) preconditioned Jacobi.

operations to converge to a fixed level. Convergence histories for full coarsening with $M_\infty = 0.1$ are shown in Fig. 6. As described in the Introduction, the poor performance of the full-coarsening algorithm is attributable to the lack of damping for grid-aligned error modes.

The results for semi-coarsening are distinctly superior to those for full coarsening with respect to grid dependence. For Jacobi, the $M_\infty = 0.3$ and 0.5 cases are grid independent, requiring 14 cycles and 8 cycles to converge, respectively, for all grid sizes. At $M_\infty = 0.8$, the coarsest two grids converge in 10 cycles while the finer grids jump to 15 and 16 cycles. The

difficulty, we believe, lies with the presence of a shock wave in the steady solution. We have found this type of grid dependence for many problems with shocks. Several researchers have proposed modifications in the multigrid algorithm to better handle discontinuous flow variations; however, these have not been pursued for this work. For the low-speed $M_\infty = 0.1$ case, the Jacobi algorithm in conjunction with semi-coarsening is no longer grid independent, with the 32×16 grid requiring 46 cycles and the 256×128 grid requiring 81 cycles to converge (see the convergence histories in Fig. 7a). However, preconditioned

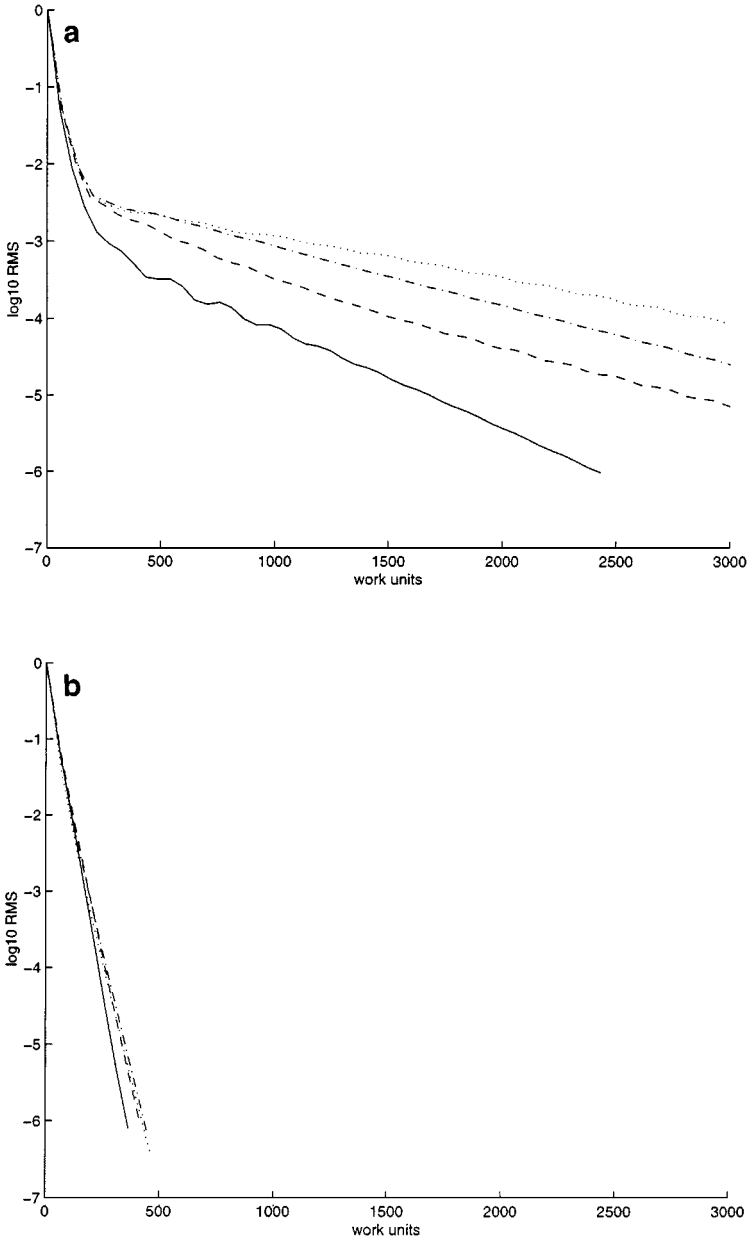

FIG. 7. Variation of convergence with grid size. Bump flow with semi-coarsening for $M_\infty = 0.1$. Solid, 32×16; dashed, 64×32; dash-dotted, 128×64; dotted, 256×128. (a) Jacobi and (b) preconditioned Jacobi.

Jacobi with semi-coarsening maintains grid independence at low Mach numbers. This is shown in Fig. 7b in which the convergence histories versus work units for all grid sizes are nearly identical. The beneficial effect of low Mach number preconditioning is even felt at $M_\infty = 0.3$. In comparison to the Jacobi algorithm which required 14 cycles to converge for all grids, the preconditioned Jacobi algorithm converges in 7 cycles for all grids, giving a factor of 2 improvement.

4.2. *Duct Flow Results*

A second set of cases was performed for duct flows. All conditions are the same as those for the bump tests except the upper boundary condition. The upper boundary of the domain is now a solid wall instead of a farfield. With the farfield boundary condition approach, error modes can propagate out of the domain. However, with a solid wall boundary on top, acoustic error modes will reflect back into the domain and could hinder with convergence. Results for Jacobi and preconditioned Jacobi duct cases are shown in Tables III and IV. All of the trends observed in the bump flow results are also evident in the duct flow cases.

4.3. *Airfoil Results*

The third set of tests simulates flow over a NACA 0012 airfoil. The grid sizes range from 96×16 to 384×128, and the farfield boundaries are 20 chord lengths away. The farfield boundary model is simply the uniform freestream although more accurate models could have

TABLE III
Duct Flow with Jacobi Results

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
32×16	67	1492	46	2393
64×32	113	2551	64	3825
128×64	240	5435	73	4659
256×128	DNC	DNC	80	5338
		$M_\infty = 0.3$		
32×16	37	824	14	729
64×32	64	1445	14	838
128×64	106	2401	14	894
256×128	182	4125	14	923
		$M_\infty = 0.5$		
32×16	23	513	9	469
64×32	37	836	8	479
128×64	62	1405	8	512
256×128	109	2471	8	528
		$M_\infty = 0.8$		
32×16	28	624	11	573
64×32	37	836	10	599
128×64	46	1042	10	639
256×128	68	1542	21	1385

Note. Six orders of magnitude drop in residual. DNC, did not converge in 300 cycles.

TABLE IV
Duct Flow with Preconditioned Jacobi Results

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
32×16	20	446	7	365
64×32	23	520	7	419
128×64	39	884	7	448
256×128	71	1610	7	462
		$M_\infty = 0.3$		
32×16	19	424	7	365
64×32	25	565	7	419
128×64	43	975	7	448
256×128	76	1723	7	462
		$M_\infty = 0.5$		
32×16	23	513	8	417
64×32	37	836	8	479
128×64	62	1405	8	511
256×128	109	2471	8	528
		$M_\infty = 0.8$		
32×16	28	624	11	573
64×32	37	836	10	599
128×64	46	1042	10	639
256×128	68	1542	21	1385

Note. Six orders of magnitude drop in residual.

been incorporated. A typical grid and a transonic solution are shown in Fig. 8. One set of grids contained 96×16, 192×32, and 384×64 cells. A second set of grids was generated by doubling the number of cells in the direction normal to the airfoil surface, giving 96×32, 192×64, and 384×128 cells. Clustering was used to allow better resolution of the flow properties in critical areas.

Convergence data for all NACA 0012 results are given in Tables V–VIII. The results follow the same trends observed with the bump and duct flows. For the Jacobi algorithm, the performance at $M_\infty = 0.1$ is extremely poor with all but the one case (semi-coarsening on the finest grid) failing to converge in 300 multigrid cycles. Furthermore, as illustrated in the convergence history plots in Figs. 9 and 10, many of these low Mach number solutions appeared to completely stall and never converge six orders of magnitude. As before, the low Mach number preconditioning completely alleviates this problem. The beneficial effect of preconditioned Jacobi is also observed at $M_\infty = 0.3$ with a factor of 2 or more improvement compared to Jacobi in most cases. At higher Mach numbers, the convergence of both Jacobi and preconditioned Jacobi is almost identical.

As observed with the bump and duct flow results, the number of cycles required by full coarsening to converge six orders of magnitude increases with increasing grid size. For the finer grids, the amount of work approximately doubles with a fourfold increase in grid size, implying an $O(N^{3/2})$ algorithm. Convergence histories for full coarsening with $M_\infty = 0.1$ are shown in Fig. 9. In both the Jacobi and preconditioned Jacobi results, the dependence of convergence on grid size can be easily observed. As described in the Introduction, the

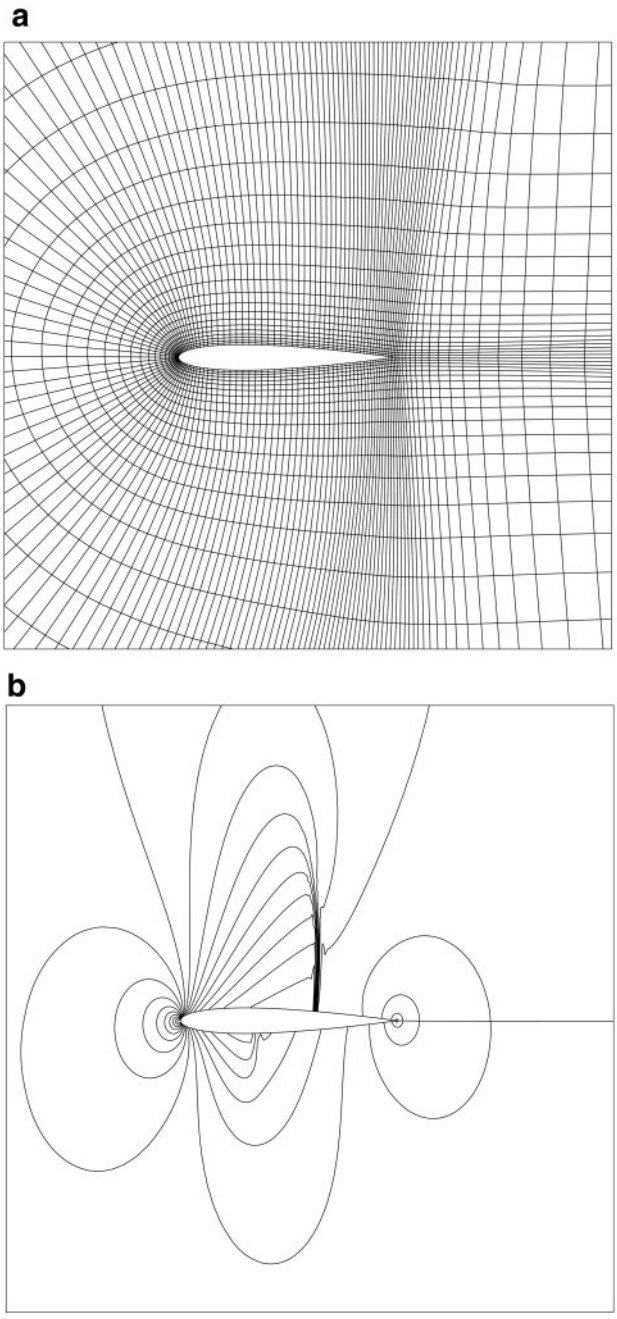

FIG. 8. Sample NACA 0012 grid and C_p data. (a) 192×32 grid and (b) C_p contours, $M_\infty = 0.8$, preconditioned Jacobi.

poor performance of the full-coarsening algorithm is attributable to the lack of damping for grid-aligned error modes.

At Mach numbers of 0.5 and 0.8, the semi-coarsening algorithm with Jacobi preconditioning performs nearly independent of grid size. At low Mach numbers, the Jacobi algorithm with semi-coarsening is no longer grid independent; however, the incorporation of low

TABLE V
NACA 0012 Flow with Jacobi Results for Set 1 Grids

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
96×16	DNC	DNC	DNC	DNC
192×32	DNC	DNC	DNC	DNC
384×64	DNC	DNC	DNC	DNC
		$M_\infty = 0.3$		
96×16	78	1737	27	1405
192×32	117	2641	29	1734
384×64	183	4144	29	1852
		$M_\infty = 0.5$		
96×16	47	1047	17	885
192×32	74	1671	15	897
384×64	131	2967	16	1022
		$M_\infty = 0.8$		
96×16	39	869	26	1353
192×32	61	1377	25	1495
384×64	107	2424	23	1469

Note. Six orders of magnitude drop in residual. DNC, did not converge in 300 cycles.

TABLE VI
NACA 0012 Flow with Preconditioned Jacobi Results for Set 1 Grids

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
96×16	35	780	18	937
192×32	51	1152	14	838
384×64	87	1971	14	894
		$M_\infty = 0.3$		
96×16	34	758	16	833
192×32	52	1174	13	778
384×64	87	1971	13	831
		$M_\infty = 0.5$		
96×16	39	869	17	885
192×32	67	1513	15	897
384×64	128	2899	16	1022
		$M_\infty = 0.8$		
96×16	38	847	26	1353
192×32	61	1377	26	1555
384×64	106	2401	23	1469

Note. Six orders of magnitude drop in residual.

TABLE VII
NACA 0012 Flow with Jacobi Results for Set 2 Grids

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
96×32	DNC	DNC	DNC	DNC
192×64	DNC	DNC	DNC	DNC
384×128	DNC	DNC	177	11664
		$M_\infty = 0.3$		
96×32	81	1829	24	1435
192×64	142	3216	31	1979
384×128	277	6278	33	2175
		$M_\infty = 0.5$		
96×32	59	1332	16	957
192×64	103	2333	17	1086
384×128	205	4646	17	1121
		$M_\infty = 0.8$		
96×32	59	1332	16	957
192×64	108	2446	17	1086
384×128	195	4420	21	1385

Note. Six orders of magnitude drop in residual. DNC, did not converge in 300 cycles.

TABLE VIII
NACA 0012 Flow with Preconditioned Jacobi Results for Set 2 Grids

	Full coarsening		Semi-coarsening	
Grid	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
96×32	48	1084	12	718
192×64	83	1880	12	767
384×128	147	3332	11	726
		$M_\infty = 0.3$		
96×32	50	1129	12	718
192×64	85	1925	11	703
384×128	156	3536	11	726
		$M_\infty = 0.5$		
96×32	55	1242	16	957
192×64	102	2310	17	1086
384×128	204	4624	17	1121
		$M_\infty = 0.8$		
96×32	59	1332	15	897
192×64	108	2446	16	1022
384×128	194	4397	20	1319

Note. Six orders of magnitude drop in residual.

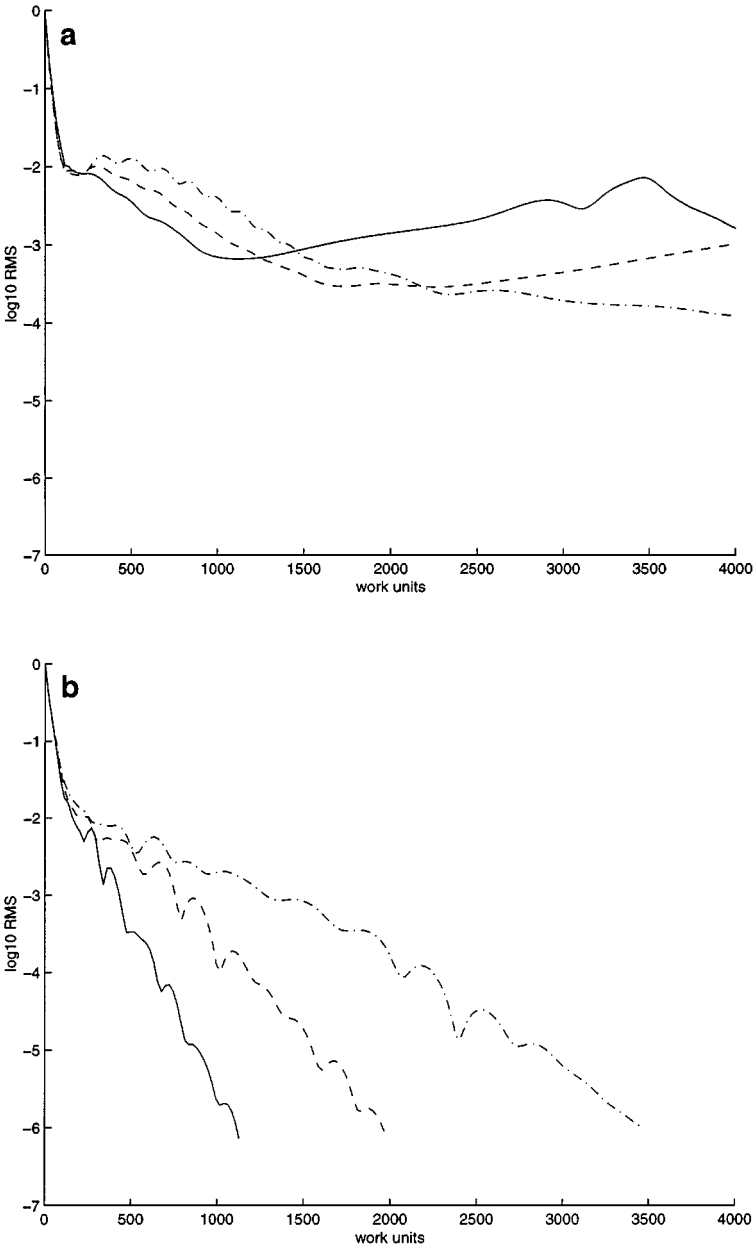

FIG. 9. Variation of convergence with grid size. NACA 0012 flow with full coarsening at $M_\infty = 0.1$ for set 2 grids. Solid, 96×32; dashed, 192×64; dash-dotted, 384×128. (a) Jacobi and (b) preconditioned Jacobi.

Mach number preconditioning again alleviates this problem. Convergence histories for the semi-coarsening results are plotted in Fig. 10.

4.4. *CPU Timings*

To further demonstrate the dependence of convergence on grid size, we plot the total CPU time required when running the simulations on a single SGI R10000 processor for the $M_\infty = 0.1$ cases with preconditioned Jacobi in Figs. 11 and 12 for the bump and airfoil flows,

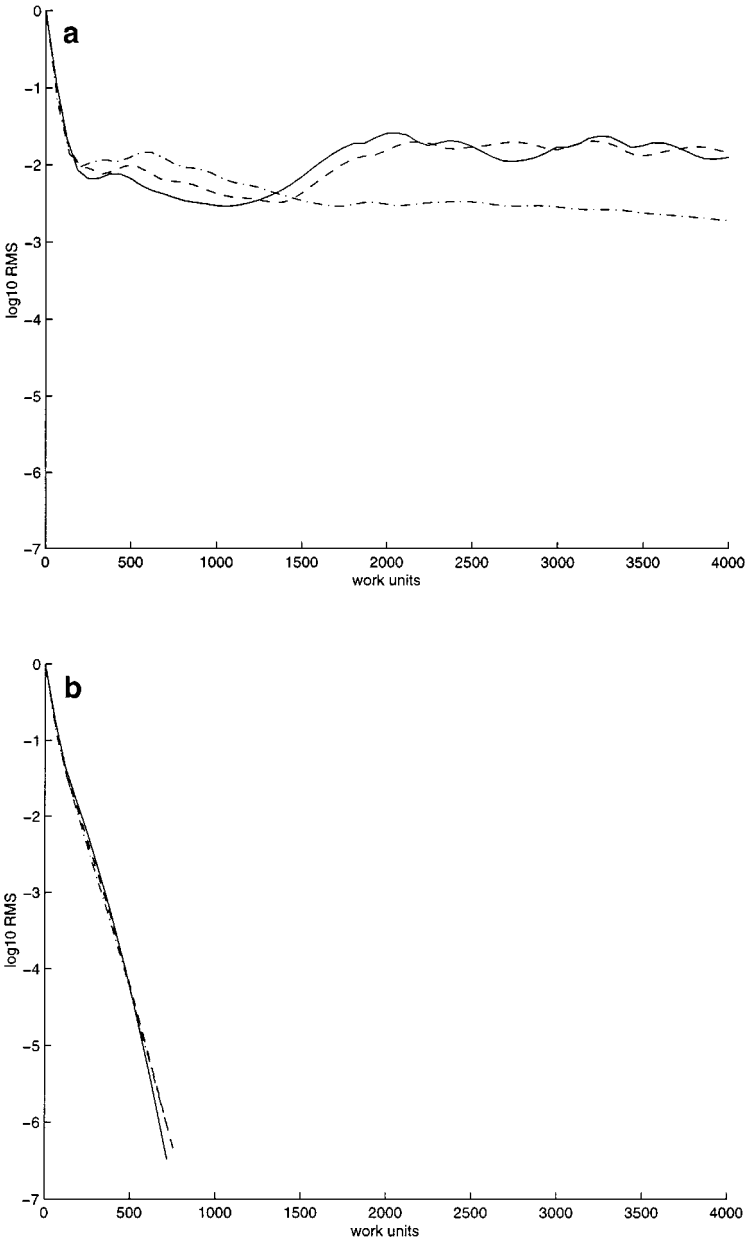

FIG. 10. Variation of convergence with grid size. NACA 0012 flow with semi-coarsening at $M_\infty = 0.1$ for set 2 grids. Solid, 96×32; dashed, 192×64; dash-dotted, 384×128. (a) Jacobi and (b) preconditioned Jacobi.

respectively. The full-coarsening CPU times (marked by $\times$) show a nonlinear increase with respect to grid size while the semi-coarsening times (marked by $\circ$) appear linear. Approximate curve fits for the timings are also shown in the figures. Specifically, for bump flows, the full-coarsening curve fit is

$$\mathrm{CPU}_{\mathrm{full}} \approx (4.4 \times 10^{-4}) N^{3/2} \text{ s},$$

and the semi-coarsening curve fit is

$$\mathrm{CPU}_{\mathrm{semi}} \approx (2.3 \times 10^{-2}) N \text{ s},$$

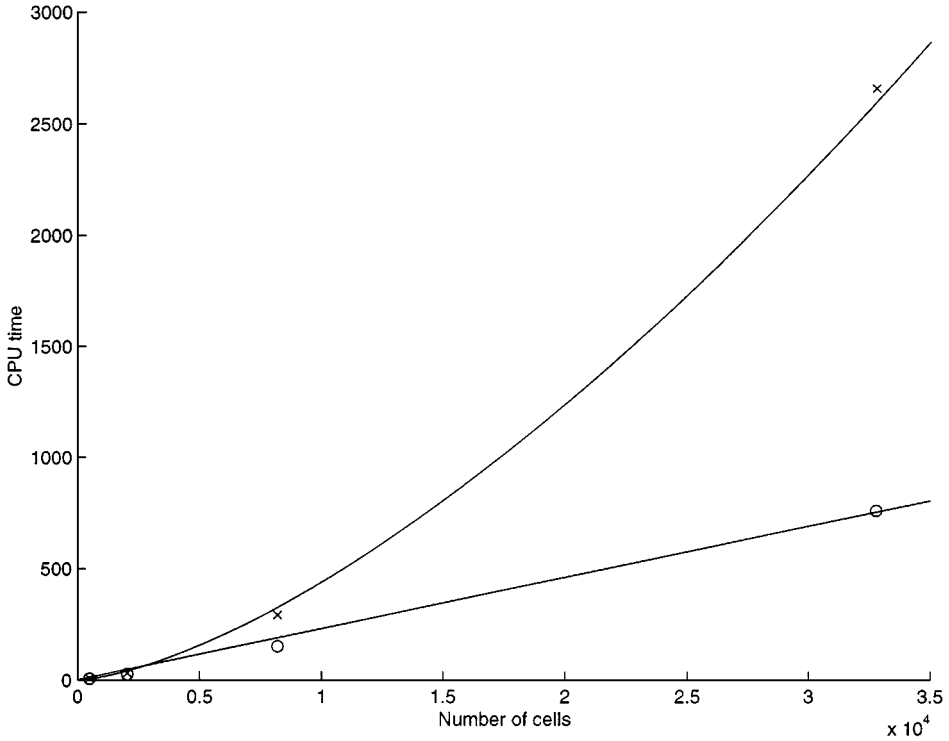

FIG. 11. CPU seconds versus number of cells, N. Bump flow with preconditioned Jacobi at $M_\infty = 0.1$. ($\times$) Full coarsening and ($\circ$) semi-coarsening. Curve fits are given by $\mathrm{CPU}_{\mathrm{full}} = (4.4 \times 10^{-4})N^{3/2}$ and $\mathrm{CPU}_{\mathrm{semi}} = (2.3 \times 10^{-2})N$.

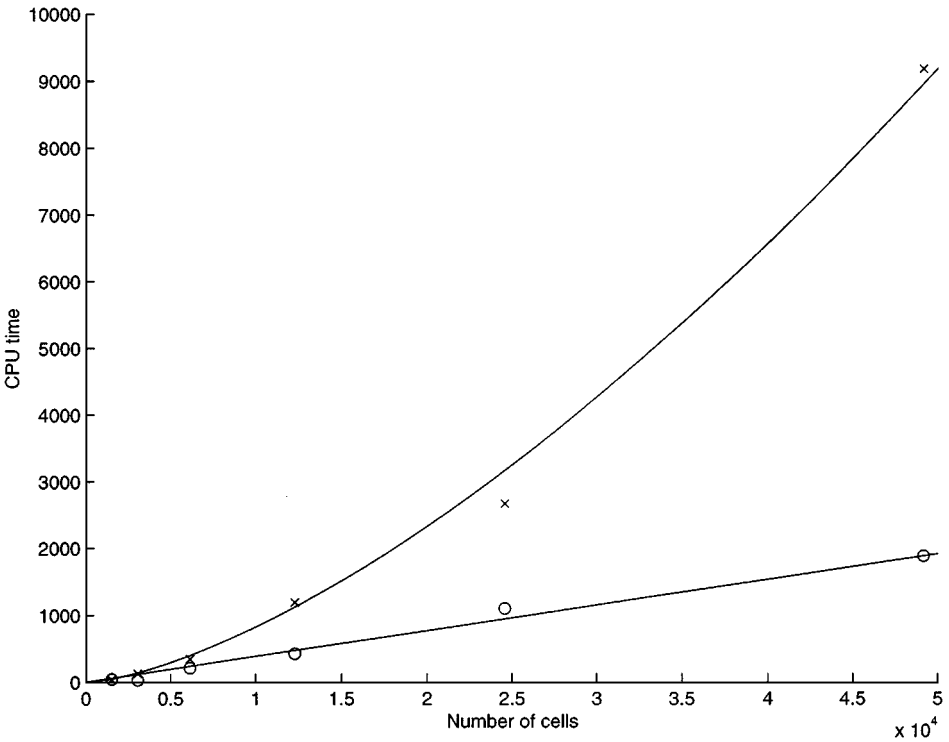

FIG. 12. CPU seconds versus number of cells, N. NACA 0012 flow with preconditioned Jacobi at $M_\infty = 0.1$. ($\times$) Full coarsening and ($\circ$) semi-coarsening. Curve fits are given by $\mathrm{CPU}_{\mathrm{full}} = (8.2 \times 10^{-4})N^{3/2}$ and $\mathrm{CPU}_{\mathrm{semi}} = (3.8 \times 10^{-2})N$.

where N is the total number of cells. For the airfoil results, the full-coarsening curve fit is

$$\mathrm{CPU}_{\mathrm{full}} \approx (8.2 \times 10^{-4}) N^{3/2} \text{ s},$$

and the semi-coarsening curve fit is

$$\mathrm{CPU}_{\mathrm{semi}} \approx (3.8 \times 10^{-2}) N \text{ s}.$$

Thus, to good approximation, full coarsening is an $O(N^{3/2})$ algorithm while semi-coarsening is $O(N)$. At coarser grid sizes, while semi-coarsening is usually faster than full coarsening, the CPU difference is minor; however, the real benefit of semi-coarsening is apparent for finer grids where the performance of full coarsening degrades.

4.5. *Effect of ϵ Limiter*

As shown from the cases above, the new ϵ limiter is a robust method which converged well in all tests. To assess the relative merits of the old (i.e., freestream-based) limiting and new limiting, we ran a set of cases on the NACA 0012 192×32 grid. For the old limit, several

TABLE IX
NACA 0012 Flow Convergence Results for Old and New ϵ-Limits

	Full coarsening		Semi-coarsening	
ϵ_{lim}	Cycles	Work	Cycles	Work
		$M_\infty = 0.1$		
Old: $\eta = 0$	51	1152	14	838
Old: $\eta = 1$	52	1174	14	838
Old: $\eta = 2$	59	1332	13	778
Old: $\eta = 3$	69	1558	15	897
Old: $\eta = 4$	77	1738	17	1017
New	51	1152	14	838
		$M_\infty = 0.3$		
Old: $\eta = 0$	52	1174	13	778
Old: $\eta = 1$	53	1197	13	778
Old: $\eta = 2$	65	1468	14	838
Old: $\eta = 3$	76	1716	16	957
Old: $\eta = 4$	85	1919	20	1196
New	52	1174	13	778
		$M_\infty = 0.5$		
Old: $\eta = 0$	67	1513	15	897
Old: $\eta = 1$	74	1671	15	897
Old: $\eta = 2$	74	1671	15	897
Old: $\eta = 3$	74	1671	15	897
Old: $\eta = 4$	74	1671	15	897
New	67	1513	15	897
		$M_\infty = 0.8$		
Old: $\eta = 0$	61	1377	26	1555
Old: $\eta = 1$	61	1377	25	1495
Old: $\eta = 2$	61	1377	25	1495
Old: $\eta = 3$	61	1377	25	1495
Old: $\eta = 4$	61	1377	25	1495
New	61	1377	26	1555

Note. Grid 192×32 cells. Six orders of magnitude drop in residual.

different values of η were used. The results are given in Table IX. The old and new ϵ limits perform similarly except for large values of η for which a noticeable drop in convergence rate is typically observed. For the higher Mach number flows, the convergence rate for the old limit is particularly insensitive to the value of η. This would seem to indicate that the potentially beneficial effect of low Mach number preconditioning in stagnation regions is not important for these flows. For higher Mach number flows with significant regions of low-speed flow such as occurs with separation, the effect of low-speed preconditioning and ϵ limiting may be more pronounced. Notably, the new ϵ limit performs well without any tuning.

To demonstrate the local limiting effect of the new ϵ limit, we investigate the old and new limiter activity for $M_\infty = 0.1$ on a 192×32 grid. For the old limit, $\eta = 3.0$. Figure 13 shows a contour plot of $\epsilon_{\text{cell}} - \epsilon(M_{\text{cell}})$ for the converged solution, where ϵ_{cell} is defined in

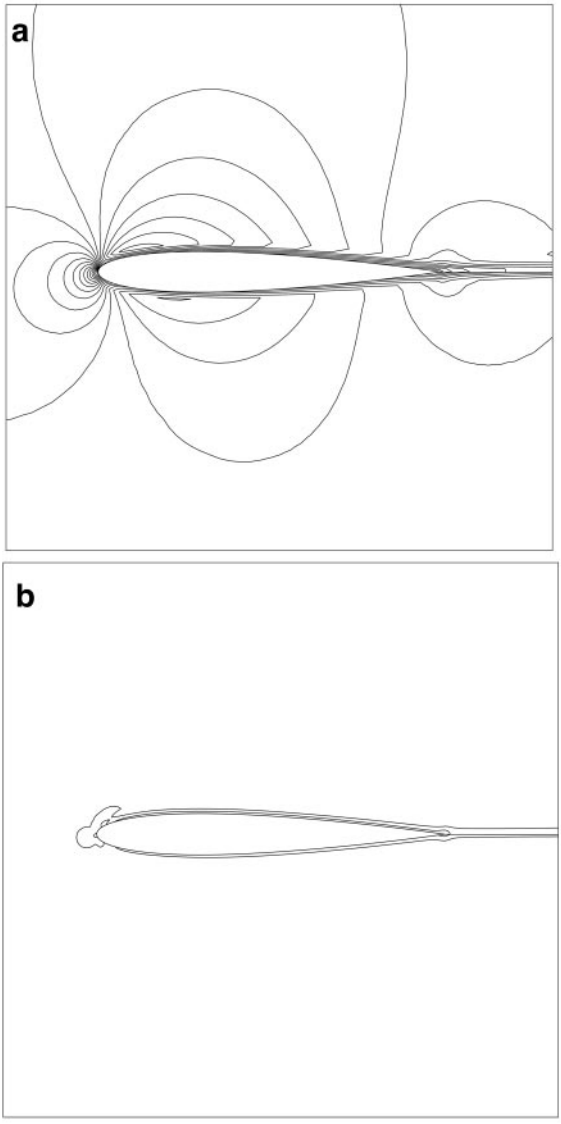

FIG. 13. Plots of $\epsilon_{\text{cell}} - \epsilon(M_{\text{cell}})$ with 41 equally spaced contours from 0 to 0.03 for NACA 0012, $M_\infty = 0.1$, and $\alpha_\infty = 1.25^\circ$ on a 192×32 grid. (a) Old ϵ limit ($\eta = 3$) and (b) new ϵ limit.

Eq. (10) and $\epsilon(M_{\text{cell}})$ is the value of ϵ evaluated with the local cell Mach number. Thus, in regions where no limiting occurs and Mach number variations are small, this quantity will be essentially zero. As shown in the plot, the new ϵ limit is active only near the body while the old ϵ limit is active throughout the flow. Surprisingly, the old ϵ limit converges only slightly slower than the new ϵ limit even though the old limit is active through an appreciable part of the flow.

Another interesting observation from the airfoil results is that $\eta = 0$ not only converges but often gives the best convergence rate for a test case. However, this is contrary to what several researchers have found [32, 18, 20, 33], particularly with airfoil problems where a stagnation point is present. This suggests that the use of the block Jacobi scheme, the maximization of ϵ over several cells as described in Section 3, or both may be contributing to the robustness of the local preconditioning observed here. In the solutions shown so far, the ϵ limits on the cell face are influenced by a total of eight cells composed of the nearest neighbors of the two cells at the face. The algorithm will pick the largest value of ϵ from this eight-cell stencil. Thus, even with $\eta = 0$, the influence from the other cells in the stencil can still keep ϵ from going to zero.

In order to gain better insight into the effect of the eight-cell stencil, the algorithm for determining ϵ_{flux} was modified to use only the Mach number from the two reconstructed states at the face. No other ϵ-limiting process was used (i.e., no freestream or Δp-based cutoff). Table X shows the results for the NACA 0012 airfoil with preconditioned Jacobi and semi-coarsening for the set 2 grids. Only one case becomes unstable (the finest grid at $M_\infty = 0.1$), while the other cases converge almost identically to the local limiter results in

TABLE X
Convergence Results for NACA 0012 Flow Using Semi-coarsening Algorithm with No ϵ Limiting for Set 2 Grids

	Semi-coarsening	
Grid	Cycles	Work
	$M_\infty = 0.1$	
96×32	12	718
192×64	12	767
384×128	UNS	UNS
	$M_\infty = 0.3$	
96×32	12	718
192×64	11	703
384×128	11	726
	$M_\infty = 0.5$	
96×32	15	897
192×64	17	1086
384×128	17	1121
	$M_\infty = 0.8$	
96×32	15	897
192×64	16	1022
384×128	20	1319

Note. Six orders of magnitude drop in residual. UNS, unstable.

Table VIII. The conclusion we draw is that the Jacobi formulation increases the robustness of the overall algorithm and, in combination with the limiting strategy described in Section 3, provides a robust and efficient algorithm for Euler calculations.

5. CLOSING REMARKS

We have developed a semi-coarsening multigrid algorithm using a point block Jacobi smoother with local preconditioning implemented for improved low Mach number performance. The locally preconditioned semi-coarsening algorithm converges at a rate which is nearly independent of grid size and Mach number for internal and external flows. CPU timings show that the computational work to converge six orders of magnitude is $O(N)$ for semi-coarsening, where N is the number of cells. In contrast, the full-coarsening algorithm computational work is $O(N^{3/2})$.

Furthermore, a preconditioning limiting strategy based on pressure changes which acts locally to limit the preconditioning in a stagnation point has been explored. This new limiting strategy in conjunction with the point block Jacobi smoother significantly increases the robustness for flows with stagnation points. In fact, based on the tests performed in this study, the point block Jacobi smoother may be the most important contributor to the added robustness.

While the semi-coarsened multigrid algorithm with local preconditioning has performed quite robustly for the cases presented, the algorithm must still be tested in more demanding circumstances. For example, the grids for all cases in this study are relatively well behaved with aspect ratios near unity and little stretching over a significant portion of the computational domain. Similarly, viscous and, more importantly, turbulent cases have not been investigated. These more severe applications remain for future work.

APPENDIX

In this Appendix, we describe the modified flux for the form of Turkel's preconditioner utilized in this research. The flux function in Eq. (2) can be expanded into its characteristic form,

$$\hat{H} = \frac{1}{2}[H(U_{\mathrm{L}}) + H(U_{\mathrm{R}})] - \frac{1}{2}\sum_{i=1}^{4} |\lambda_i|^* \Delta w_i \hat{\mathbf{P}}^{-1}\vec{r}_i,$$

where λ_i are the eigenvalues and $\vec{r}_i$ are the right eigenvectors of the matrix **PA**. The wave strengths, Δw_i, are the projection of the conserved state vector changes, ΔU, onto the corresponding left eigenvector, $\vec{l}_i^{\mathrm{T}}$, of **PA**; i.e., $\Delta w_i = \vec{l}_i^{\mathrm{T}} \Delta U$. Leaving ϵ a free parameter, the preconditioned eigenvalues and eigenvectors are

$$\begin{aligned}
\lambda_1 &= \tfrac{1}{2}[(1+\epsilon)u_{\mathrm{g}} - \tau],\\
\lambda_2 &= u_{\mathrm{g}},\\
\lambda_3 &= u_{\mathrm{g}},\\
\lambda_4 &= \tfrac{1}{2}[(1+\epsilon)u_{\mathrm{g}} + \tau],
\end{aligned}$$

where u_{g} is the velocity component normal to the cell face, and

$$\tau = \sqrt{(1-\epsilon)^2 u_{\mathrm{g}}^2 + 4\epsilon c^2},$$

with c the speed of sound. For the results contained in this paper, we have applied an entropy fix to these eigenvalues; specifically, we define

$$|\lambda_i|^* = \begin{cases} \Delta\lambda_i, & \text{if } |\lambda_i| < \Delta\lambda_i, \\ |\lambda_i|, & \text{if } |\lambda_i| \geq \Delta\lambda_i, \end{cases}$$

where $\Delta\lambda_i = 2|\lambda_{i\mathrm{L}} - \lambda_{i\mathrm{R}}|$.

The corresponding preconditioned right eigenvectors are

$$\mathbf{P}^{-1}\vec{r}_1 = \frac{1}{\epsilon\tau}\begin{pmatrix} s^+ \\ us^+ - 2\epsilon c^2 n_x \\ vs^+ - 2\epsilon c^2 n_y \\ Hs^+ - 2\epsilon c^2 u_{\mathrm{g}} \end{pmatrix},$$

$$\mathbf{P}^{-1}\vec{r}_2 = \begin{pmatrix} 0 \\ -n_y \\ n_x \\ v_{\mathrm{g}} \end{pmatrix},$$

$$\mathbf{P}^{-1}\vec{r}_3 = \begin{pmatrix} 1 \\ u \\ v \\ \frac{1}{2}(u^2 + v^2) \end{pmatrix},$$

$$\mathbf{P}^{-1}\vec{r}_4 = \frac{1}{\epsilon\tau}\begin{pmatrix} s^- \\ us^- + 2\epsilon c^2 n_x \\ vs^- + 2\epsilon c^2 n_y \\ Hs^- + 2\epsilon c^2 u_{\mathrm{g}} \end{pmatrix},$$

where

$$s^+ = \tau + (1-\epsilon)u_{\mathrm{g}},$$
$$s^- = \tau - (1-\epsilon)u_{\mathrm{g}}.$$

Note, v_{g} is the velocity tangential to the grid, H is the stagnation enthalpy, and (n_x, n_y) are the (x, y) components of the unit face normal. The left eigenvectors are

$$\vec{l}_1 = \frac{1}{4c^2}\begin{pmatrix} s^- u_{\mathrm{g}} + (\gamma-1)(u^2+v^2) \\ -s^- n_x - (\gamma-1)u \\ -s^- n_y - (\gamma-1)v \\ 2(\gamma-1) \end{pmatrix},$$

$$\vec{l}_2 = \begin{pmatrix} -v_{\mathrm{g}} \\ -n_y \\ n_x \\ 0 \end{pmatrix},$$

$$\vec{l}_3 = \frac{1}{c^2}\begin{pmatrix} c^2 - \frac{1}{2}(\gamma-1)(u^2+v^2) \\ (\gamma-1)u \\ (\gamma-1)v \\ -(\gamma-1) \end{pmatrix},$$

$$\vec{l}_4 = \frac{1}{4c^2}\begin{pmatrix} -s^+ u_{\mathrm{g}} + (\gamma-1)(u^2+v^2) \\ s^+ n_x - (\gamma-1)u \\ s^+ n_y - (\gamma-1)v \\ 2(\gamma-1) \end{pmatrix}.$$

Finally, the wave strengths are given by

$$\begin{aligned} \Delta w_1 &= \frac{1}{2c^2}\left(\Delta p - \frac{1}{2}\rho s^- \Delta u_{\mathrm{g}}\right), \\ \Delta w_2 &= \rho \Delta v_{\mathrm{g}}, \\ \Delta w_3 &= \Delta\rho - \Delta p/c^2, \\ \Delta w_4 &= \frac{1}{2c^2}\left(\Delta p + \frac{1}{2}\rho s^+ \Delta u_{\mathrm{g}}\right). \end{aligned}$$

ACKNOWLEDGMENTS

A portion of this work was completed while the first author was a Summer Visitor at ICASE in August 1997. This work has been partially supported by the NSF through NSF CAREER Award (ACS-9702435), the Boeing Company, and NASA Langley.

REFERENCES

1. A. Brandt, A guide to multigrid development, in *Multigrid Methods*, edited by W. Hackbush and U. Trottenberg (Springer-Verlag, Berlin/New York, 1981).
2. S. R. Allmaras, Analysis of a local matrix preconditioner for the 2-D Navier–Stokes equations, AIAA Paper 93-3330 (1993).
3. D. C. Jespersen, Design and implementation of a multigrid code for the Euler equations, *Appl. Math. Comput.* **13**, 357 (1983).
4. W. A. Mulder, Multigrid relaxation for the Euler equations, *J. Comput. Phys.* **60**, 235 (1985).
5. W. K. Anderson, J. L. Thomas, and D. L. Whitfield, Multigrid acceleration of the flux split Euler equations, AIAA Paper 86-0274 (1986).
6. P. W. Hemker and S. P. Spekreijse, Multiple grid and Osher's scheme for the efficient solution of the steady Euler equations, *Appl. Numer. Math.* **2**, 475 (1986).
7. B. Koren, Defect correction and multigrid for an efficient and accurate computation of airfoil flows, *J. Comput. Phys.* **77**, 183 (1988).
8. B. Koren, Multigrid and defect correction for the steady Navier–Stokes equations, *J. Comput. Phys.* **87**, 25 (1990).
9. W. K. Anderson, R. D. Rausch, and D. L. Bonhaus, Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids, *J. Comput. Phys.* **128**, 391 (1996).
10. A. Jameson, Solution of the Euler equations for two-dimensional transonic flow by a multigrid method, *Appl. Math. Comput.* **13**, 327 (1983).
11. B. van Leer, J. Thomas, P. Roe, and R. Newsome, A comparison of numerical flux formulas for the Euler and Navier–Stokes equations, AIAA Paper 87-1104-CP (1987).

12. S. R. Allmaras, Contamination of laminar boundary layers by artificial dissipation in Navier–Stokes solutions, in *Proceedings, Conf. Numerical Methods in Fluid Dynamics, 1992*.
13. R. C. Swanson and E. Turkel, Aspects of a high-resolution scheme for the Navier–Stokes equations, AIAA Paper 93-3372-CP (1993).
14. W. A. Mulder, A new approach to convection problems, *J. Comput. Phys.* **83**, 303 (1989).
15. W. A. Mulder, A high resolution Euler solver based on multigrid, semi-coarsening, and defect correction, *J. Comput. Phys.* **100**, 91 (1992).
16. E. Turkel, Preconditioned methods for solving the incompressible and low speed compressible equations, *J. Comput. Phys.* **72**, 277 (1987).
17. B. van Leer, W. T. Lee, and P. L. Roe, Characteristic time-stepping or local preconditioning of the Euler equations, AIAA Paper 91-1552 (1991).
18. D. Mavriplis, Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes, AIAA Paper 97-1952 (1997).
19. E. Turkel, Preconditioning-squared methods for multidimensional aerodynamics, AIAA Paper 97-2025 (1997).
20. B. van Leer, L. Mesaros, C. H. Tai, and E. Turkel, Local preconditioning in a stagnation point, AIAA Paper 95-1654 (1995).
21. D. L. Darmofal and P. J. Schmid, The importance of eigenvectors for local preconditioners of the Euler equations, *J. Comput. Phys.* **127**, 346 (1996).
22. P. L. Roe, Approximate Riemann solvers, parametric vectors, and difference schemes, *J. Comput. Phys.* **43**, 357 (1981).
23. B. van Leer, Upwind-difference methods for aerodynamic problems governed by the Euler equations, *Lect. Appl. Math.* **22** (1985).
24. J. M. Weiss and W. A. Smith, Preconditioning applied to variable and constant density flows, *AIAA J.* **33**(11), 2050 (1995).
25. J. F. Lynn and B. van Leer, A semi-coarsening multigrid solver for the Euler and Navier–Stokes equations with local preconditioning, AIAA Paper 95-1667 (1995).
26. J. Lynn, *Multigrid Solution of the Euler Equations with Local Preconditioning*, Ph.D. thesis, University of Michigan (1995).
27. R. Radespiel and R. C. Swanson, Progress with multigrid schemes for hypersonic flow problems, *J. Comput. Phys.* **116**, 103 (1995).
28. K. Siu, *A Robust Locally Preconditioned Multigrid Algorithm for the 2-d Euler Equations*, Master's thesis, Texas A&M University (1998).
29. D. L. Darmofal and B. van Leer, Local preconditioning: Manipulating Mother Nature to fool Father Time, in *Computing the Future II: Advances and Prospects in Computational Aerodynamics*, edited by M. Hafez and D. A. Caughey (Wiley, New York, 1998).
30. E. Turkel, V. N. Vatsa, and R. Radespiel, Preconditioning methods for low-speed flows, AIAA Paper 96-2460 (1996).
31. D. Jespersen, T. Pulliam, and P. Buning, Recent enhancements to OVERFLOW, AIAA Paper 97-0644 (1997).
32. D. Lee, *Local Preconditioning of the Euler and Navier–Stokes Equations*, Ph.D. thesis, University of Michigan (1996).
33. C. L. Reed and D. A. Anderson, Application of low speed preconditioning to the compressible Navier–Stokes equations, AIAA Paper 97-0873 (1997).