Electronic Transactions on Numerical Analysis.
Volume 6, pp. 182-197, Decembber 1997.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

DIRECTIONAL COARSENING AND SMOOTHING FOR ANISOTROPIC NAVIER-STOKES PROBLEMS*

DIMITRI J. MAVRIPLIS†

Abstract. Unstructured multigrid techniques for relieving the stiffness associated with high-Reynolds number viscous flow simulations on extremely stretched grids are investigated. One approach consists of employing a semi-coarsening or directional-coarsening technique, based on the directions of strong coupling within the mesh, in order to construct more optimal coarse grid levels. An alternate approach is developed which employs directional implicit smoothing with regular fully coarsened multigrid levels. The directional implicit smoothing is obtained by constructing implicit lines in the unstructured mesh based on the directions of strong coupling. Both approaches yield large increases in convergence rates over the traditional explicit full-coarsening multigrid algorithm. However, maximum benefits are achieved by combining the two approaches in a coupled manner into a single algorithm. An order of magnitude increase in convergence rate over the traditional explicit full-coarsening algorithm is demonstrated, and convergence rates for high-Reynolds number viscous flows which are independent of the grid aspect ratio are obtained.

Key words. multigird, anisotropic, Navier-Stokes.

AMS subject classification. 65B99.

1. Introduction. Multigrid methods have proven to be very effective techniques for accelerating convergence to steady state of both elliptic and hyperbolic problems. For simple elliptic problems, such as a Poisson equation, convergence rates of 0.1 are achievable, meaning that for each multigrid cycle, the numerical error can be reduced by one order of magnitude.

For hyperbolic problems, such as the Euler equations in computational fluid dynamics, the best rate that theoretically can be achieved for a second order discretization is 0.75, according to the analysis discussed by Mulder [25]. Indeed, many structured as well as unstructured Euler solvers achieve convergence rates close to 0.75 [2, 16, 26, 27, 35]. However, for high-Reynolds number viscous flow solutions, multigrid Navier-Stokes solvers generally result in convergence rates which are an order of magnitude or more slower than those obtained for inviscid flows. The main reason for this breakdown in efficiency of the multigrid algorithm is the use of highly stretched anisotropic meshes which are required to efficiently resolve boundary layer and wake regions in viscous flows. Indeed, the higher the Reynolds number, the more grid stretching is required, and the worse the convergence rate becomes.

The classic multigrid remedy for this problem is to resort to semi-coarsening, or to employ smoothers which are implicit in the direction normal to the stretching [4]. The idea of semi-coarsening is to coarsen the mesh only in the direction normal to the grid stretching, rather than in all coordinate directions simultaneously. This idea was used by Mulder [24, 25] to overcome the stiffness associated with the grid alignment phenomenon for an upwind scheme on non-stretched structured meshes. Since different regions of the flow field may contain anisotropies in differing directions, a complete sequence of grids, each coarsened in a single coordinate direction is generally required. Radespiel and Swanson [28] employed semi-coarsening to alleviate the stiffness due to stretched meshes for viscous flow calculations. More recently, Allmaras [1] has shown how the use of preconditioners coupled with semi-coarsening can help alleviate grid stretching induced stiffness. Pierce and Giles [27] have demonstrated improved convergence rates for turbulent Navier-Stokes flows using

* Submitted May 16, 1997. Accepted for publication August 16, 1997. Communicated by I. Yanveh.
† Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681

diagonal preconditioning coupled with a J-coarsening technique on structured grids, where the grid is only coarsened in the J-coordinate direction, which is normal to the boundary layer.

Semi-coarsening techniques can be generalized to unstructured meshes as directional coarsening methods. Graph algorithms can be constructed to remove mesh vertices based on the local degree and direction of anisotropy in either the grid or the discretized equations. This is achieved by basing point removal decisions on the values of the discrete stencil coefficients. This is the basis for algebraic multigrid methods [33], which operate on sparse matrices directly, rather than on geometric meshes. These techniques are more general than those available for structured meshes, since they can deal with multiple regions of anisotropies in conflicting directions. They offer the possibility of constructing algorithms which attempt to generate the "optimal" coarse grid for the problem at hand. Morano et al. [23] have demonstrated how such techniques can produce almost identical convergence rates for a Poisson equation on an isotropic cartesian mesh, and a highly stretched unstructured mesh. More recently, Francescatto [7] has demonstrated convergence improvements for the Navier-Stokes equations using directional coarsening multigrid.

One of the drawbacks of semi- or directional-coarsening techniques is that they result in coarse grids of higher complexity. While a full-coarsening approach reduces grid complexity between successively coarser levels by a factor of 4 in 2D, and 8 in 3D, semi-coarsening techniques only achieve a grid complexity reduction of 2, in both 2D and 3D. This increases the cost of a multigrid V-cycle, and makes the use of W-cycles impractical. Perhaps more importantly for unstructured mesh calculations, the amount of memory required to store the coarse levels is dramatically increased, particularly in 3D. Raw [30] advocates the use of directional coarsening, but at a fixed coarsening rate of 10 to 1, in order to reduce overheads. This generally results in the removal of multiple neighboring points in the coarsening process, and thus requires a stronger smoother than a simple explicit scheme. An alternative to semi-coarsening is to use a line solver in the direction normal to the grid stretching coupled with a regular full coarsening multigrid algorithm, at least for structured grid problems [4].

In the following sections, we examine the benefits obtained through the use of directional coarsening and implicit line solvers, and combine the two approaches to construct an efficient Reynolds averaged Navier-Stokes solver for very highly stretched meshes.

2. Base Solver . The Reynolds averaged Navier-Stokes equations are discretized by a finite-volume technique on meshes of mixed triangular and quadrilateral elements. The governing equations are given in integral form as:

$$\frac{\partial}{\partial t}(\mathbf{u}V) + \int_{\partial\Omega} (\mathbf{f}.\mathbf{n})\, dS \;=\; \int_{\partial\Omega} (\mathbf{g}.\mathbf{n})\, dS \,. \tag{2.1}$$

The solution $\mathbf{u}$ is given by

$$\mathbf{u} \;=\; \begin{pmatrix} \rho \\ \rho u_j \\ E \end{pmatrix} , \tag{2.2}$$

and the ith components of the convective and viscous flux vectors are, respectively:

$$\mathbf{f_i} = \begin{pmatrix} \rho u_i \\ \rho u_i u_j + \delta_{ij} p \\ u_i(E+p) \end{pmatrix} \qquad \mathbf{g_i} = \begin{pmatrix} 0 \\ \tau_{ij} \\ u_k \tau_{ik} - q_i \end{pmatrix} , \tag{2.3}$$

where ρ is the density, $u_j,\ j = 1, 2$ represents the x and y velocity components, E is the total energy, and p is the pressure, which is related to the other variables by the perfect gas law:

$$p \;=\; (\gamma - 1)\,(E - \frac{1}{2}\rho(u^2 + v^2))\,, \tag{2.4}$$

where γ is a constant. τ_{ij} and q_i represent the stress tensor and the heat flux vector. In the thin layer approximation to the Navier-Stokes equations, these terms can be taken as:

$$\tau_{ij} \;=\; \mu \, \frac{\partial u_i}{\partial x_j} \,, \qquad\qquad q_i \;=\; -\kappa \, \frac{\partial \frac{p}{\rho}}{\partial x_i} \,, \tag{2.5}$$

where μ and κ denote the fluid viscosity and thermal conductivity, respectively. These values are determined as the sum of the corresponding fluid properties, which are given, and the turbulence quantities, which are obtained through the solution of an additional turbulence modeling equation [37]. In the finite volume method, the variables $\mathbf{u}$ are stored at the vertices of the mesh, V represents the volume of the control volume associated with each vertex, as depicted in Figure 1, and the flux integrals are taken over the boundary of the control volume, denoted as $\partial\Omega$, with $\mathbf{n}$ being the normal vector at the control volume boundary.

The simplest way to compute these flux integrals is to precompute the fluxes at all vertices given the $\mathbf{u}$ values, and then integrate an average of the two fluxes on either side of a control-volume face with the face normal vector. This leads to a central difference scheme which is unstable unless additional artificial dissipation is employed. An alternative strategy is to use an upwind scheme. The upwinded fluxes can be constructed at a control-volume interface using an approximate Riemann solver [32] which computes an upwinded convective flux vector, given solution values $\mathbf{u_{left}}$ and $\mathbf{u_{right}}$ on either side of the control-volume boundary. If these solution values at the control-volume interfaces are taken as the values at the control-volume centers, a first order accurate scheme is obtained. To ensure second order accuracy, the interface values are extrapolated from the corresponding vertex values according to a Tayor expansion, using a control-volume gradient of $\mathbf{u}$ computed through a Green-Gauss contour integral about the control volumes. For multigrid calculations, a second-order discretization is employed for the convective terms on the fine grid, while a first-order discretization is used on the coarse grid levels.

In the simplified thin-layer form given above, the viscous fluxes are analogous to the terms of a simple diffusion equation, and are discretized by a finite difference scheme.

A sample mesh is depicted in Figure 2. Isotropic triangular elements are employed in regions of inviscid flow, and stretched quadrilateral elements are used in the boundary layer and wake regions. All elements of the grid are handled by a single unifying edge-based data-structure in the flow solver [19]. Triangular elements could easily be employed in the boundary layer regions simply by splitting each quadrilateral element into two triangular elements.

As shown in Figure 1, the resulting control-volumes for quadrilateral elements produce stencils with strong coupling in the direction normal to the grid stretching and weak coupling in the direction of stretching. When triangular elements are employed in regions of high mesh stretching, the stencils are complicated by the presence of diagonal connections, and do not decouple as simply in the normal and stretching directions as for quadrilateral elements. Therefore, the use of quadrilateral elements in regions of high mesh stretching is central to the solution algorithms described in this paper.

The basic time-stepping scheme is a three-stage explicit multistage (Runge-Kutta) scheme with stage coefficients optimized for high frequency damping properties [43], and a CFL number of 1.8. Convergence is accelerated by a local block Jacobi preconditioner, which involves inverting a 4×4 matrix for each vertex at each stage [22, 26, 27, 31]. In this approach, the time step Δt_i in the original multi-stage scheme:

$$\mathbf{u}_i^{(k)} = \mathbf{u}_i^{(0)} + \, CFL \, \alpha_k \, \Delta t_i \times \mathbf{R}_i(\mathbf{u}^{(k-1)}) \tag{2.6}$$

is replaced by the inverse of the point-wise Jacobian $[\mathbf{D}]_i$:

$$\mathbf{u}_i^{(k)} = \mathbf{u}_i^{(0)} + \, CFL\, \alpha_k \, [\mathbf{D}]_i^{\,-1} \times \mathbf{R}_i(\mathbf{u}^{(k-1)}) \tag{2.7}$$

where CFL is the $Courant - Friedrichs - Lewy$ number, α_k is the coefficient of the k^{th} Runge-Kutta stage, and $\mathbf{R}_i$ represents the discrete residual at vertex i. This approach, which can either be interpreted as a pre-conditioner, or as a local matrix time-step [41], has been shown to produce superior convergence rates for upwind schemes. No other techniques such as enthalpy damping or residual smoothing are employed [9].

The single equation turbulence model of Spalart and Allmaras [37] is utilized to account for turbulence effects. This equation is discretized and solved in a manner completely analogous to the flow equations, with the exception that the convective terms are only discretized to first-order accuracy.

3. Directional-Coarsening . In the context of unstructured meshes, there exists various strategies for implementing multigrid techniques. Two approaches that have been explored extensively by the author are the method of overset meshes, and the method of control-volume agglomeration [12, 16, 20, 36]. In the overset-mesh approach, a sequence of fine and coarse unstructured meshes is constructed either by hand, or in some automated fashion. These meshes are then employed in the multigrid algorithm, and variables are transferred between the various meshes of the sequence using linear interpolation. In the agglomeration approach, coarse levels are constructed by fusing together neighboring fine grid control volumes to form a smaller number of larger and more complex control volumes on the coarse grid.

While directional coarsening strategies can be employed in both multigrid approaches, for practical reasons we have chosen to utilize only the overset-mesh multigrid approach for these preliminary investigations. In fact, the same coarsening algorithm may be used for both approaches. In the overset-mesh approach, the graph coarsening algorithm is employed to select a subset of points from the fine grid from which the coarse grid will be formed. Once the coarse grid points have been determined, they must be triangulated in order to form a consistent coarse grid.

The coarsening algorithm is based on a weighted graph. Each edge of the mesh is assigned a weight which represents the degree of coupling in the discretization. In the true algebraic multigrid sense, these weights should be formed from the stencil coefficients. However, since the Navier-Stokes equations represent a system of equations, multiple coefficients exist for each edge. For simplicity, the edge weights are taken as the inverse of the edge length. For each fine grid vertex, the average and the maximum weight of all incident edges are precomputed and stored. This ratio of maximum to average weight is an indication of the local anisotropy in the mesh at each vertex. The coarsening algorithm begins by choosing an initial vertex as a coarse grid point or seed point, and attempts to remove neighboring points by examining the corresponding edge weights. If the ratio of maximum to average weights at the seed point is greater than α, (usually taken as $\alpha = 4$), then only the neighboring vertex along the edge of maximum weight is removed. Otherwise, (i.e. in isotropic regions) all neighboring edges are removed. The next seed point is then taken from a priority list which contains points which are adjacent to points which have previously been deleted.

In the present implementation, the graph-based coarsening algorithm is only employed in the boundary-layer and wake regions. Once these regions have been coarsened, the remaining regions of the domain are regridded using a Delaunay advancing-front technique with user specified resolution. This approach is purely for convenience, since the original mesh is generated by a two-step procedure, which employs an advancing-layers technique in the regions of viscous flow, and an advancing-front Delaunay triangulation in regions of inviscid flow

[17, 18]. The full weighted-graph coarsening algorithm will be implemented in the context of agglomeration multigrid in future work.

The first test case illustrates the convergence rates achievable for isotropic problems with the present algorithm. The inviscid transonic flow over a NACA 0012 airfoil is computed at a Mach number of 0.73 and incidence of 2.31 degrees. The mesh contains 5849 vertices and consists uniquely of isotropic triangular elements. The convergence history is documented in Figure 3. A total of 5 multigrid levels were employed, and a residual reduction of 11 orders of magnitude over 100 multigrid W-cycles was obtained. The overall convergence rate for this case is 0.77, which is very close to the theoretical limit of 0.75.

The second test case illustrates the stiffness induced by anisotropy. The viscous turbulent flow over the same geometry at the same conditions with a Reynolds number of 5 million is computed on the mesh depicted in Figure 1. This mesh contains a total of 4880 points. The cells on the airfoil surface have a height of 2.e-06 chords, and the maximum cell aspect ratio in the mesh is 20,000. This type of mesh is required in order to capture the boundary layer gradients. The computed Mach contours at these conditions are displayed in Figure 4. The convergence rate is depicted in Figure 5, using 5 multigrid levels which were constructed using the unweighted or full-coarsening version of the coarsening algorithm, as described in [20]. The slowdown in convergence over the inviscid test case is dramatic. After an initial phase of rapid convergence, the residual reduction rate slows down to less than 0.99 per multigrid W-cycle. Figure 5 also depicts the convergence rate of the same algorithm when a sequence of directionally coarsened grids is employed in the multigrid algorithm. The improvement is substantial, yielding a residual reduction of 0.91 per multigrid V-cycle.

4. Directional Implicit Smoothers. Although directional coarsening strategies for multigrid can achieve large increases in convergence speed, as demonstrated in the previous case, the coarse grids are more complex than those obtained in the full coarsening strategy. Note for example in the previous case that a V-cycle was required, since the W-cycle is impractical in this case. As mentioned previously, the overhead required to store the coarse levels is also greatly increased in such cases.

An alternative approach is to use a directionally implicit smoother in conjunction with full coarsening multigrid. For structured grids, an example of a directionally implicit smoother is a line solver. Line solvers are attractive because they result in block-tridiagonal matrices which can be solved very efficiently. For unstructured grids, predetermined grid lines do not exist. However, line solvers can still be employed, provided lines are artificially constructed in the unstructured grid. Techniques for constructing lines in an unstructured grid have previously been described in the literature [8, 14]. In those efforts, lines which span the entire grid were constructed using unweighted graph techniques. In the present context, the role of the line solver is to relieve the stiffness induced by grid anisotropy. Therefore, lines are desirable only in regions of strong anisotropy, and in these regions they must propagate along the direction of strong coupling.

Given these requirements, an algorithm to build lines in an anisotropic mesh can be constructed using a weighted graph technique, in a manner analogous to the algorithm for directional coarsening described previously. The edge weights are defined as previously, and the ratio of maximum to average adjacent edge weight is pre-computed for every mesh vertex. The vertices are then sorted according to this ratio. The first vertex in this ordered list is then picked as the starting point for a line. The line is built by adding to the original vertex the neighboring vertex which is most strongly connected to the current vertex, provided this vertex does not already belong to a line, and provided the ratio of maximum to minimum edge weights for the current vertex is greater than α, (using $\alpha = 4$ in all cases). The line terminates when no additional vertex can be found. If the originating vertex is not a boundary point then the procedure must be repeated beginning at the original vertex, and proceeding with the second strongest connection to this point. When the entire line is completed, a new line is initiated by proceeding to the next available vertex in the ordered list. Ordering of the initial vertex list in this manner ensures that lines originate in regions of maximum anisotropy, and terminate in isotropic regions of the mesh. The algorithm results in a set of lines of variable length. In isotropic regions, lines containing only one point are obtained, and the point explicit scheme is recovered.

On vector machines, the block-tridiagonal line solves must be vectorized across the lines. Because the lines have varying lengths, all lines must be made of similar length by padding the matrices of the shorter lines with zeros on the off-diagonals and ones on the diagonal entries, in such a way that zero additional corrections are generated at these locations by the implicit solver. To minimize the amount of padding required, the lines are sorted into groups, such that within each group, all lines are close in size to one another. Vectorization then takes place over the lines within each group. Using groups of size 100, the additional overhead due to padding is of the order of 10%. An alternative approach would be to replace the tridiagonal inversion routine with a cyclic reduction algorithm which can be vectorized directly. This may however result in substantially shorter vector lengths.

In the current approach, the size of the vector groups also determines the amount of memory required for the line solves, since the tridiagonal matrices are constructed just prior to, and discarded just after the lines are solved, and all lines are uncoupled. For scalar machines, lines may be processed individually, and the memory requirements (i.e. additional working memory required by the implicit solver) are determined by the length of the longest line in the grid.

The implicit system generated by the set of lines can be viewed as a simplification of the general Jacobian obtained from a linearization of a backwards Euler time discretization, where the Jacobian is that obtained from a first-order discretization. For block-diagonal preconditioning, all off-diagonal block entries are deleted, while in the line-implicit method, the block entries corresponding to the edges which constitute the lines are preserved. The implicit line solver is applied as a preconditioner to the three-stage explicit scheme described previously. At each stage in the multi-stage scheme, the corrections previously obtained by multiplying the residual vector by the inverted block-diagonal matrix are replaced by corrections obtained by solving the implicit system of block-tridiagonal matrices generated from the set of lines. This implementation has the desirable feature that it reduces exactly to the block-diagonal preconditioned multi-stage scheme when the line length becomes one (i.e. 1 vertex and zero edges), as is the case in isotropic regions of the mesh.

As an example, the viscous flow case of the previous section has been recomputed using the line-implicit solver with full-coarsening multigrid. The set of lines generated in the mesh of Figure 1 are depicted in Figure 6. The lines extend through the boundary layer and wake regions, and have mostly a length of 1 (i.e. 1 vertex and zero edges) in the regions of inviscid flow where the mesh is isotropic. A total of 5 meshes was employed in the multigrid sequence. The convergence rate for this algorithm is depicted in Figure 7. The residuals are reduced by over 4 orders of magnitude in 100 cycles, which corresponds to a residual reduction rate of 0.92 per multigrid W-cycle. This rate is close to that obtained by the point-wise scheme using directional coarsening. However, the coarse grids are of lower complexity in this case and a W-cycle has been used.

5. Combining Directional Coarsening and Smoothing . There are obvious similarities between the directional coarsening algorithm and the technique used to construct lines for the directional implicit method. These two techniques can be combined, in a coupled manner, to produce a more robust and efficient overall algorithm. The simplest way to combine these techniques is to use the pre-conditioned line-implicit smoother with a sequence of directionally coarsened coarse multigrid levels. In order to more closely couple these two techniques, we use exactly the same criteria for coarsening and for line construction. This ensures that coarsening will proceed in the same direction and along the lines determined for the implicit solver.

An example of this combined algorithm is depicted by the convergence plot in Figure 8. In this case, the combined directional-implicit-coarsening algorithm has been used to solve the same viscous turbulent flow as described in the previous sections. The fine mesh for this case is similar to the one displayed in Figure 1, but contains 5828 points, and the mesh cells near the airfoil boundary have a height of 2.e-07 chord lengths, and the maximum aspect-ratio cell in the mesh is 200,000. This represents an order of magnitude more anisotropy than the previous mesh. Even on this extremely stretched grid, the residuals are reduced by over 4 orders of magnitude over 100 cycles, which results in a average convergence rate of 0.92 per multigrid V-cycle. This rate is comparable to that achieved by either algorithm separately on the previous case. However, on this more highly stretched grid, neither algorithm alone could deliver this type of performance.

On the other hand, this case is still plagued by the high coarse grid complexities of the semi-coarsening approach. However, these two techniques, directional coarsening and directional implicit smoothing, are two strategies for treating the same problem. In this respect they are more overlapping in nature than complementary, and one of these techniques may be relaxed somewhat. We therefore propose to perform directional-coarsening as described previously, along the direction of the implicit lines, but at a faster coarsening rate of 4:1. Therefore, rather than remove every second point along the implicit lines, we remove three points for every preserved coarse grid point along the implicit lines. In isotropic regions, the coarsening algorithm remains unchanged. This has the effect of generating a sequence of coarse grids which has roughly the same complexity as that obtained by the full-coarsening technique. To illustrate this approach, the same case has been recomputed using the line-implicit smoother and directional coarsening at a 4:1 rate. The convergence rate is compared with that obtained previously in Figure 8. The average residual reduction rate for this case is 0.88. The fact that this rate is even faster than that achieved in the previous example is attributed to the use of W-cycles in the current calculation, which is made possible due to the low complexity of the coarse grids.

The combined directional implicit-coarsening multigrid algorithm produces convergence rates which are insensitive to the degree of grid stretching. This is illustrated by computing the flow over an RAE 2822 airfoil on three different grids of the same streamwise resolution but with varying normal wall and wake resolution. The first grid contains a normal wall spacing of 10^{-5} chords, and a total of 12,568 points, while the second grid contains a normal wall spacing of 10^{-6} chords, and 16,167 points, and the third grid a normal wall spacing of 10^{-7} chords, and 19,784 points. The second grid contains what is generally regarded as suitable normal and streamwise resolution for accurate computation of this type of problem, while the first and third grids are most likely under-resolved and over-resolved in the direction normal to the boundary layer, respectively. The second grid is displayed in Figure 9, while the Mach contours of the solution computed on this grid are displayed in Figure 10. A matrix-based artificial dissipation discretization [38] is employed for these calculations rather than the upwind scheme described previously. This discretization delivers similar accuracy as the

Roe Riemann-solver based upwind scheme, but does not require the use of a limiter, which may adversely affect convergence for transonic flow cases.

The convergence rates obtained on these three grids are displayed in Figure 11, using the original explicit full-coarsening multigrid algorithm, and the new directional-implicit-coarsening multigrid algorithm. The convergence of the original algorithm degrades as the mesh stretching increases, whereas the convergence of the new algorithm is essentially constant for all three meshes. Considering that these three meshes represent a two order of magnitude variation in the amount of mesh stretching, the performance of the directional-implicit-coarsening multigrid algorithm can be qualified as mesh aspect-ratio independent.

6. Conclusions and Further Work . In the above discussion, the cost of the graph algorithms used to construct the lines and the coarse grid levels is not considered. In all cases, the cost of this pre-processing is negligible compared to the cost of the multigrid solution process. The preprocessing requires no more than several seconds of CPU time, while the multigrid solution requires of the order of one or more hours for the examples described in this paper. This is despite the fact that the worst case complexity of the pre-processing graph algorithms is $O(NlogN)$ which is higher than the optimal complexity of $O(N)$ for the multigrid algorithm.

The comparisons between various schemes have all been made on a per cycle basis. While this is useful for determining the relative effectiveness of each technique as a multigrid smoother, and the degree to which the overall algorithm approaches the hypothetical "optimal" algorithm, it does not convey the relative costs of these various schemes. One of the reasons direct cpu comparisons have not been made is that the current code is not sufficiently optimized to provide a fair comparison with the baseline algorithm. Another reason, is that the technique used for coarse grid construction results in less than optimal coarse grid complexity since the inviscid regions of the flow are actually regridded, rather than coarsened by point removal. (However, the boundary layer and wake regions exhibit optimal coarsening).

The ultimate goal of this work is to incorporate these techniques into the more practical agglomeration or algebraic multigrid method described in [20, 21]. The development of an optimal grid coarsening scheme for agglomeration multigrid is currently under development. Figure 12 illustrates the first agglomerated level of a stretched unstructured grid, where the agglomeration has been constrained to proceed along the implicit-lines in the boundary layer regions, at a rate of 4:1. In the isotropic regions of the mesh, this algorithm reverts to that developed in [20, 21].

Preliminary results using the agglomeration multigrid strategy provide a comparison of the original isotropic explicit multigrid scheme with the directional implicit multigrid scheme based on cpu time, as shown in Figure 13. The explicit multigrid scheme employs a five stage Runge-Kutta smoother, and a residual smoothing technique for convergence acceleration [9], while the directional implicit multigrid scheme employs a three-stage Runge-Kutta smoother with line preconditioning. This represents the empirically attained optimal strategies for both types of multigrid algorithms. Further efficiencies are obtained in the latter case by inverting the line Jacobians at the first stage of the Runge-Kutta smoother, and freezing these inverted Jacobians for the remaining stages. This results in the cost of a directional implicit multigrid cycle being almost identical to that achieved by the explicit multigrid scheme. The comparison based on CPU time in Figure 13 is therefore very similar to the comparisons in the previous section based on multigrid cycles.

Although the proposed algorithms in this work have demonstrated convergence rates independent of the degree of grid stretching, the convergence rates obtained for viscous flows are still somewhat slower than what may be achieved for simple inviscid flow problems. Additional research is required to further reduce these rates consistently for all types of viscous flow problems. Perhaps the most promising avenue of research is to develop more sophisticated preconditioners in an effort to provide a better multigrid smoother at low additional cost [13, 40].

REFERENCES

[1] S. R. ALLMARAS, *Analysis of semi-implicit preconditioners for multigrid solution of the 2d compressible navier-stokes equations*, In *Proceedings of the 12th AIAA CFD Conference, San Diego CA*, June 1995, AIAA Paper 95-1651-CP.

[2] W. K. ANDERSON R. RAUSCH AND D. BONHAUS, *Implicit multigrid algorithms for incompressible turbulent flows on unstructured grids*, In Proceedings of the 12th AIAA CFD Conference, San Diego CA, June 1995, AIAA Paper 95-1740-CP.

[3] T. J. BARTH AND S. W. LINTON, *An unstructured mesh newton solver for compressible fluid flow and its parallel implementation*, AIAA paper 95-0221, January 1995.

[4] A. BRANDT, *Multigrid techniques with applications to fluid dynamics:1984 guide*, In VKI Lecture Series, pages 1–176, March 1984.

[5] Y. CHOI AND C. MERKLE, *The application of preconditioning to viscous flows*, J. Comp. Phys., 105(1993), pp. 207-223.

[6] D. L. DARMOFAL AND P. J. SCHMID, *The importance of eigenvectors for local preconditioners of the euler equations*, In Proceedings of the 12th AIAA CFD Conference, San Diego, CA, pages 102–117, June 1995. AIAA Paper 95-1655-CP.

[7] J. FRANCESCATTO, *Résolution de l'équation de Poisson sur des maillages étirés par une methode multigrille*, INRIA Report No. 2712, November 1995.

[8] O. HASSAN, K. MORGAN, AND J. PERAIRE, *An adaptive implicit/explicit finite element scheme for compressible high speed flows*, AIAA Paper 89-0363, January 1989.

[9] A. JAMESON, W. SCHMIDT, AND E. TURKEL, *Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes*, AIAA Paper 81-1259, 1981.

[10] D. JESPERSEN T. PULLIAM AND P. BUNING, *Recent enhancements to OVERFLOW*, AIAA Paper 97-0644, January 1997.

[11] G. KARYPIS AND V. KUMAR, *A fast and high quality multilevel scheme for partitioning irregular graphs*, Technical Report Technical Report 95-035, University of Minnesota, 1995. A short version appears in Intl. Conf. on Parallel Processing 1995.

[12] M. LALLEMAND, H. STEVE, AND A. DERVIEUX, *Unstructured multigridding by volume agglomeration: Current status*, Comput. & Fluids, 21(1992), pp. 397–433.

[13] J. F. LYNN AND B. VAN LEER, *A semi-coarsened multigrid solver for the euler and navier-stokes equations with local preconditioning*, In Proceedings of the 12th AIAA CFD Conference, San Diego, CA, pages 242–252, June 1995. AIAA Paper 95-1667-CP.

[14] D. MARTIN AND LÖHNER, *An implicit linelet-based solver for incompressible flows*, AIAA Paper 92-0668, January 1992.

[15] L. MARTINELLI AND A. JAMESON, *Validation of a multigrid method for the Reynolds-averaged Navier-Stokes Equations*, AIAA Paper 88-0414, January 1988.

[16] D. J. MAVRIPLIS, *Three-dimensional multigrid for the Euler equations*, AIAA Journal, 30(7):1753–1761, July 1992.

[17] ———, *An advancing-front Delaunay triangulation algorithm designed for robustness*, AIAA paper 93-0671, January 1993.

[18] ———, *Unstructured mesh generation and adaptivity*, In 26th VKI Lecture Series on Computational Fluid Dynamics, Lecture Series 1995-02, Von Karman Institute for Fluid Dynamics, Rhode Saint Genese, Belgium,arch 1995.

[19] D. J. MAVRIPLIS AND V. VENKATAKRISHNAN, *A unified multigrid solver for the Navier-Stokes equations on mixed element meshes*, AIAA Paper 95-1666, June 1995.

[20] ———, *Agglomeration multigrid for two dimensional viscous flows*, Comput. & Fluids, 24(1995), pp. 553–570.

[21] ———, *A 3D agglomeration multigrid solver for the Reynolds-averaged Navier-Stokes equations on unstructured meshes*, Internat. J. Numer. Methods Fluids, 23(1996), pp. 527–544.

[22] E. MORANO AND A. DERVIEUX, *Looking for O(N) Navier-Stokes solutions on non-structured meshes*, In 6th Copper Mountain Conf. on Multigrid Methods, NASA Conference Publication 3224, 1993, pp.

449–464.

[23] E. MORANO D. J. MAVRIPLIS AND V. VENKATAKRISHNAN, *Coarsening strategies for unstructured multi-grid techniques with application to anisotropic problems*, In 7th Copper Mountain Conf. on Multigrid Methods, NASA Conference Publication 3339, April 1995, pp. 591–606.

[24] W. A. MULDER, *A new multigrid approach to convection problems*, J. Comp. Phys., 83(1989), pp. 303–323.

[25] ———, *A high resolution Euler solver based on multigrid semi-coarsening and defect correction*, J. Comput. Phys., 100(1992), pp. 91–104.

[26] C. OLLIVIER-GOOCH, *Towards problem-independent multigrid cnvergence rates for unstructured mesh methods i: Inviscid and laminar flows*, In Proceedings of the 6th International Symposium on CFD, Lake Tahoe, NV, September 1995.

[27] N. PIERCE AND M. GILES, *Preconditioning on stretched meshes*, AIAA paper 96-0889, January 1996.

[28] R. RADESPIEL AND R. C. SWANSON, *Progress with multigrid schemes for hypersonic flow problems*, J. of Comput. Phys., 116(1995), pp. 103–122.

[29] ———, *An investigation of cell-centered and cell-vertex multigrid schemes for the Navier-Stokes equations*, AIAA paper 89-053, January 1989.

[30] M. RAW, *Robustness of coupled algebraic multigrid for the Navier-Stokes equations*, AIAA paper 96-0297, January 1996.

[31] K. RIEMSLAGH AND E. DICK, *A multigrid method for steady Euler equations on unstructured adaptive grids*, In 6th Copper Mountain Conf. on Multigrid Methods, NASA conference publication 3224, 1993, pp. 527–542.

[32] P. L. ROE, *Approximate Riemann solvers, parameter vectors and difference schemes*, J. Comp. Phys., 43(1981), pp. 357–372.

[33] J. W. RUGE AND K. STÜBEN, *Algebraic multigrid*, In Multigrid Methods, S. F. McCormick, ed., SIAM Frontiers in Applied Mathematics, Philadelphia, 1987, pp. 73–131.

[34] Y. SAAD AND M. H. SCHULTZ, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7(1986), pp. 856–869.

[35] D. SIDILKOVER, *A genuinely multidimensional upwind scheme and efficient multigrid solver for the compressible Euler equations*, ICASE Report 94-84, October 1994.

[36] W. A. SMITH, *Multigrid solution of transonic flow on unstructured grids*, In Recent Advances and Applications in Computational Fluid Dynamics, ed., O. Baysal, Proceedings of the ASME Winter Annual Meeting, November 1990.

[37] P. R. SPALART AND S. R. ALLMARAS, *A one-equation turbulence model for aerodynamic flows*, La Recherche Aérospatiale, 1(1994), pp. 5–21.

[38] R. C. SWANSON AND E. TURKEL, *On central difference and upwind schemes*, J. Comput. Phys., 101(1992), pp. 292–306.

[39] E. TURKEL, *Preconditioning methods for solving the incompressible and low speed compressible equations*, J. Comp. Phys., 72(1987), pp. 277–298.

[40] E. TURKEL V. N. VATSA AND R. RADESPIEL, *Preconditioning methods for low speed flow*, AIAA Paper 96-2460, 1996.

[41] B. VAN LEER W. T. LEE AND P. ROE, *Characteristic time-stepping or local preconditioning of the Euler equations*, In Proceedings of the 10th AIAA CFD Conference, Honolulu, Hawaii, pages 260–282, June 1991. AIAA Paper 91-1552-CP.

[42] B. VAN LEER E. TURKEL C. H. TAI AND L. MESAROS, *Local preconditioning in a stagnation point*, In Proceedings of the 12th AIAA CFD Conference, San Diego, CA, pages 88–101, June 1995. AIAA Paper 95-1654-CP.

[43] B. VAN LEER, C. H. TAI, AND K. G. POWELL, *Design of optimally-smoothing multi-stage schemes for the Euler equations*, AIAA Paper 89-1933, June 1989.

[44] V. VENKATAKRISHNAN, *On the accuracy of limiters and convergence to steady state solutions*, J. Comp. Phys., 118(1995), pp. 120–130.

[45] J. M. WEISS AND W. A SMITH, *Preconditioning applied to variable and constant density time-accurate flows on unstructured meshes*, AIAA Paper 94-2209, June 1994.

[46] L. B. WIGTON N. J. YU AND D. P. YOUNG, *GMRES acceleration of computational fluid dynamic codes*, In Proceedings of the 7th AIAA CFD Conference, pages 67–74, July 1985. AIAA Paper 85-1494-CP.

ETNA
Kent State University
etna@mcs.kent.edu

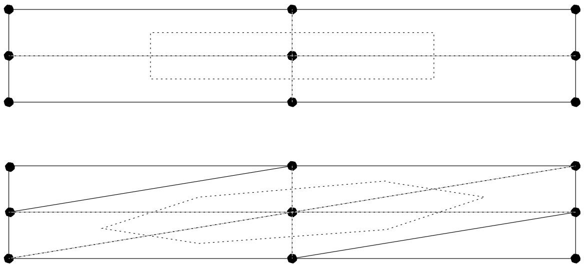

FIG. 2.1. *Median control-volumes for stretched quadrilateral and triangular elements*

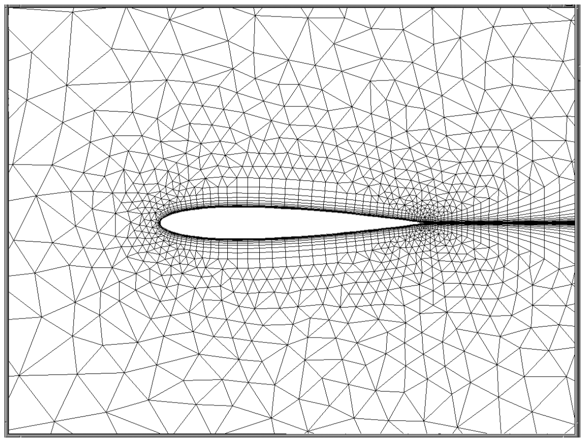

FIG. 2.2. *Mixed element grid used for viscous flow calculations about NACA 0012 airfoil; Number of vertices = 4880*

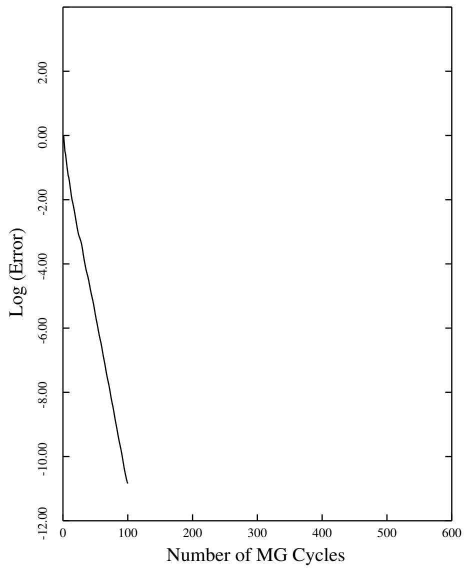

FIG. 3.1. *Multigrid Convergence Rate using Explicit Smoothing and Full-Coarsening for inviscid flow over NACA 0012 airfoil*

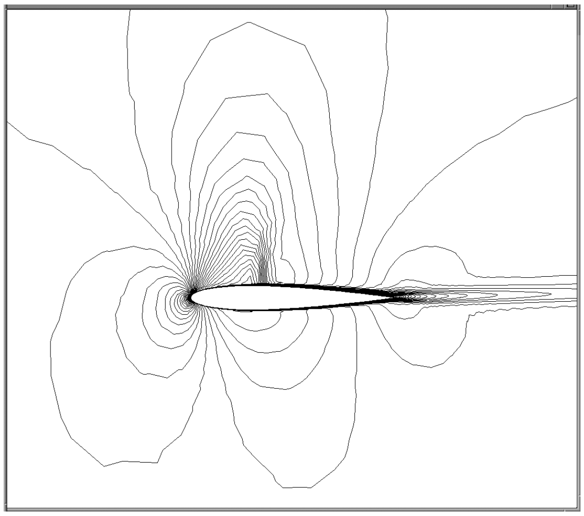

FIG. 3.2. *Computed Mach contours for viscous flow over NACA 0012 airfoil*

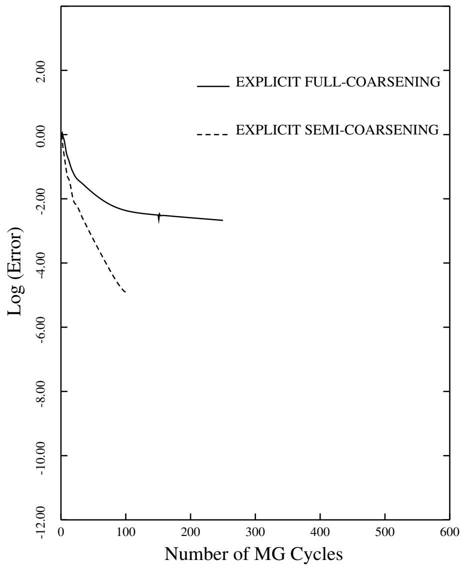

FIG. 3.3. *Comparison of Multigrid Convergence Rate using Explicit Smoothing and Full-Coarsening versus Explicit Smoothing and Semi-Coarsening for viscous flow over NACA 0012 airfoil*

ETNA
Kent State University
etna@mcs.kent.edu

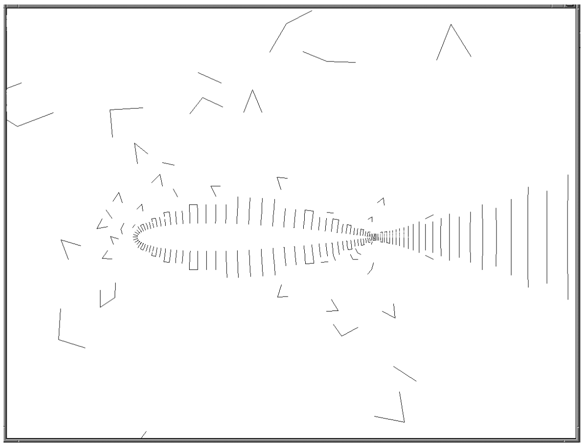

FIG. 4.1. *Implicit lines produced by the current algorithm on the grid of Figure 1*

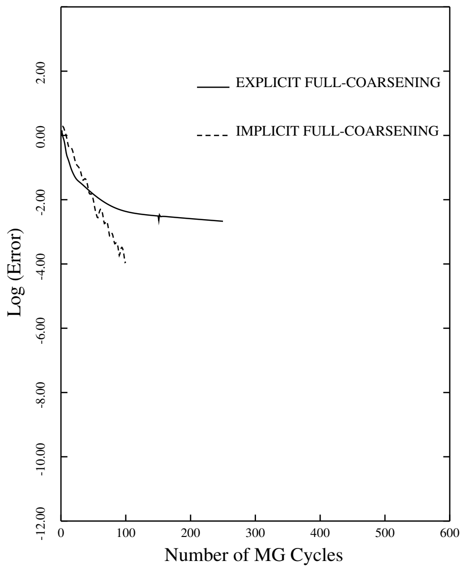

FIG. 4.2. *Comparison of Multigrid Convergence Rate using Explicit Smoothing and Full Coarsening versus Implicit Line Solver and Full-Coarsening for viscous flow over NACA 0012 airfoil*

ETNA
Kent State University
etna@mcs.kent.edu

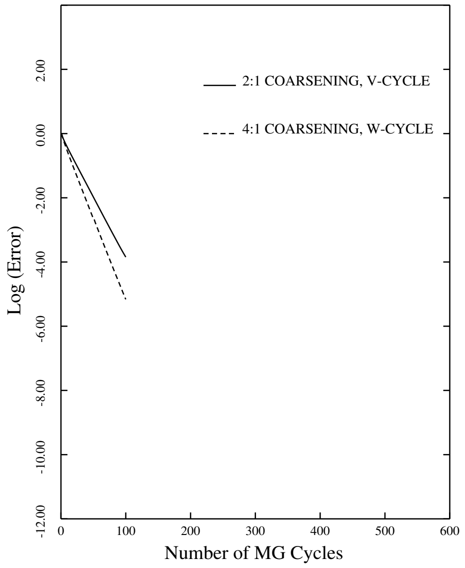

FIG. 5.1. *Multigrid Convergence Rate using Implicit Line-Solver and Semi-Coarsening for viscous flow over NACA 0012 airfoil*

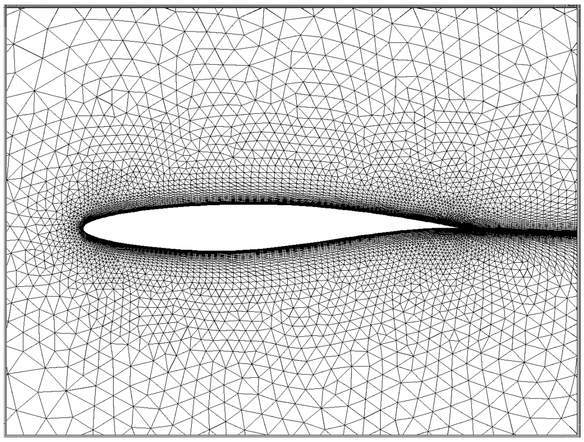

FIG. 5.2. *Unstructured Grid Used for Computation of Transonic Flow Over RAE 2822 Airfoil. Number of Points = 16167, Wall Resolution =* 10^{-6} *chords*

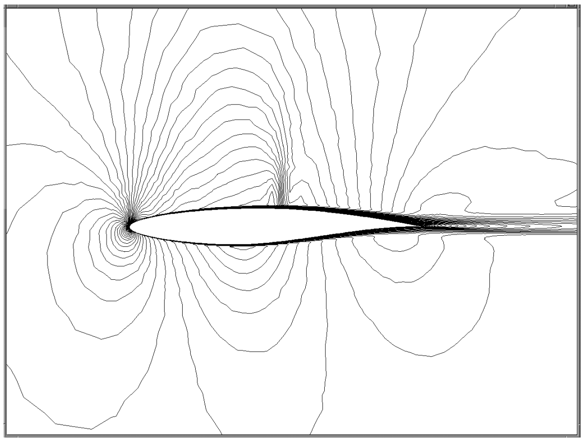

FIG. 5.3. *Computed Mach Contours on above Grid. Mach = 0.73, Incidence = 2.31 degrees, Re = 6.5 million*

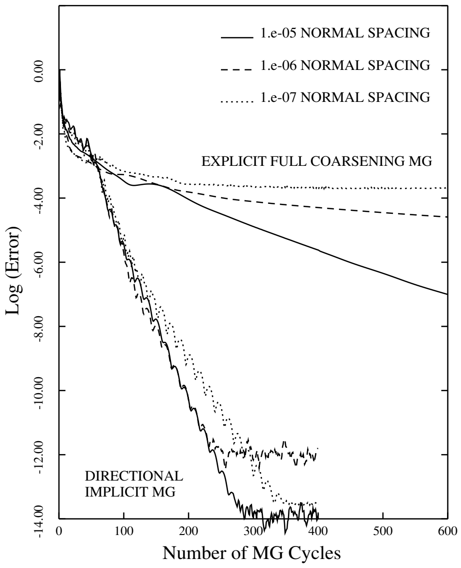

FIG. 5.4. *Comparison of original explicit full coarsening multigrid algorithm and new directional-implicit-coarsening multigrid algorithm on meshes of varying degrees of anisotropy*

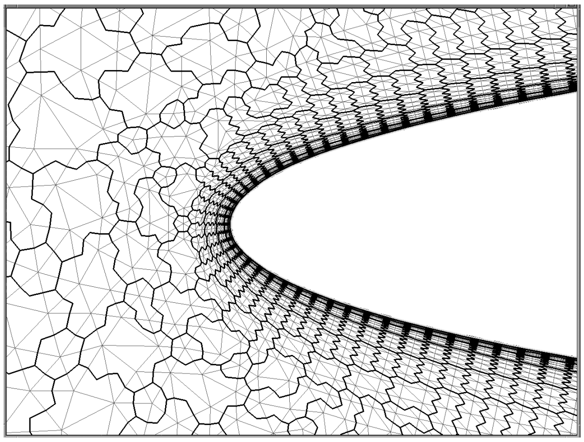

FIG. 6.1. *Example of Agglomerated Grid Using 4:1 Coarsening along Lines in Boundary-Layer Region*

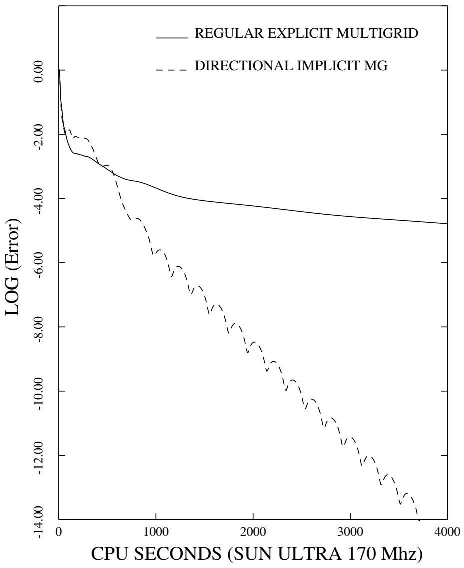

FIG. 6.2. *Comparison of Explicit Isotropic Agglomeration Multigrid with Directional Implicit Multigrid for Computation of Transonic Flow on Grid of Figure 9 in terms of CPU Time.*