AIAA JOURNAL
Vol. 33, No. 10, October 1995

Multigrid Acceleration of an Upwind Euler Solver on Unstructured Meshes

Carl F. Ollivier-Gooch*
National Research Council, Moffett Field, California 94035

Multigrid acceleration has been implemented for an upwind flow solver on unstructured meshes. The flow solver is a straightforward implementation of Barth and Jespersen's unstructured scheme, with least-squares linear reconstruction and a directional implementation of Venkatakrishnan's limiter. The multigrid scheme itself is designed to work on mesh systems that are not nested, allowing great flexibility in generating coarse meshes and in adapting fine meshes. A new scheme for automatically generating coarse unstructured meshes from fine ones is presented. A subset of the fine mesh vertices is selected for retention in the coarse mesh. The coarse mesh is generated incrementally from the fine mesh by removing one rejected vertex at a time. In this way, a valid coarse mesh triangulation is guaranteed. Factors affecting multigrid convergence rate for inviscid flow are thoroughly examined, including the effect of the number of coarse meshes used, the type of multigrid cycle employed, the spatial discretization used on coarse meshes, and the nature of the flow. The present multigrid scheme is very successful in reducing computational time for inviscid flows in the subsonic and transonic regime.

Nomenclature

A	= area of control volume in the dual mesh
C	= coarse mesh correction to solution
c	= speed of sound
CFL	= Courant–Friedrichs–Lewy number
E	= energy density
$\boldsymbol{F}$	= tensor of flux density
$I_{c\to f}$	= operator for interpolating from coarse to fine mesh
$I^{Q}_{f\to c}$	= operator for interpolation of solution from fine to coarse mesh
$I^{R}_{f\to c}$	= operator for conservative restriction of residuals from fine to coarse mesh
K	= constant in Venkatakrishnan flux limiter
$\boldsymbol{n}$	= outward-facing normal, scaled by area of face of the dual mesh
$\boldsymbol{r}$	= vector from one vertex to an adjacent one
P	= pressure
Q	= vector of conserved variables
$R(\cdot)$	= residual operator
S	= multigrid source term
s	= arclength along boundary of control volume
t	= time
u, v	= velocity components in the x and y directions
x, y	= space coordinates
α	= stage coefficients for multistage time-stepping scheme
ρ	= mass density
Ψ	= factor by which gradient is reduced by limiting
Ω	= control volume in dual mesh

Subscripts

c	= coarse mesh
f	= fine mesh
i	= vertex index; covers entire mesh
j	= vertex index; covers neighbors of a given vertex

Superscripts

k	= stage of multistage time-stepping scheme
n	= time level

Received Dec. 8, 1994; revision received April 3, 1995; accepted for publication April 3, 1995.

*Research Associate, Mail Stop T27B-2, NASA Ames Research Center. Associate Member AIAA.

Introduction

THE growing popularity of unstructured mesh methods is due to the relative ease with which these methods can be applied to solve flow problems around complex geometries and the ease with which local refinement of unstructured meshes can be carried out. A major impediment to their widespread use remains the relatively large time per mesh point required to obtain converged solutions on unstructured meshes compared with structured meshes.

The use of multigrid convergence acceleration to improve the convergence rate of flow problems on structured meshes is quite common. Jameson obtained the first multigrid flow solutions in the late 1970s for the transonic potential equations,[1] and since that time multigrid methods have been applied to a variety of explicit and implicit structured mesh flow solvers with great success.[2–5] Beginning in the late 1980s, a great deal of work has been done in the area of unstructured multigrid flow solvers.[6–10] These works have exclusively used central-difference schemes in conjunction with multigrid.

The present work applies multigrid methods to an upwind Euler flow solver on unstructured meshes. The flow solver is an upwind Euler finite volume solver of the type described by Barth and Jespersen[11] and Barth.[12] The multigrid scheme uses a sequence of pregenerated coarse meshes, which are not assumed to be nested, nor are the coarse mesh points assumed to be a subset of the fine mesh points. This allows maximum flexibility in generation of coarse meshes.

A new scheme for automatic generation of coarse meshes has been developed. The coarse mesh is generated by removing vertices from the fine mesh. This incremental process guarantees a valid triangulation at each step in the process of generating the coarse mesh. The vertices that remain are chosen so that no two coarse mesh vertices are connected by an edge in the fine mesh.

Various options in multigrid cycling and coarse mesh discretization are discussed and compared on the basis of efficiency.

Flow Solver

The equations to be solved are the Euler equations in two dimensions, which can be written in integral form for a computational control volume Ω as

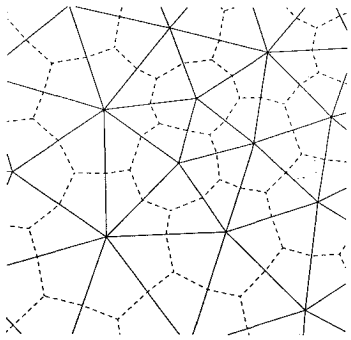

Fig. 1 Triangular mesh (solid lines) with median dual control volumes (dashed lines).

$$\frac{\partial}{\partial t}\left(\iint_{\Omega} Q\,dx\,dy\right)+\int_{\partial\Omega} \boldsymbol{F}\cdot\hat{\boldsymbol{n}}\,ds=0 \tag{1}$$

where

$$Q=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix} \tag{2}$$

$$\boldsymbol{F}=\begin{bmatrix}\rho u\\ \rho u^2+P\\ \rho uv\\ u(E+P)\end{bmatrix}\hat{\imath}+\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+P\\ v(E+P)\end{bmatrix}\hat{\jmath} \tag{3}$$

The perfect gas equation of state is used to relate the energy per unit volume E and the pressure P:

$$P=(\gamma-1)\left(E-\rho\frac{u^2+v^2}{2}\right) \tag{4}$$

The conserved variables Q are stored at the vertices of a triangular mesh, and Eq. (1) is solved using the upwind finite volume method of Barth and Jespersen[11] and Barth.[12] The computational domain is decomposed into small, nonoverlapping control volumes using the median dual of the triangular mesh. This dual, shown in Fig. 1, is formed by connecting the centroids and midsides of triangular cells.

The conserved variables are reconstructed locally in each control volume. In the present work, a least-squares linear reconstruction is used.[12,13] The reconstructed gradients must be limited to ensure monotonicity of the solution. Venkatakrishnan's limiter[14] is used because of its superior convergence properties. This limiter reduces the reconstructed gradient $\nabla\phi$ at the vertex V_i locally by a factor of

$$\Psi_i=\max_{\text{neighbors}\,j}\begin{cases}\dfrac{1}{\Delta_2}\dfrac{\left(\Delta_{1,\max}^2+\epsilon^2\right)\Delta_2+2\Delta_2^2\Delta_{1,\max}}{\Delta_{1,\max}^2+2\Delta_2^2+\Delta_{1,\max}\Delta_2+\epsilon^2} & \text{if } \Delta_2>0\\[2ex] \dfrac{1}{\Delta_2}\dfrac{\left(\Delta_{1,\min}^2+\epsilon^2\right)\Delta_2+2\Delta_2^2\Delta_{1,\min}}{\Delta_{1,\min}^2+2\Delta_2^2+\Delta_{1,\min}\Delta_2+\epsilon^2} & \text{if } \Delta_2<0\end{cases} \tag{5}$$

where

$$\Delta_2\equiv\tfrac{1}{2}\nabla\phi_i\cdot r_{ij}$$

$$\Delta_{1,\max}\equiv\left(\max_j\phi_j\right)-\phi_i$$

$$\Delta_{1,\min}\equiv\left(\min_j\phi_j\right)-\phi_i$$

and ϵ^2 is chosen to control the amount of overshoot and undershoot allowed. This value depends on an estimate of length scale, $\epsilon^2\equiv(K\Delta x)^3$. In the present work, Δx is a global length scale, defined in terms of the mean area of triangles in the computational domain,

$$\Delta x=\sqrt{\frac{4}{\sqrt{3}}\frac{\text{total area}}{\text{number of cells}}} \tag{6}$$

K is set to be 0.075.

To further reduce numerical dissipation, the limiter is invoked directionally. If a new extremum would appear in the reconstruction along edge $\overline{0j}$, then the gradient is reduced by a factor of Ψ in the direction r_{0j} but not reduced in the direction normal to r_{0j}. This approach has been shown by Aftosmis et al.[15] to significantly reduce the frequency and severity of limiting as compared with isotropic limiting.

After the solution is reconstructed, a flux quadrature is performed around each control volume using a single point along each face of the control volume. This quadrature is exact for a linear variation of flux along the face. Since the reconstructed data are not continuous at cell faces, the flux at the face must be computed using a Riemann solver. For the present work, Roe's approximate Riemann solver was used.[16]

Once the flux integral has been computed, the solution is advanced in time using a multistage Runge–Kutta scheme:

$$\begin{aligned}Q_i^{(0)}&=Q_i^n\\ Q_i^{(k)}&=Q_i^n-\alpha_k\frac{\Delta t_i}{A_i}R_i\left[Q^{(k-1)}\right]\\ Q_i^{n+1}&=Q_i^{(k)}\end{aligned} \tag{7}$$

At present, the coefficients α_k are $\{1/15,\ 7/45,\ 2/7,\ 1/2,\ 1\}$. These coefficients were chosen on the basis of model problem analysis, which showed that a five-stage scheme with these coefficients has good stability and damping properties for the wave equation.

The scheme uses local time stepping, with Δt defined as[12]

$$\Delta t_i=CFL\frac{A_i}{-\sum_{\text{neighbors}}\max\left[0,\ (\boldsymbol{u}\cdot\hat{\boldsymbol{n}}_{ij}+c)\right]\|\boldsymbol{n}_{ij}\|} \tag{8}$$

No residual smoothing[17] is done, and enthalpy damping[18] is not performed in anticipation of using the scheme for viscous flows, where total enthalpy is not constant.

Multigrid Acceleration

Multigrid methods are intended to accelerate convergence to steady state by using a series of coarse meshes to produce corrections to the solution on the finest mesh. This approach was developed in the 1960s and 1970s for elliptic problems and has since been applied with great success to mixed systems of equations such as the Euler and Navier–Stokes equations. Most time-marching schemes damp ill-resolved waves very strongly, and the use of multigrid allows this effect to damp features of progressively larger size. This process is analogous to the effects of multigrid for elliptic systems. Unlike elliptic systems, hyperbolic systems also have errors that must propagate out of the domain. The use of coarse meshes in a multigrid scheme transmits these errors to the boundaries of the domain with significantly less computational effort than would be required on the fine mesh. Even so, this effect prevents multigrid efficiency from being as high for mixed systems as for elliptic systems.

In this work, the full approximation scheme (FAS) of Brandt[19] has been used. In this approach, the entire nonlinear problem is transferred from the fine mesh to the coarser meshes. To ensure that the corrections computed in the multigrid process lead to the physically correct solution, the equations on the coarser meshes must be modified by the addition of a source term:

$$S_c=I_{f\to c}^R R_f(Q_f)-R_c\left(I_{f\to c}^Q Q_f\right) \tag{9}$$

This term is designed so that evolution on the coarse mesh is driven by the residual on the finest mesh. Time advance on the coarse

mesh proceeds as before, with the source term added to the residual at each stage:

$$Q_i^{(k)} = Q_i^n - \alpha_k \frac{\Delta t_i}{A_i}\left\{R_i\left[Q^{(k-1)}\right] + S_i\right\} \tag{10}$$

The fine-to-coarse transfer, or restriction, operators $I^R_{f\to c}$ and $I^Q_{f\to c}$ need not be the same. Residual restriction $I^R_{f\to c}$ should be performed conservatively; this ensures that the sum of the residuals on the coarse mesh matches that sum on the fine mesh at the first coarse mesh iteration and improves convergence rate. In the present work, solution restriction is performed by interpolation among nearby values of the solution on the fine mesh.

Also, the residual computations R_f and R_c need not be performed in the same way. The fine-mesh residual R_f completely determines the converged solution of the equations. This allows the freedom to choose R_c solely to improve convergence rate. The use of a first-order discretization on coarse meshes is an obvious choice for computing R_c, since first-order upwind spatial discretization has excellent damping properties. The utility of this approach is discussed in a later section.

After an approximate solution is obtained on the coarse mesh, the change in the solution is used to update the solution on the fine mesh. The change C_c in the coarse mesh solution is defined as the difference between $\hat{Q}_c$, the approximate solution on the coarse mesh, and $I^Q_{f\to c}Q_f$, the most recent approximation on the fine mesh, interpolated to the coarse mesh. This quantity is interpolated to provide a correction C_f that is then added to the fine mesh solution:

$$C_c \equiv \hat{Q}_c - I^Q_{f\to c}Q_f \tag{11}$$

$$C_f = I_{c\to f}C_c \tag{12}$$

$$Q_f \leftarrow Q_f + C_f \tag{13}$$

This interpolation is done linearly, with no regard for discontinuities in the flow. The corrections thus obtained are smooth even for nonsmooth flows after an initial transient phase, during which shocks reach what is nearly their final position.

The cycles used in the present work are sawtooth W-cycles. These are identical to traditional W-cycles except that relaxation steps are made only as the cycle progresses from coarse meshes to fine ones. This is opposite to the usual multigrid sawtooth cycles (e.g., Ref. 17), in which relaxation is only performed when progressing from fine to coarse meshes.

Mesh-to-Mesh Transfers

In Fig. 2, vertices P, Q, and R define a cell on the coarse mesh, whereas vertices 1, 2, and 3 define a cell on the fine mesh. The solution at vertex R on the coarse mesh is found by linear interpolation of the values at vertices 1, 2, and 3. Corrections are linearly interpolated from vertices P, Q, and R to vertex 1. The weights used for this interpolation are also used to distribute the residual at vertex 1 to the coarse mesh vertices P, Q, and R. This ensures that the residual transfer is conservative, because the sum of these interpolation weights is always one.

The interpolation weights can easily be found once it is known which cell on the coarse (resp. fine) mesh contains a given vertex from the fine (resp. coarse) mesh. A naive search algorithm for finding the appropriate cell would require $\mathcal{O}(\mathcal{N}^2)$ time, which quickly becomes prohibitive for large meshes in two dimensions and is unthinkable in three dimensions. Instead, a list of cells whose Cartesian bounding boxes include a given vertex is found using an alternating digital tree search, as described by Morgan et al.[20] The tree can be created in $\mathcal{O}(\mathcal{N}\log\mathcal{N})$ time, whereas the list of cells that could contain a given vertex can be assembled in $\mathcal{O}(\log\mathcal{N})$ time and has length of $\mathcal{O}(1)$. Checking each cell in the list to determine which one actually contains a vertex is trivial. Finally, the interpolation weights for vertex R are simply its barycentric coordinates with respect to the triangle $\triangle 123$. The total complexity of the algorithm to compute all of the interpolation weights is of $\mathcal{O}(\mathcal{N}\log\mathcal{N})$.

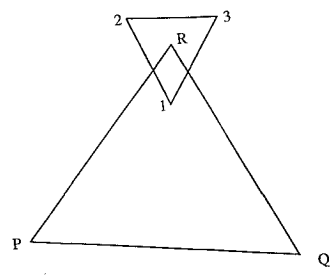

Fig. 2 Interpolation stencil for mesh-to-mesh transfer.

Coarse Mesh Generation

The generation of very coarse unstructured meshes is quite difficult, especially in three dimensions, where preservation of boundary integrity becomes a problem. Several automated approaches to this task have been developed. Guillard[21] generates coarse meshes by selecting a subset of the fine mesh vertices and retriangulating them. This approach is not guaranteed to produce valid, boundary-preserving triangulations in three dimensions. More recently, Venkatakrishnan and Mavriplis[9] and Mavriplis and Venkatakrishnan[10] have introduced a scheme in which adjacent fine-mesh control volumes are fused together to form coarse-mesh control volumes. This approach requires great care in the construction of the coarse-mesh discretization.

A new method for automatically generating coarse meshes that addresses both of these issues has been developed. The coarse triangulation is derived incrementally from the fine triangulation by removing vertices, guaranteeing a valid triangulation at every step in the procedure. Furthermore, the existence of a coarse triangulation as opposed to agglomerated control volumes allows the use of the same discretization scheme on all meshes.

Coarse mesh vertices are selected using a maximal independent set (MIS) approach; that is, a subset of the fine mesh vertices is selected such that no two selected vertices share an edge. This procedure is first applied to the boundary discretization. Care is taken to preserve important geometric features—typically sharp corners. Additional boundary vertices are retained as necessary to ensure that at least three vertices define each closed boundary curve. An automatic check for chains of edges normal to the surface allows the retention of quasistructured regions of a mesh near a body around which a viscous mesh has been generated. Finally, a maximal independent set of the remaining vertices is also marked for retention. During this final phase, boundary vertices and vertices in normal chains do not have their retention status changed.

A randomized optimization procedure is used to approximately maximize the size of the maximal independent set of vertices selected for retention. Maximizing the size of this set is done to trade a decrease in the work per iteration on the coarse mesh for better mesh quality and therefore improved convergence per iteration. The heuristic optimization process tries to exchange one selected vertex for an adjacent, unselected vertex that has no other selected neighbors. The decision about whether to exchange a selected vertex for an unselected one is random, with a bias towards selecting the vertex with the larger number of selected second neighbors. This bias improves the compactness of the vertices retained and often allows an additional vertex to be marked for retention. The randomness of the algorithm prevents lock-in on locally compact groupings of retained vertices that hurt compactness on a larger scale. This in turn increases the number of vertices selected for inclusion in the coarse mesh over a deterministic movement algorithm. The creation of this maximal independent set requires the use of vertex adjacency

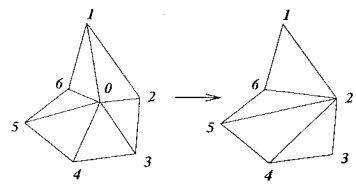

Fig. 3 Mesh contraction schematic.

information, in addition to the usual edge data structure for unstructured meshes.

Once the coarse mesh vertices have been selected, the coarse mesh is generated by mesh contraction. The left half of Fig. 3 shows a vertex 0 that is to be removed from the mesh and its immediate neighbors in the mesh. Vertex 0 will be removed by sliding it along the edge $\overline{02}$ to vertex 2. In the process, cells $\triangle 012$ and $\triangle 023$ are removed, as are edges $\overline{01}$ and $\overline{03}$. The resulting mesh fragment is shown in the right half of Fig. 3. After a complete pass has been made, attempting to remove each candidate vertex by edge contraction, edge swapping is done to locally minimize the maximum angle between edges. As implemented, mesh contraction requires the use of cell-to-edge connectivity data, as does edge swapping.

Some care must be taken to ensure that invalid triangulations do not result from this procedure. For example, vertex 0 cannot be moved onto vertex 1 or vertex 5 without causing one of the resulting cells (cell $\triangle 156$) to have a negative area. This case is easy to check for and in many cases a single edge swap will eliminate it. For example, if vertices 0, 4, 5, and 6 are reconfigured to form cells $\triangle 046$ and $\triangle 456$, vertex 0 could be removed by contracting edge $\overline{01}$. No cases have yet been found in two dimensions where a vertex tagged for removal via edge contraction could not be successfully removed by contraction of some incident edge, although no proof is available to show that this is indeed a general property. (This same algorithm extends fairly easily to three dimensions; in this case, however, the existence of an edge that can be shrunk is definitely not guaranteed, since not all polyhedral regions in three space can be tetrahedralized without adding an interior vertex.)

Results

The first case used for benchmarking the convergence acceleration of this scheme was a NACA 0012 airfoil at Mach number of 0.5 and zero angle of attack. The solution to this case is of course well known and of little intrinsic interest; the case was chosen because it is a smooth flow, which will allow comparison of convergence rates for a variety of conditions, including the presence or absence of limiting in the reconstruction of the solution. Figure 4 shows a close-up of the fine mesh, which contains 3084 vertices and 5971 cells. The far-field boundary for each mesh is a square 40 chords on a side and centered at the leading edge of the airfoil. Three coarse meshes were generated, containing 759, 186, and 50 vertices. The first coarse mesh is shown in Fig. 5 as an example of the mesh quality typically obtained by the automatic coarse mesh generator. The mesh quality tends to deteriorate for very coarse meshes because the number of interior points is very small.

The solutions obtained for this case were independent of the number of meshes used. A lift coefficient of less than 10^{-5} and a drag coefficient of $2.2 \cdot 10^{-4}$ were computed.

Unless otherwise noted, all cases were run using a global Courant–Friedrichs–Lewy (CFL) number of 2.5, sawtooth W-cycles, first-order discretization on coarse meshes, and Venkatakrishnan's limiter. For each case, the same starting solution was used; this solution was obtained using full multigrid, performing 15 sawtooth W-cycles at each level. The solution obtained on the second finest mesh was interpolated onto the finest mesh and used as a starting solution. This comparison is actually somewhat biased in favor of the single-mesh case, because obtaining a comparable solution on a single mesh would be time consuming. To give fair timing comparisons among cases, all timings are given in terms of work units, where a work unit is the time required to perform one multistage time step on the finest mesh. Convergence rates are compared quantitatively using the convergence factor, the ratio by which the maximum residual is decreased per work unit.

Figure 6 shows the effect on convergence rate of increasing the number of meshes in the multigrid hierarchy. For a single mesh, convergence is quite slow, with the residual being reduced by a factor of 0.9983 per work unit asymptotically. At the other end of the scale, the use of either three or four meshes leads a convergence rate of 0.968 per work unit. The convergence rate on four meshes corresponds to

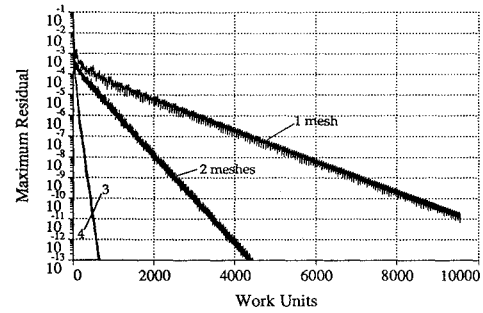

Fig. 6 Convergence rate for various numbers of coarse meshes, subsonic case.

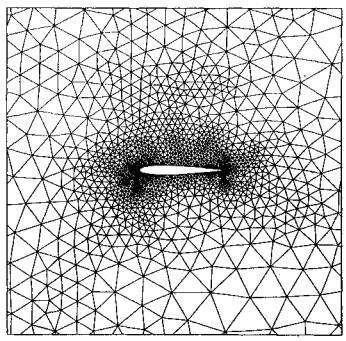

Fig. 4 Finest mesh, subsonic case (3084 nodes).

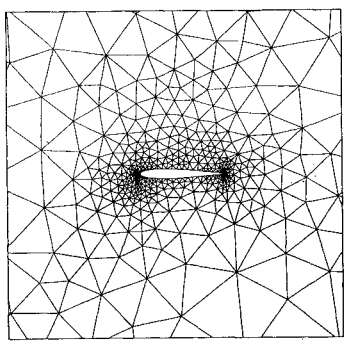

Fig. 5 First coarse mesh, subsonic case (759 nodes).

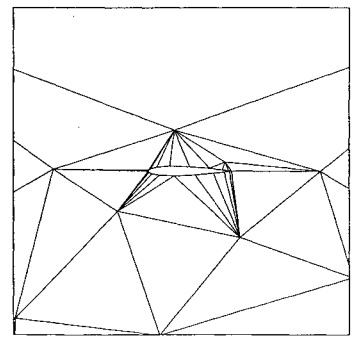

Fig. 7 Coarsest mesh for subsonic case, showing poorly conditioned cells.

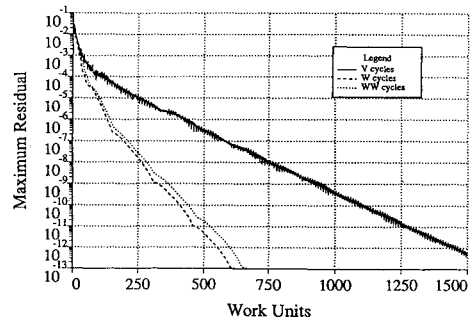

Fig. 8 Convergence rate for various types of multigrid cycles.

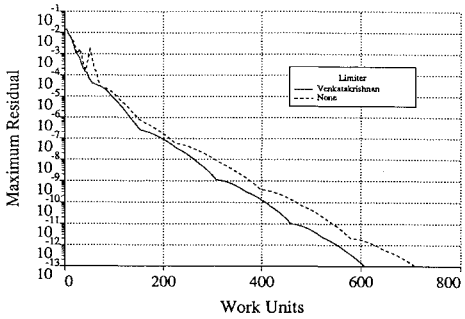

Fig. 9 Effect on convergence rate of the use of a limiter.

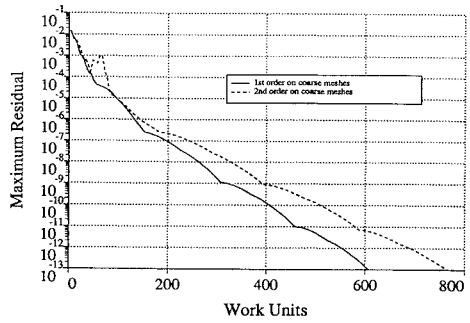

Fig. 10 Effect of coarse mesh discretization on convergence rate.

reducing the residual by an order of magnitude every 71 work units or about 20 sawtooth W-cycles. This is an improvement of a factor of 20 over the single mesh case. The leveling of convergence rate with excessive mesh coarseness is probably related to the fact that most of the cells on the coarsest mesh have extremely high aspect ratio (see Fig. 7). In addition to being poorly conditioned, these cells are unlikely to allow smooth, accurate interpolation to the next finer mesh.

Next let us compare the relative efficiency of various cycle types. Figure 8 shows convergence histories for sawtooth V-cycles, sawtooth W-cycles, and sawtooth "WW"-cycles. The WW-cycle transfers from a given mesh to the next coarser mesh three times, whereas the W-cycle does so twice and the V-cycle once. The best convergence rate of the three is for the W-cycle, which is 2.5 times as efficient as the V-cycle, and 15% more efficient than the WW-cycle. Experiments with other cases indicate that this result is not particularly problem dependent; W-cycles appear to be much more efficient in general than V-cycles and slightly more efficient than WW-cycles. This is because the V-cycle spends a larger proportion of its time on the finest mesh, where convergence is slow, and the WW-cycle apparently does not have recent enough information about what the multigrid forcing terms should be. The W-cycle seems to strike a balance between these two demands.

The effect of the limiter on convergence rate was investigated, and it was found that the limiter actually improves convergence rate slightly for this case when convergence rate is based on work units (see Fig. 9). Both cases take the same number of multigrid cycles, however. Since the limiter is not used on coarse meshes, where the solution is only first-order accurate, the presence of the limiter on the fine mesh rescales the work unit and thus causes this small change. Because the use of a limiter does not inhibit convergence, convergence rates for nonsmooth flows may also be expected to be good.

Finally, the effect on convergence rate of the choice of discretization on the coarse meshes is shown in Fig. 10. The choice of coarse-mesh discretization has no effect on the solution, provided that the fine-mesh discretization is the same. The use of first-order discretization on the coarse meshes leads to an improvement of over 20% in the asymptotic convergence rate. The second-order discretization actually requires about 3% fewer cycles, but the cost of gradient evaluation and limited reconstruction more than offsets this advantage.

Case 1 of the AGARD suite[22] was computed as a second benchmark. In this case, a NACA 0012 airfoil is placed in a Mach 0.8 flow at 1.25-deg angle of attack. A close-up of the fine mesh used

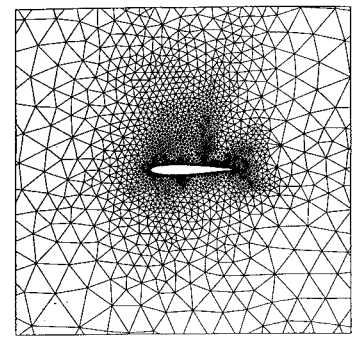

Fig. 11 Finest mesh, transonic case (4156 nodes).

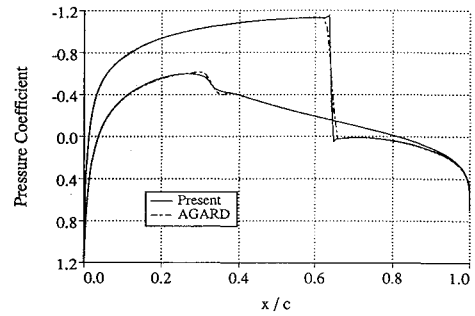

Fig. 12 Surface pressure coefficient for NACA 0012, $\alpha = 1.25$, $M = 0.8$.

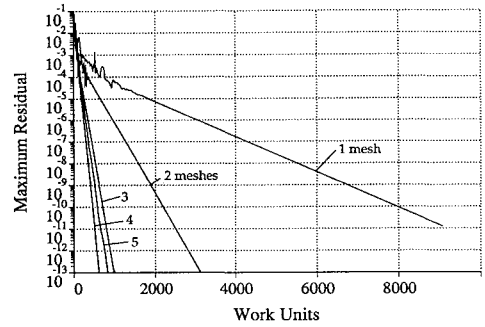

Fig. 13 Convergence rate for various numbers of coarse meshes, transonic case.

for this case is shown in Fig. 11. The far-field boundary is 100 chords square. The finest mesh was generated by solution-adapting a mesh that was known to have a good distribution of surface vertices for this case and therefore was expected to give a good approximate solution in terms of shock location. The adaptation was keyed to divided differences of density. The finest mesh has 4156 vertices and 8071 cells. The first and second coarse meshes were generated by the process described in the previous section, with Laplacian smoothing used as a postprocessing step. The two coarsest meshes in this series are derived from the two coarsest meshes for the previous subsonic case. The four coarse meshes have 997, 224, 186, and 50 vertices.

All cases were run using a CFL number of 2.5 on all meshes, sawtooth W-cycles, first-order discretization on coarse meshes, and Venkatakrishnan's limiter. For each case, the same starting solution was used; this solution was obtained in the same way as for the subsonic case.

Figure 12 shows surface pressure coefficient compared with the result tabulated in the AGARD report.[22] The upper shock is in the correct location and is captured within two cells. The lower shock is weak enough that adaptation did not add vertices in this region, and so the lower shock is less well resolved in the present result than in the accepted AGARD solution. The lift coefficient (0.361) compares well with a range of published results (0.359 ±0.010)(Ref.22), as do the drag (0.0227; 0.0229 ±0.0008) and moment (−0.0399; −0.0391 ±0.0025) coefficients.

Figure 13 shows the effect on convergence rate of increasing the number of meshes in the multigrid hierarchy. For a single mesh, the

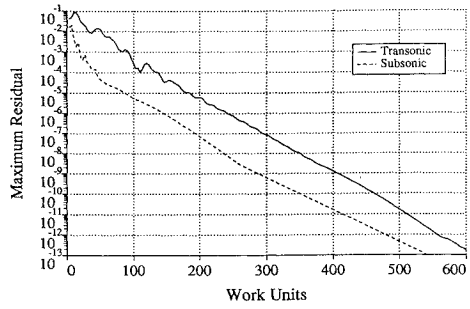

Fig. 14 Comparison of convergence rate for transonic and subsonic flows on the same mesh system.

convergence factor is 0.9981 per work unit. The convergence factor is best for four meshes, at 0.957 per work unit, with the use of an additional mesh hurting the convergence rate as before. This represents an improvement of a factor of 23 over the single mesh case.

To get a good comparison between convergence rates for smooth and nonsmooth flows, a subsonic case ($M = 0.5$ and $\alpha = 0$) was computed using the mesh system generated for the transonic case. The convergence rates for four meshes are compared in Figure 14. The asymptotic convergence rate for the transonic case is actually slightly better than for the subsonic case. However, the subsonic case converged more quickly overall. This is because shock movement and/or formation slows the initial convergence of the transonic case. Whether the transonic or subsonic case converges faster is not exactly the point, however; this could easily change on a case-by-case basis. The essential feature is that the convergence rates for both cases are comparable, and indeed both are comparable to the convergence rate for the previous inviscid case. This gives a strong indication that this multigrid implementation will perform well for a variety of inviscid flows.

Conclusions

Multigrid acceleration has been implemented for an upwind flow solver on unstructured meshes. The flow solver is a straightforward implementation of Barth and Jespersen's unstructured scheme, with least-squares linear reconstruction and a directional implementation of Venkatakrishnan's limiter. The multigrid scheme itself is designed to work on mesh systems that are not nested, allowing great flexibility in generating coarse meshes and in adapting fine meshes.

A new scheme for automatic generation of coarse meshes is presented. The coarse mesh vertices form a maximal independent subset of the fine mesh vertices. Fine mesh vertices are removed one by one to create the coarse mesh. The incremental nature of this process guarantees a valid boundary-preserving triangulation at each step of generating the coarse mesh; this property is expected to be very helpful in generating coarse meshes for complex three-dimensional geometries.

A thorough examination of factors affecting multigrid convergence rate for a subsonic inviscid flow has been done. It was found that the multigrid scheme improves the convergence rate by a factor of 20 for this case, with some dropoff in convergence acceleration as the coarse mesh becomes too coarse. W-cycles were shown to be more than twice as efficient as V-cycles in converging to a steady-state solution. The presence of Venkatakrishnan's limiter had no effect on the number of multigrid cycles required to reach convergence. The use of first-order upwind spatial discretization on coarse meshes leads to a noticeable improvement in convergence rate as compared with using second-order discretization on coarse meshes.

A transonic flow was also computed, and the multigrid scheme was found to improve convergence by a factor of 23 compared with the single mesh flow solver. The rate of convergence for subsonic and transonic flow on the same set of meshes was nearly identical after the initial transients died out; the transient for transonic flow—related to the formation and movement of shocks—took significantly longer to die out.

The present multigrid scheme is very successful in reducing computational time for inviscid flows in the subsonic and transonic regime. Work is currently under way to verify the performance of the scheme for viscous flows, to investigate the application of the scheme to implicit solvers, and to extend both the multigrid flow solver and the coarse mesh generation procedure to three dimensions.

References

[1]Jameson, A., "Acceleration of Transonic Potential Flow Calculations on Arbitrary Meshes by the Multiple Grid Method," AIAA Paper 79-1458, July 1979.

[2]Ni, R.-H., "A Multiple-Grid Scheme for Solving the Euler Equations," *AIAA Journal*, Vol. 20, No. 11, 1982, pp. 1565–1571.

[3]Jameson, A., "Solution of the Euler Equations for Two Dimensional Transonic Flow," *Applied Mathematics and Computation*, Vol. 13, Nov. 1983, pp. 327–355.

[4]Yoon, S., Chang, L., and Kwak, D., "Multigrid Convergence of an Implicit Symmetric Relaxation Scheme," AIAA Paper 93-3357, July 1993.

[5]Caughey, D., "Implicit Multigrid Euler Solutions with Symmetric Total-Variation-Diminishing Dissipation," AIAA Paper 93-3358, July 1993.

[6]Mavriplis, D., and Jameson, A., "Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes," Inst. for Computer Applications in Science and Engineering, ICASE Rept. 87-53, Hampton, VA, July 1987.

[7]Mavriplis, D., "Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Meshes," *AIAA Journal*, Vol. 28, No. 2, 1990, pp. 213–221.

[8]Mavriplis, D., "Three Dimensional Unstructured Multigrid for the Euler Equations," Inst. for Computer Applications in Science and Engineering, ICASE Rept. 91-41, Hampton, VA, May 1991.

[9]Venkatakrishnan, V., and Mavriplis, D. J., "Agglomeration Multigrid for the 3D Euler Equations," AIAA Paper 94-0069, Jan. 1994.

[10]Mavriplis, D. J., and Venkatakrishnan, V., "Agglomeration Multigrid for Viscous Turbulent Flows," AIAA Paper 94-2332, June 1994.

[11]Barth, T. J., and Jespersen, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA Paper 89-0366, Jan. 1989.

[12]Barth, T. J., "Aspects of Unstructured Grids and Finite-Volume Solvers for the Euler and Navier–Stokes Equations," *Unstructured Grid Methods for Advection-Dominated Flows*, AGARD, AGARD-R-787, Neuilly sur Seine, France, 1992, Chap. 6.

[13]Barth, T. J., "Recent Developments in High Order K-Exact Reconstruction on Unstructured Meshes," AIAA Paper 93-0668, Jan. 1993.

[14]Venkatakrishnan, V., "On the Accuracy of Limiters and Convergence to Steady-State Solutions," AIAA Paper 93-0880, Jan. 1993.

[15]Aftosmis, M. J., Gaitonde, D., and Tavares, T. S., "On the Accuracy, Stability, and Monotonicity of Various Reconstruction Algorithms for Unstructured Meshes," AIAA Paper 94-0415, Jan. 1994.

[16]Roe, P., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[17]Mavriplis, D., and Jameson, A., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," AIAA Paper 87-0353, Jan. 1987.

[18]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time Stepping Schemes," AIAA Paper 81-1259, June 1981.

[19]Brandt, A., "Guide to Multigrid Development," in *Multigrid Methods*, edited by W. Hackbusch and U. Trottenberg, Vol. 960, Lecture Notes in Mathematics, Springer–Verlag, 1982, pp. 220–311.

[20]Morgan, K., Peraire, J., and Peiró, J., "Unstructured Grid Methods for Compressible Flows," *Unstructured Grid Methods for Advection-Dominated Flows*, AGARD, AGARD-R-787, Neuilly sur Seine, France, 1992, Chap. 5.

[21]Guillard, H., "Node Nested Multigrid with Delaunay Coarsening," Institut National de Recherche en Informatique et en Automatique, INRIA Rept. 1898, Valbonne, France, 1993.

[22]Anon., *Test Cases for Inviscid Flow Field Methods*, AGARD Fluid Dynamics Panel, AGARD Advisory Rept. AR-211, May 1985.

[23]Pulliam, T. H., and Barton, J. T., "Euler Computations of AGARD Working Group 07 Airfoil Test Cases," AIAA Paper 85-0018, Jan. 1985.