AIAA JOURNAL
Vol. 33, No. 4, April 1995

Agglomeration Multigrid for the Three-Dimensional Euler Equations

V. Venkatakrishnan* and D. J. Mavriplis†
NASA Langley Research Center, Hampton, VA 23681

A multigrid procedure that makes use of coarse grids generated by the agglomeration of control volumes is advocated as a practical approach for solving the three-dimensional Euler equations on unstructured grids about complex configurations. It is shown that the agglomeration procedure can be tailored to achieve certain coarse grid properties such as the sizes of the coarse grids and aspect ratios of the coarse grid cells. The agglomeration is done as a preprocessing step and runs in linear time. The implications for multigrid of using arbitrary polyhedral coarse grids are discussed. The agglomeration multigrid technique compares very favorably with existing multigrid procedures both in terms of convergence rates and elapsed times. The main advantage of the present approach is the ease with which coarse grids of any desired degree of coarseness may be generated in three dimensions, without being constrained by considerations of geometry. Inviscid flows over a variety of complex configurations are computed using the agglomeration multigrid strategy.

I. Introduction

OVER the last few years, the multigrid method has been demonstrated as an efficient means for obtaining steady-state solutions to the Euler equations on unstructured meshes in two and three dimensions.[1–9] The main difficulty with multigrid techniques on unstructured meshes is the generation of coarse grids. In the case of structured grids, coarse grids are easily derived from a given fine grid by omitting alternate grid lines in each coordinate direction. In the case of unstructured grids, three different approaches can be adopted.

The first approach begins with a coarse mesh definition and generates finer grids by refinement.[5,6] The advantage is that the intergrid operators become simple because of the nesting of grids. There are several disadvantages with this approach. The first drawback is the dependence of the fine grid point distribution on the coarse levels. Ideally, the fine grid points should be distributed to capture features of the geometry and to control the aspect ratios of the grid cells and grid stretching, whereas in this method, the coarsest grid determines these features. Furthermore, the coarsest level that can be used in the multigrid is predetermined. This may be too fine if the geometry is to be captured on this coarse grid.

The second approach uses non-nested unstructured grids as coarse grids and has been shown to be quite successful in both two- and three-dimensional unstructured grid computations.[1–4] Grids of varying degrees of coarseness are generated independently using any given grid generation strategy. Piecewise-linear interpolation operators for the transference of flow variables, residuals, and correction are derived during a preprocessing step by using efficient search procedures. Since the coarse grids are generated independently from one another, the coarse grids are not nested with the fine grids and, in general, do not even contain points in common with the fine levels. A variant of this approach chooses a subset of the fine grid points and retriangulates them.[7] The grids are still non-nested even though the coarse grid points are contained in the fine grid.

The approaches just outlined share a common problem, that of generating coarse grids. For complex geometries, especially in three dimensions, constructing coarse grids that faithfully represent the complex geometries can become a difficult proposition. In other words, there comes a point when the critical features of geometry are too fine to be captured by the coarse grid, with the result that the geometry or sometimes even the topology is altered. The requirement to generate not just one grid but multiple grids that preserve the geometry places too much of a burden on a user. One approach that circumvents this problem is the generation of coarse grids through agglomeration as developed by Lallemand et al.[8] and independently by Smith.[9] Lallemand et al.[8] use a base scheme that stores the variables at the vertices of the triangular mesh, whereas Smith[9] uses a scheme that stores the variables at the centers of triangles. In the present work, a vertex-based scheme is employed. Two-dimensional triangular grids contain twice as many cells as vertices (neglecting boundary effects) and, in practice, three-dimensional tetrahedral meshes contain five to seven times more cells than vertices. Thus, on a given grid, a vertex scheme incurs substantially less computational overhead than a cell-based scheme. Increased accuracy may be expected from a cell-based scheme, since this involves the solution of a larger number of unknowns. However, the increase in accuracy does not appear to justify the additional computational overheads, particularly in three dimensions.

The central idea behind the agglomeration strategy of Lallemand et al.[8] is to fuse or agglomerate the control volumes for the vertices using heuristics. The centroidal dual, composed of segments of the median planes of the triangulation, is a collection of the control volumes over which the Euler equations in integral form are solved. For simple geometries and on relatively coarse grids, Lallemand et al. were able to show that the agglomerated multigrid technique performed as well as the multigrid technique which makes use of non-nested coarse grids. However, the convergence rates, especially for the second-order accurate version of the scheme, appeared to degrade somewhat. Furthermore, the validation of such a strategy for more complicated geometries and using much finer grids, as well as the incorporation of viscous terms for the Navier-Stokes equations remains, to be demonstrated. The work of Smith[9] constitutes the basis of a commercially available computational fluid dynamics code and as such has been applied to a number of complex geometries.[10] However, consistently competitive multigrid convergence rates have yet to be demonstrated.

In the present work, the agglomeration multigrid strategy is explored further. The issues involved in a proper agglomeration and the implications for the choice of the restriction and prolongation operators are addressed. Inviscid flows over nonsimple two- and three-dimensional geometries involving a large number of unknowns are computed with the agglomeration multigrid strategy. This approach is compared with the unstructured multigrid algorithm of Refs. 1 and 2 which makes use of non-nested coarse grids.

Received Dec. 31, 1993; presented as Paper 94-0069 at the AIAA 32nd Aerospace Sciences Meeting, Reno, NV, Jan. 10–13, 1994; revision received June 17, 1994; accepted for publication June 20, 1994. Copyright © 1994 by V. Venkatakrishnan and D. J. Mavriplis. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

*Senior Staff Scientist, Institute for Computer Applications in Science and Engineering. Member AIAA.

†Senior Staff Scientist, Institute for Computer Applications in Science and Engineering. Member AIAA.

Convergence rates as well as CPU times are compared using both methods.

II. Governing Equations and Discretization

The Euler equations in integral form for a control volume Ω with boundary $\partial\Omega$ read

$$\frac{\partial}{\partial t}\int_{\Omega} \boldsymbol{u}\,\mathrm{d}v + \oint_{\partial\Omega} \boldsymbol{F}(\boldsymbol{u},\boldsymbol{n})\,\mathrm{d}S = 0 \qquad (1)$$

Here $\boldsymbol{u}$ is the solution vector comprised of the conservative variables: density, the components of momentum, and total energy. The vector $\boldsymbol{F}(\boldsymbol{u},\boldsymbol{n})$ represents the inviscid flux vector for a surface with normal vector $\boldsymbol{n}$. Equation (1) states that the time rate of change of the variables inside the control volume is the negative of the net flux of the variables through the boundaries of the control volume. This net flux through the control volume boundary is termed the residual. In the present scheme, the variables are stored at the vertices of a mesh composed of simplices i.e., triangles in two dimensions and tetrahedra in three dimensions. Most of the description in the paper will deal with the three-dimensional situation, the reduction to the two-dimensional situation being obvious. The control volumes are nonoverlapping polyhedra which surround the vertices of the mesh. They form the dual of the mesh, which is composed of segments of median planes. Associated with each edge of the original mesh is a (segmented) dual face. The contour integrals in Eq. (1) are replaced by discrete path integrals over the faces of the control volume which are computed by using the trapezoidal rule. This technique can be shown to be equivalent to using a piecewise-linear finite element discretization. Using edge-based data structures, the path integrals can be cast as a loop over the edges, even in three dimensions as shown by Mavriplis.[2] For dissipative terms, a blend of Laplacian and biharmonic operators is employed. The Laplacian term acts only in the vicinity of shocks and is inactive elsewhere, whereas the biharmonic term acts only in regions of smooth flow. A multistage Runge-Kutta scheme is used to advance the solution in time. In addition, local time stepping, enthalpy damping, and residual averaging are used to accelerate convergence.[11] In previous work,[1,2] as well as in the present work, only the Laplacian dissipative term (with constant coefficients) is used on the coarse grids. Thus, the fine grid solution itself is second-order accurate, whereas the solver is only first-order accurate on the coarse grids.

III. Details of Agglomeration

The object of agglomeration is to derive coarse grids from a given fine grid. In graph theoretical terms, if the initial grid is interpreted as a graph, the agglomeration problem is that of finding a maximal independent set with certain desirable properties. A subset of the vertices of a graph is termed an independent set if no two vertices in the set are adjacent. An independent set is maximal if any vertex not in the set is dominated by (adjacent to) at least one vertex in it. A desirable property for the coarse grids in multigrid is that the number of grid cells should decrease by a nearly constant factor when moving from a fine to a coarse grid. This factor is four in two dimensions and eight in three dimensions. The graph problem reduces to finding the maximal independent graph with the minimum cardinality (size) and falls under the category of intractable problems termed NP-complete. Hence, a heuristic algorithm is used to derive the coarse grids.

The agglomeration is accomplished by using a greedy-type frontal algorithm. It is a variation on the one used by Lallemand et al.[8] We call this the isotropic version, since it fuses in step 2 all of the neighboring fine grid control volumes that are not already fused. It is outlined as follows:

1) Pick a starting vertex on one of the surface elements.

2) Agglomerate the control volumes associated with its neighboring vertices which are not already agglomerated.

3) Define a front as comprised of the exterior faces of the agglomerated control volumes. Place the exposed edges (duals to the faces) in a queue.

4) Pick the new starting vertex as the unprocessed vertex incident to a new starting edge which is chosen from the following choices given by order of priority: an edge on the front that is on the solid wall, an edge on the solid wall, an edge on the front that is on the far-field boundary, an edge on the far-field boundary, and the first edge in the queue.

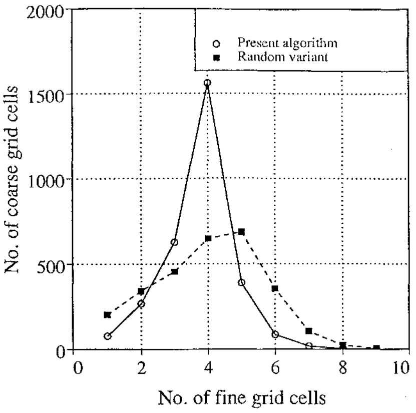

Fig. 1 Number of coarse grid cells vs the number of fine grid cells they contain.

5) Go to step 2 until the control volumes for all vertices have been agglomerated.

There are many other ways of choosing the starting vertex in step 4 of the algorithm, but we have found the preceding strategy to be the best. The efficacy of the agglomeration algorithm can be characterized by a histogram of the number of fine grid cells comprising each coarse grid cell. In two dimensions, ideally, each coarse grid cell will be made up of exactly four fine grid cells. The various strategies can be evaluated by how close they come to this ideal situation. One variation is to pick the starting edge randomly from the edges currently on the front. Figure 1 shows a plot of the number of coarse grid cells as a function of the number of fine grid cells they contain for a triangular mesh with 11,340 vertices, with the agglomeration algorithm just described, and with the variant. It is clear that our agglomeration algorithm is superior to the variant. The number of singletons (coarse grid cells having exactly one fine cell) is also much smaller with our algorithm compared to the variant. We have also investigated another variation of the algorithm where the starting vertex in step 4 is randomly picked from the field; this turns out to be much worse. It is possible to identify the singleton cells and agglomerate them with the neighboring cells, but this has not been done. Finally, it should be noted that with the isotropic version of the algorithm, one has no control over the shapes or the aspect ratios of the coarse grid cells.

To have some control over the aspect ratios of the coarse grid cells, we have developed a variant loosely termed the semicoarsened version. This is similar in spirit to semicoarsening for highly stretched structured grids, where the coarsening is done in the direction of stretching. The algorithm produces more uniform coarse grids and is expected to be beneficial especially for viscous calculations. This algorithm is a combination of breadth-first and depth-first searches and is more expensive than the isotropic version. In the semicoarsened version, steps 2 and 3 of the isotropic algorithm are replaced by the following steps:

1) For each starting vertex, form a list of neighboring control volumes.

2) Fuse the control volume from the list that maximizes the aspect ratio of the coarse grid cell defined as V/S^n, where V is the volume, S is the surface area, and n is 2 in two dimensions and 1.5 in three dimensions.

3) Update the front by adding the exposed edges (duals to the exposed faces) to the queue.

4) Update the list of neighboring control volumes.

5) Go to step 2 until a prespecified number of control volumes are fused into a coarse grid cell.

We compare the two versions of the agglomeration algorithm by inspecting the coarse grids obtained on a small test problem. The fine grid about a NACA 0012 airfoil contains 4224 vertices. The third-level coarse grids obtained with the isotropic and the semicoarsened algorithms are displayed in Figs. 2 and 3, and they contain 80 and

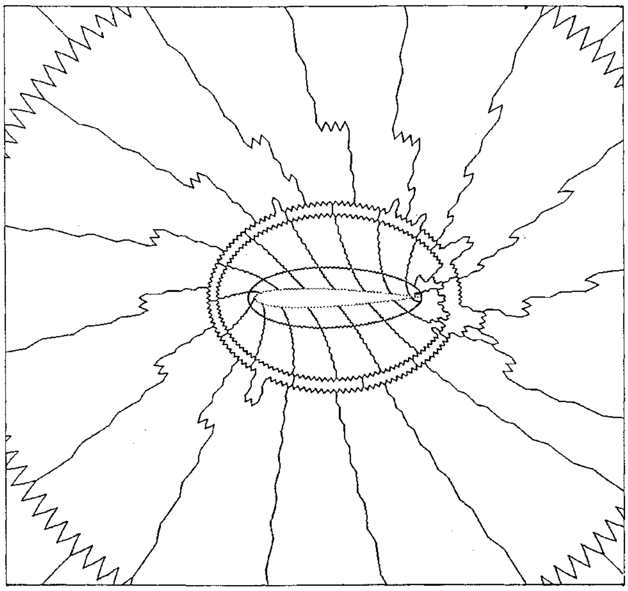

Fig. 2 Third-level coarse grid obtained with the isotropic agglomeration algorithm for a fine grid with 4224 vertices.

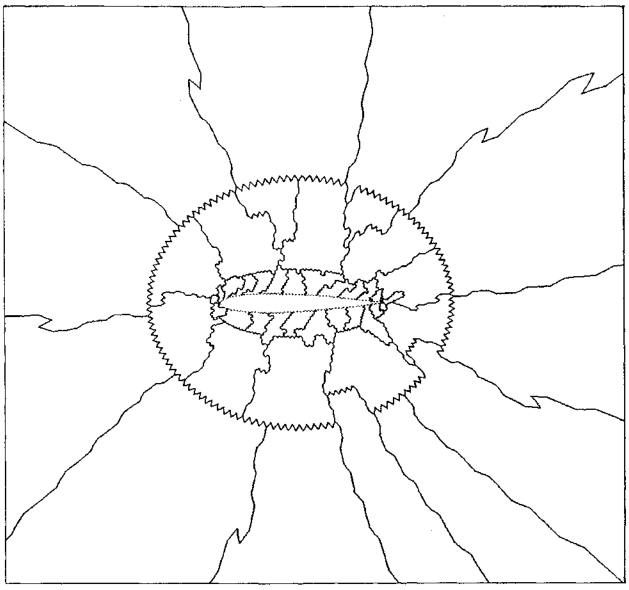

Fig. 3 Third-level coarse grid obtained with the semicoarsened agglomeration algorithm for a fine grid with 4224 vertices.

72 cells, respectively. The grid obtained with the semicoarsened version is more uniform and has more favorable aspect ratios.

A three-dimensional application of the isotropic agglomeration algorithm is shown in Figs. 4a–4c. The fine grid about an ONERA M6 wing contains 9428 vertices. Figure 4a shows the dual to the surface grid. The coarse grids contain 1466, 253, and 46 vertices, respectively. Figures 4b and 4c show the level-1 and level-3 surface grids.

The procedures outlined earlier are applied recursively to create coarser grids. The boundaries between the control volumes on the coarse grids are composed of the edges/faces of the fine grid control volumes. In two dimensions, we have observed that the number of such edges only decreases by a factor of two when going from a fine to a coarse grid, even though the number of vertices decreases by a factor of four. Since the computational load is proportional to the number of edges, this is unacceptable in the context of multigrid. However, if we recognize that the multiple edges separating two control volumes can be replaced by a single edge connecting the end points, then the number of edges does go down by a nearly a factor of four. This is illustrated in Fig. 5 which shows the fine

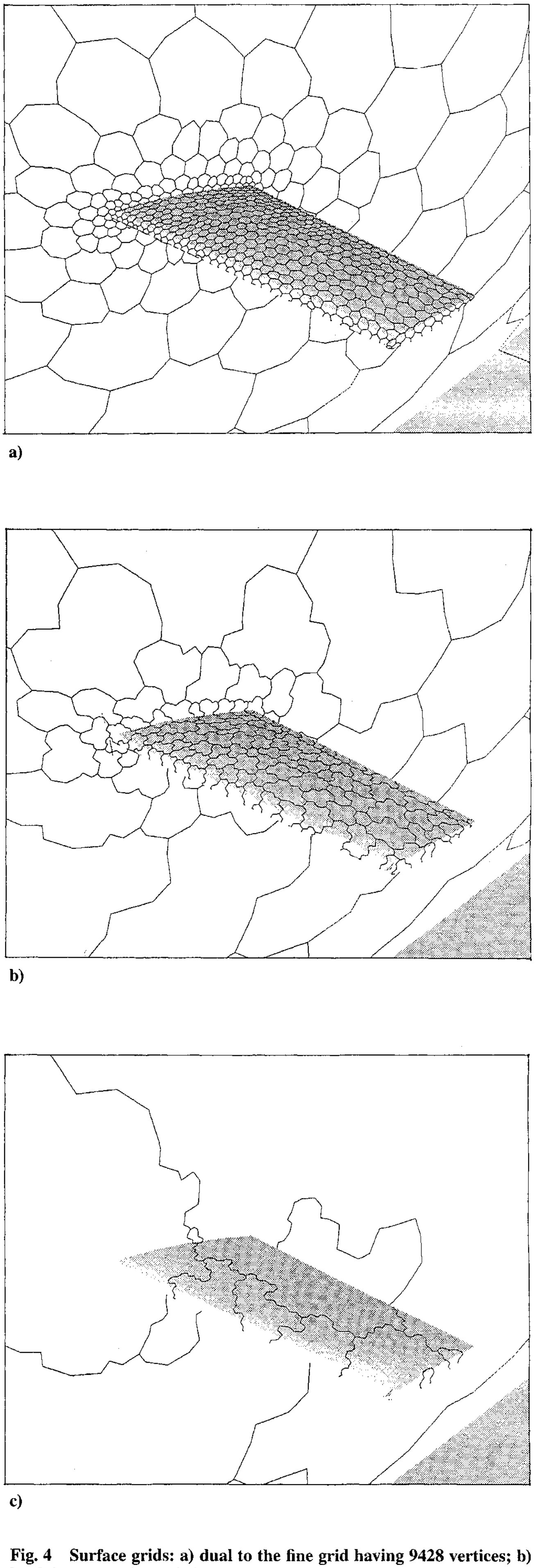

Fig. 4 Surface grids: a) dual to the fine grid having 9428 vertices; b) level-1 coarse grid; and c) level-3 coarse grid.

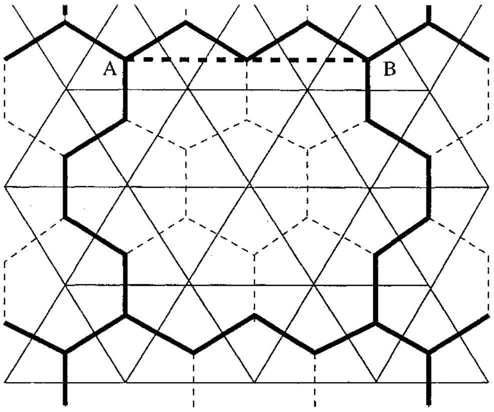

Fig. 5 Example of an agglomerated coarse grid.

grid edges (solid lines), fine grid dual edges (dashed lines), and the edges for the coarse grid control volumes (thick solid lines). The edges connecting A and B can be effectively replaced by a single straight edge shown by the thick dashed line. A similar construction can be made in three dimensions as well. There is no approximation involved in this step if the geometry enters the flux function in a linear fashion. However, in most instances, the geometry enters the flux function in a nonlinear fashion. Some flux functions require the absolute value of the metric terms multiplied by velocity components to compute a dissipation coefficient. Other flux formulas, such as Roe's approximate Riemann solver[12] involve the metrics in a nonlinear manner as well. In these cases, the construction already described is a good approximation, especially since only a first-order discretization is used on the coarse grids. As a result of making this approximation, the work decreases by nearly a constant factor (4 in 2d and 8 in 3d) when moving from a fine to a coarse grid.

Both versions of the algorithm have been optimized to run in linear time. They employ hashing to combine the multiple fine grid control volume faces separating two coarse grid cells into one planar face. This has enabled us to run very large three-dimensional problems (tetrahedral grids with over 800,000 vertices) in a reasonable amount of time. The semicoarsened version is more expensive than the isotropic version but creates more uniform grids. Another advantage of the semicoarsened option is that an arbitrarily sized coarse grid can be created from a fine grid, since the number of fine grid cells comprising a coarse grid cell is specified as an input. This flexibility can be used effectively to study the tradeoff between the convergence of the multigrid process and the amount of work performed on the coarse grid, although this aspect has not been investigated in this paper. Whereas the isotropic version of the algorithm creates coarse grids that are maximal independent sets, the semicoarsening version does not in general yield maximal independent sets or, for that matter, even independent sets. The three-dimensional versions of the algorithm require 1200 bytes of memory per fine grid vertex on a Sun Sparc 10 workstation. The times required to create four coarse grids by agglomeration from a three-dimensional tetrahedral grid with 50,000 fine grid vertices on this workstation are 63 and 128 s with the isotropic and the semicoarsened versions, respectively. The output of the agglomeration is a list of fine grid cells belonging to each coarse grid cell and edge coefficients, which represent the projected areas of the planar control-volume faces onto the coordinate planes.

IV. Details of Multigrid

The base scheme needs to be modified to handle arbitrary polyhedral coarse grids. The time-step calculation and the computations of the Euler and dissipative fluxes are recast in such a way as to only utilize the volumes and the edge coefficients. In the original scheme, an isotropic dissipative coefficient based on the spectral radius integrated over the control volume faces was used. For very coarse and highly nonuniform polyhedral grids, we have had to use a dissipative coefficient based on the spectral radius for each face to obtain stable coarse grid operators. Another way to ensure a stable coarse grid operator is to formulate a first-order Roe's upwind solver for polyhedral grids,[13] but this was found to be considerably more expensive.[14]

Since the fine grid control volumes comprising a coarse grid control volume are known, the restriction is similar to that used for structured grids. The residuals are simply summed from the fine grid cells and the variables are interpolated in a volume-weighted manner. For the prolongation operator, we use a simple injection (a piecewise-constant interpolation). This is an unfortunate but an unavoidable consequence of using the agglomeration strategy. A piecewise-linear prolongation operator implies a triangulation, the avoidance of which is the main motivation for the agglomeration. In actuality, the agglomeration procedure itself implies a triangulation. In inspecting Fig. 5 which shows an agglomerated coarse grid, we notice that three dual edges meet at every vertex, such as A or B. Thus, each vertex in this figure represents a triangle made up of the three vertices representing the three coarse grid cells. However, we have found this triangulation to be invalid, in general, with a number of crossing grid lines, especially near the trailing edges of the airfoils. Another possible way to achieve a piecewise-linear interpolation is to utilize the gradient that is computed in a least squares sense by making use of data from some defined neighborhood of the coarse grid cell. We do not follow either of these approaches because of their associated complexity. The adverse impact of the injection is minimized by employing additional smoothing steps. This is achieved by applying an averaging procedure to the injected corrections. In an explicit scheme, solution updates are directly proportional to the computed residuals. Thus, by analogy, for the multigrid scheme, corrections may be smoothed by a procedure previously developed for implicit residual smoothing.[1] The implicit equations for the smoothed corrections are solved using two iterations of a Jacobi scheme after prolongation at each grid level.

The agglomeration procedure can be interpreted as a Galerkin coarse grid approximation.[15] If the fine grid operator can be written as

$$Au = f \tag{2}$$

the Galerkin coarse grid approximation can be derived as

$$\bar{A}\bar{u} = \bar{f} \tag{3}$$

with

$$\bar{A} = RAP \tag{4}$$

and

$$\bar{f} = Rf \tag{5}$$

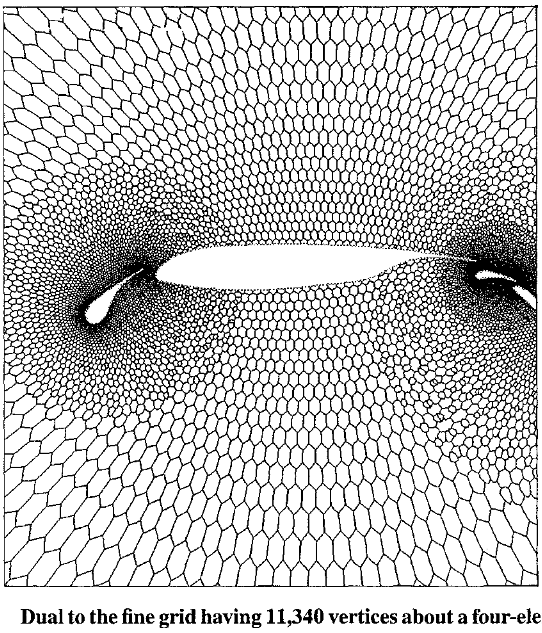

Fig. 6 Dual to the fine grid having 11,340 vertices about a four-element airfoil.

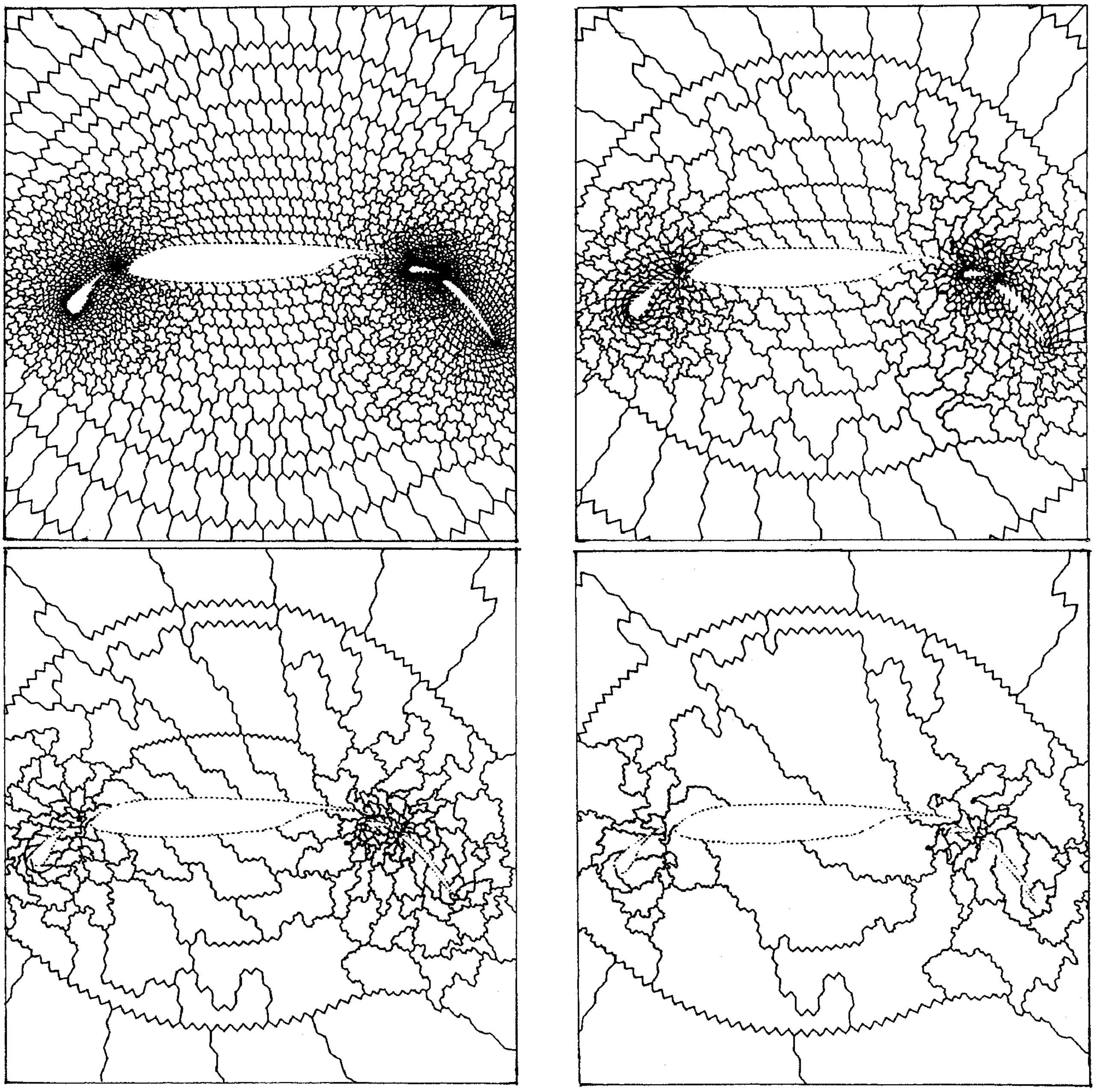

Fig. 7 Four agglomerated coarse grids for the four-element test case.

Here R and P are the restriction and prolongation operators, u is the fine grid vector, and $\bar{u}$ is the coarse grid vector. In the case of agglomeration, both R and P are chosen to be piecewise-constant operators. Note that the scaling of R is immaterial in this formulation. Agglomeration can thus be viewed as identifying all of the fine grid cells with the coarse grid cell to which they belong, and then summing the equations corresponding to these cells. Interpreted in this manner, agglomeration multigrid is similar to algebraic multigrid procedure of Ruge and Stüben[16] except that the equations to be combined are prespecified.

V. Results and Discussion

Representative two- and three-dimensional flows are computed over geometries with varying degrees of geometrical complexity. First, results are presented for a two-dimensional flow problem. The performance of the agglomerated multigrid algorithm is compared with that of the non-nested multigrid algorithm.[1,2] Inviscid flow over a four-element airfoil in landing configuration is computed. The freestream Mach number is 0.2, and the angle of attack is 5 deg. The fine grid has 11,340 vertices, and its dual is shown in Fig. 6. The coarse grids for use with the non-nested multigrid algorithm (not shown) contain 2942 and 727 vertices. It was not

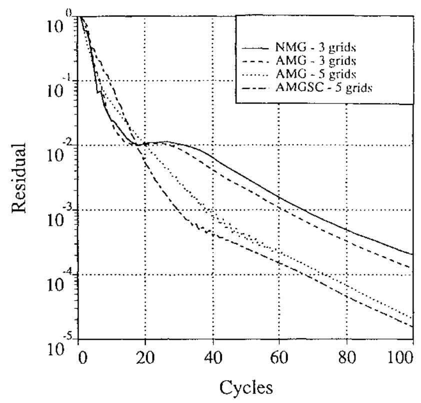

Fig. 8 Convergence histories with the agglomerated and original multigrid: non-nested multigrid (NMG), agglomerated multigrid using the isotropic algorithm (AMG), and agglomerated multigrid using the semicoarsened algorithm (AMGSC).

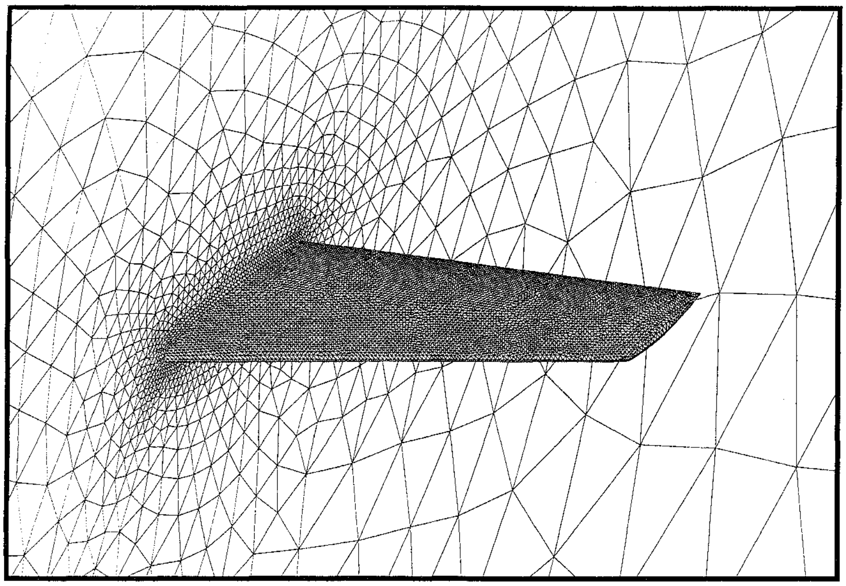

Fig. 9 Surface mesh for a coarse grid with 53,961 vertices about an ONERA M6 wing; fine grid has 357,000 vertices.

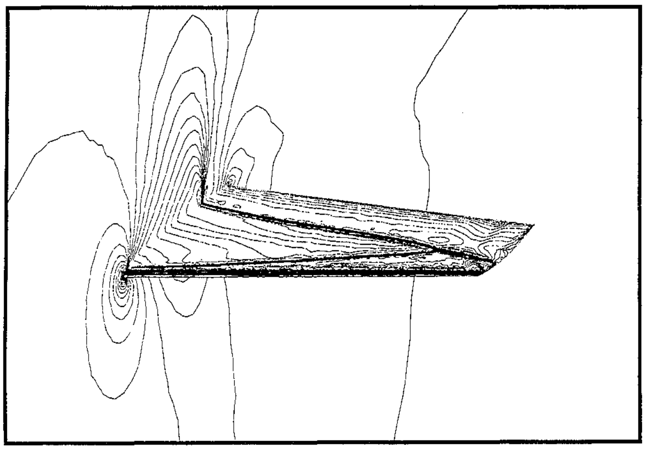

Fig. 10 Mach contours on the surface for transonic flow over the ONERA M6 wing.

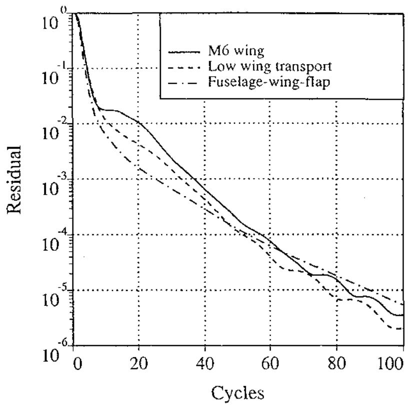

Fig. 11 Convergence histories with agglomeration multigrid: the six-level scheme for the ONERA M6 wing case, the seven-level for the low-wing transport case, and the six-level scheme for the fuselage-wing-flap geometry.

possible to generate coarser grids for this complex geometry. Four agglomerated coarse grids obtained using the isotropic version of the algorithm are shown in Fig. 7. These grids contain 3027, 822, 217, and 63 vertices (regions), respectively. The convergence histories of the non-nested and agglomeration multigrid algorithm are shown in Fig. 8. The convergence histories of the three-level multigrid are comparable but the convergence is slightly better with the agglomerated multigrid strategy. This is a bit surprising since the original multigrid algorithm employs a piecewise-linear prolongation operator. A possible explanation is that the coarse grids created by the agglomeration algorithm are better than those employed in the non-nested algorithm. Convergence improves considerably with the five-level multigrid algorithm. The CPU times required on one processor of Cray Y-MP for the three-level multigrid are 59 and 60 s with the original and the agglomerated algorithms, respectively. The CPU time required for the five level agglomeration multigrid is 73 s. The convergence rate per work unit with the three-level non-nested multigrid algorithm is 0.940. The convergence rates per work unit with the three-level and five-level agglomerated multigrid algorithm are 0.938 and 0.920, respectively. We also observe in Fig. 8 that there is only a slight difference in convergence rates between the two versions of the agglomeration algorithm. The convergence rates per multigrid cycle with the isotropic and semicoarsened versions of the agglomeration algorithm are 0.898 and 0.894, respectively, for the five-level multigrid scheme. Similar behavior has been observed in most of our two- and three-dimensional computations. This is not surprising since the aspect ratios of the cells for these Euler grids are quite benign. The payoff, we believe, will be in the computation of viscous flows, where cell aspect ratios of 1000 are typical. Nevertheless, in the rest of the section, all of the coarse grids have been derived using the semicoarsened version of the algorithm.

We next present results from three-dimensional calculations of flows over geometries of increasing complexity. The first flow considered is transonic flow over an ONERA M6 wing at a freestream Mach number of 0.84 and an angle of attack of 3.06 deg. The geometry consists of the wing delimited by a symmetry plane. The mesh was generated using the advancing-front grid generation technique.[17] The grid contains 357,900 vertices and just over 2×10^6 tetrahedra. However, due to limitations in printing resolution, we only display in Fig. 9 the surface mesh corresponding to a coarser grid with 53,961 vertices. The semicoarsened agglomeration algorithm is used to generate five coarse grids. These coarse grids contain 62,594, 13,033, 3,105, 804, and 54 vertices, respectively. Figure 10 shows the Mach contours on the surface where the familiar double shock pattern for this standard test case may be observed. The convergence rate of 0.878 per cycle is achieved which is comparable to the rate of 0.871 achieved by Mavriplis[2] with four-level non-nested full multigrid procedure for the same case. The final solution is identical to that obtained by Mavriplis[2] since the same fine mesh discretization is employed. This particular

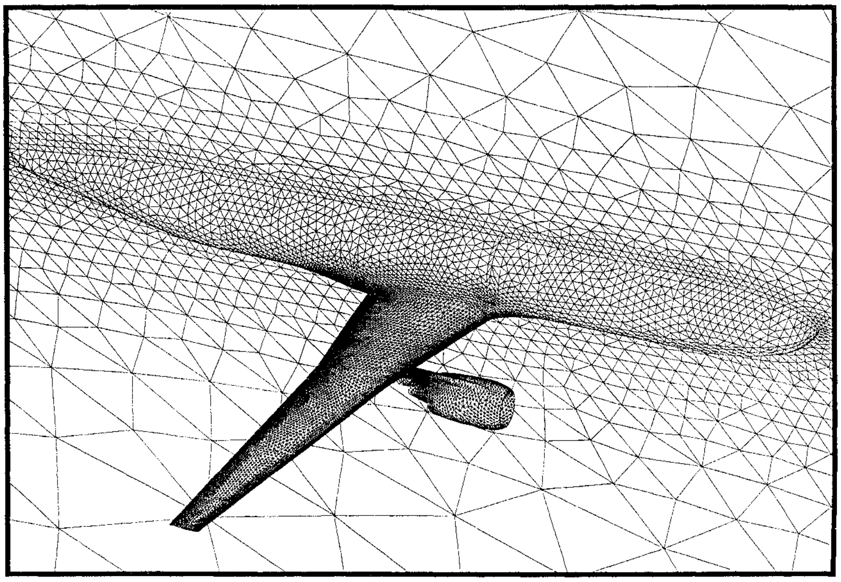

Fig. 12 Surface mesh for a coarse grid with 106,064 vertices about a low-wing transport configuration; fine mesh has 804,056 vertices.

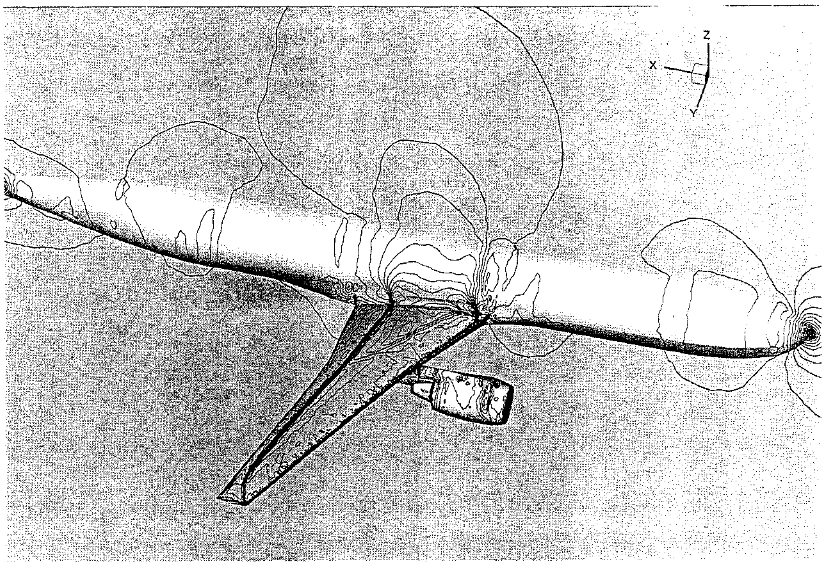

Fig. 13 Mach contours on the surface for transonic flow over the low-wing transport configuration.

solution has also been shown to be in good agreement with other inviscid computations for this case.[2] The convergence history for the six-level multigrid using the five agglomerated coarse grids is shown in Fig. 11. The residuals are reduced by nearly six orders of magnitude in 100 W-cycles, which required 21 min of CPU time and 44 MW of memory on one processor of Cray C-90. By comparison, the single grid run resulted in a residual reduction of merely three orders of magnitude in 400 cycles. A complete multigrid cycle requires about 80% more CPU time than the driving single grid cycle and a 30% increase in memory requirements.

The next case involves the solution of inviscid transonic flow over a low-wing transport configuration. The geometry consists of a half-fuselage bounded by a symmetry plane with a wing and a nacelle. The fine mesh for this case contains 804,056 vertices and approximately 4.5×10^6 tetrahedra. Again, due to limitations in printing resolution, only the surface mesh for a coarser mesh with 106,064 vertices is depicted in Fig. 12. This computation represents one of the largest unstructured grid cases attempted to date. However, this is believed to be representative of the type of resolution required for engineering calculations for configurations of this type. Six coarse grids are derived by the semicoarsened agglomeration algorithm and contain 132,865, 25,489, 5,678, 1,421, 373, and 99 vertices, respectively. The freestream conditions are Mach number of 0.77 and 1.116-deg incidence. The convergence history using a seven-level agglomerated multigrid strategy is also shown in Fig. 11. The residuals are reduced by nearly six orders of magnitude in 100 W-cycles, which corresponds to a convergence rate of 0.876 per cycle. This compares favorably with the rate of 0.870 obtained using a five-level non-nested multigrid strategy for the same case. The agglomerated multigrid computation required 96 MW of memory and 45 min of CPU time on one processor of Cray C-90. The computed Mach contours are depicted in Fig. 13 where the upper surface wing shock pattern is evident.

Finally, we consider flow over a fuselage-wing-flap geometry. This case, which has the additional complexity of a flap system, is representative of a simplified Boeing 737 landing configuration geometry. The fine grid contains 134,539 vertices, and the surface mesh is shown in Fig. 14. This mesh was provided by the author of Ref. 18. Although the grid is too coarse to provide accurate engineering results for this geometry, this case has been included to illustrate the power of the agglomeration technique in handling complex geometries. The small gap sizes between neighboring flap elements restrict the effectiveness of traditional multigrid methods. In the non-nested mesh approach, for example, the coarsest

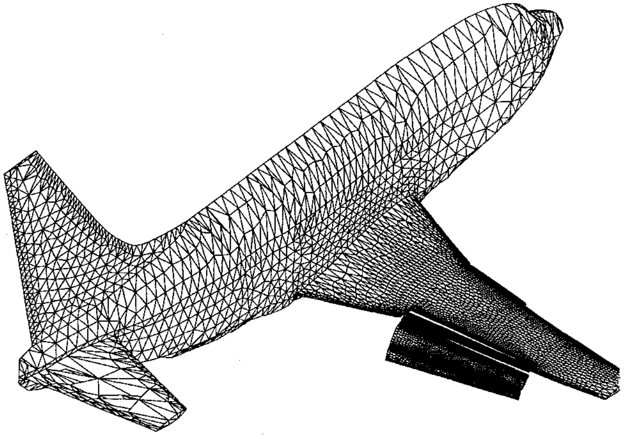

Fig. 14 Surface mesh for a grid with 134,539 vertices about a fuselage-wing-flap geometry.

meshes must contain cells small enough to resolve these gaps, or risk altering the geometry. For the agglomeration approach, on the other hand, these restrictions are completely removed. We have constructed five coarse agglomerated grids of 22,238, 4,043, 810, 205, and 52 vertices. The topology of the geometry is allowed to change on these coarse grids; the interpretation of agglomeration as an algebraic technique [cf. Eq. (3)] guarantees the validity of the procedure in this case. For a freestream Mach number of 0.4 and 0-deg incidence, the six-level agglomeration strategy results in a residual reduction of 5.5 orders of magnitude in 100 W-cycles for a convergence rate of 0.885 per cycle. This rate is consistent with the rates obtained in the previous cases. The convergence history for this case is displayed in Fig. 11 as well. This computation was done in slightly under 10 min on a single processor of Cray C-90. It should be noted that engineering accuracy for the cases shown in this paper can be achieved in fewer than 50 multigrid cycles.

Conclusions

It has been shown that the agglomeration multigrid strategy can be made to approximate the efficiency of the unstructured multigrid algorithm using independent, non-nested coarse meshes, both in terms of convergence rates and CPU times. It is further shown that arbitrarily coarse grids can be obtained with the agglomeration technique, although care must betaken to ensure that the coarse grid operator is convergent on these grids. The agglomeration itself is done very efficiently as a preprocessing step. It has been shown that the agglomeration algorithm can be tailored to achieve desired coarse grid properties. Applications to flows over complex configurations have been provided to illustrate the power and flexibility of the agglomeration multigrid strategy.

Acknowledgments

This work was supported, in part, under NASA Contract NAS2-12961 during the first author's employment at Computer Science Corporation and, in part, by NASA Contract NAS1-19480 during both authors' residence at the Institute for Computer Applications in Science and Engineering.

References

[1] Mavriplis, D. J., and Jameson, A., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes," *AIAA Journal*, Vol. 26, No. 7, 1988, pp. 824–831.

[2] Mavriplis, D. J., "Three Dimensional Multigrid for the Euler Equations," AIAA Paper 91-1549, July 1991.

[3] Leclercq, M. P., "Resolution des Equations d'Euler par des Methodes Multigrilles Conditions aux Limites en Regime Hypersonique," Ph.D Thesis, Applied Math, Université de Saint-Etienne, France, April 1990.

[4] Peraire, J., Peiro, J., and Morgan, K., "A 3D Finite-Element Multigrid Solver for the Euler Equations," AIAA Paper 92-0449, Jan. 1992.

[5] Perez, E., "A 3D Finite-Element Multigrid Solver for the Euler Equations," INRIA Rep. 442, Sept. 1985.

[6] Connell, S. D., and Holmes, D. G., "A 3D Unstructured Adaptive Multigrid Scheme for the Euler Equations," *AIAA Journal*, Vol. 32, No. 8, 1994, pp. 1626–1632.

[7] Guillard, H., "Node Nested Multigrid with Delaunay Coarsening," INRIA Rep. 1898, 1993.

[8] Lallemand, M., Steve, H., and Dervieux, A., "Unstructured Multigridding by Volume Agglomeration: Current Status," *Computers and Fluids*, Vol. 21, No. 3, 1992, pp. 397–433.

[9] Smith, W. A., "Multigrid Solution of Transonic Flow on Unstructured Grids," *Recent Advances and Applications in Computational Fluid Dynamics*, Proceedings of the American Society of Mechanical Engineers Winter Annual Meeting, edited by O. Baysal, Nov. 1990.

[10] Spragle, G., Smith, W. A., and Yadlin, Y., "Application of an Unstructured Flow Solver to Planes, Trains and Automobiles," AIAA Paper 93-0889, Jan. 1993.

[11] Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods using Multi-Stage Time Stepping Schemes," AIAA Paper 81-1259, 1981.

[12] Roe, P. L., "Characteristic-based Schemes for the Euler Equations," *Annual Review of Fluid Mechanics,* Vol. 18, 1986, pp. 337–365.

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

[14] Venkatakrishnan, V., Mavriplis, D. J., and Berger, M. J., "Unstructured Multigrid through Agglomeration," Proceedings of the 6th Copper Mountain Multigrid Conf., April 1993.

[15] Wesseling, P., "An Introduction to Multigrid Methods," *John Wiley*, New York, 1992.

[16] Ruge, J. W., and Stüben, K., "Algebraic Multigrid," *Multigrid Methods*, edited by S. F. McCormick, SIAM Frontiers in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987, pp. 73–131.

[17] Parikh, P., Pirzadeh, S., and Lohner, R.,"A Package for 3-D Unstructured Grid Generation, Finite-element Solution and Flow-visualization," NASA CR-182090, Sept. 1990.

[18] Dodbele, S. S., "Three-dimensional Aerodynamic Analysis of a High-lift Transport Configuration," AIAA Paper 93-3536, Aug. 1993.