UNSTRUCTURED MESH ALGORITHMS FOR AERODYNAMIC CALCULATIONS

D. J. Mavriplis

Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA

1. ABSTRACT

The use of unstructured mesh techniques for solving complex aerodynamic flows is discussed. The principle advantages of unstructured mesh strategies, as they relate to complex geometries, adaptive meshing capabilities, and parallel processing are emphasized. The various aspects required for the efficient and accurate solution of aerodynamic flows are addressed. These include mesh generation, mesh adaptivity, solution algorithms, convergence acceleration and turbulence modeling. Computations of viscous turbulent two-dimensional flows and inviscid three-dimensional flows about complex configurations are demonstrated. Remaining obstacles and directions for future research are also outlined.

2. INTRODUCTION

Over the last decade, much attention has been devoted to the development and use of unstructured mesh methodologies within the research community. This enthusiasm however, has not always been shared by the applications and industrial community. The promise of easily enabling the discretization of complex geometries has been counterbalanced by questions of accuracy and efficiency. Furthermore, the dearth of results concerning viscous flow calculations using unstructured meshes has produced skepticism concerning the value of unstructured mesh techniques for practical aerodynamic calculations.

There is no doubt that block-structured techniques have proved extremely effective in discretizing very complex geometries. However, unstructured grid techniques offer additional inherent advantages which may not at first appear evident. The possibility of easily performing adaptive meshing is perhaps the largest advantage of unstructured grid methods. In fact, the implementation of adaptive meshing techniques for structured meshes has generally been found to incur unstructured-mesh type overheads [1]. Furthermore, although unstructured grid data-sets are irregular, they are homogeneous (as opposed to block structured grids where differentiation between block boundaries and interiors must be made). One of the consequences of this property is that unstructured-mesh type solvers are relatively easily parallelizable. While unstructured mesh solvers always incur additional memory and CPU-time overheads due to the random nature of their data-sets, large gains in efficiency can be obtained by careful choices of data-structures, and by resorting to more efficient implicit or multi-level solution procedures. When combined with adaptive meshing and parallelization, these can result in truly competitive solution procedures.

In the following sections, the application of unstructured mesh techniques to various aerodynamic flow problems are discussed. The particular approach chosen (i.e. a vertex based Galerkin finite-element discretization with additional artificial dissipation terms and an unstructured multigrid algorithm for convergence acceleration), represents the methodology adopted over several years of research by the author, and constitutes but one of several competing approaches. Both inviscid and viscous flows are considered, although exclusively steady-state solution procedures are discussed. Both two and three-dimensional problems are addressed.

3. SOLUTION PROCEDURE

In non-dimensional conservative vector form, the Navier-Stokes equations read

$$\frac{\partial w}{\partial t} + \nabla . \mathbf{F}_c = \frac{1}{\mathrm{Re}_\infty} \nabla . \mathbf{F}_v \qquad (1)$$

where Re_∞ denotes the overall flow Reynolds number, and w represents the conserved variables

$$w = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix} \qquad (2)$$

ρ being the fluid density, u, v ,and w the cartesian velocity components, and E the internal energy. $\mathbf{F}_c$ represents the convective flux vector, the components of which are algebraic functions of the conserved variables and the pressure, which itself can be related to the conserved variables through the perfect gas relation. $\mathbf{F}_v$ denotes the viscous flux vector, the components of which are functions of the first derivatives of the conserved variables. Equation (1) represents a set of partial differential equations which must be discretized in space in order to obtain a set of coupled ordinary differential equations, which can then be integrated in time to obtain the steady-state solution. Spacial discretization is performed using a Galerkin finite-element type formulation. The following derivation is restricted to the two-dimensional case for the sake of clarity, since the extension from two-dimensions to three-dimensions is entirely straight-forward. Multiplying equation (1) by a test function ϕ, and integrating over physical space yields

$$\frac{\partial}{\partial t}\iint_\Omega \phi w \, dxdy + \iint_\Omega \phi \nabla . \mathbf{F}_c \, dxdy = \frac{1}{\mathrm{Re}_\infty}\iint_\Omega \phi \nabla . \mathbf{F}_v \, dxdy \qquad (3)$$

Integrating the flux integrals by parts, and neglecting boundary terms gives

$$\frac{\partial}{\partial t}\iint_\Omega \phi w \, dxdy = \iint_\Omega \mathbf{F}_c . \nabla \phi \, dxdy - \frac{1}{\mathrm{Re}_\infty}\iint_\Omega \mathbf{F}_v . \nabla \phi \, dxdy \qquad (4)$$

In order to evaluate the flux balance equations at a vertex P, ϕ is taken as a piecewise linear function which has the value 1 at node P, and vanishes at all other vertices. Therefore, the integrals in the above equation are non-zero only over triangles (tetrahedra in three dimensions) which contain the vertex P, thus defining the domain of influence of node P, as shown in Figure 1, for the two-dimensional case. To evaluate the above integrals, we make use of the fact that ϕ_x and ϕ_y are constant over a triangle, and evaluate spatial derivatives of ϕ and w over a triangle using vertex values, by Green's contour integral theorem. The convective fluxes $\mathbf{F}_c$ are taken as piecewise linear functions in space, and the viscous fluxes $\mathbf{F}_v$ are piecewise constant over each triangle, since they are formed from first derivatives in the flow variables. Evaluating the flux integrals with these assumptions, one obtains

$$\frac{\partial}{\partial t}\iint_\Omega \phi w \, dxdy = \sum_{s=1}^{n} \frac{\mathbf{F}_c^A + \mathbf{F}_c^B}{6} . \Delta \mathbf{L}_{AB} - \frac{1}{\mathrm{Re}_\infty} \sum_{s=1}^{n} \frac{\mathbf{F}_v^s}{2} . \Delta \mathbf{L}_{AB} \qquad (5)$$

where the summations are over all triangles in the domain of influence, as shown in Figure 1. Δ_{AB} represents the directed (normal) edge length of the face of each triangle on the outer boundary of the domain, $\mathbf{F}_c^A$ $\mathbf{F}_c^B$ are the convective fluxes at the two vertices at either end of this edge, and $\mathbf{F}_v^e$ is the viscous flux in triangle e, e being a triangle in the domain of influence of ϕ. If the integral on the left hand side of equation (5) is evaluated in the same manner, the time derivatives become coupled in space. Since we are not interested in the time-accuracy of the scheme, but only in the final steady-state solution, we employ the concept of a lumped mass matrix. This is equivalent to assuming w to be constant over the domain of influence while integrating the left hand side. Hence, we obtain

$$\Omega_p \frac{\partial w_p}{\partial t} = \sum_{e=1}^{n} \frac{\mathbf{F}_c^A + \mathbf{F}_c^B}{2} \cdot \Delta \mathbf{L}_{AB} - \frac{1}{\mathrm{Re}_\infty} \sum_{e=1}^{n} \frac{3}{2} (\mathbf{F}_v^e \cdot \Delta \mathbf{L}_{AB}) \quad (6)$$

where the factor of 1/3 is introduced by the integration of ϕ over the domain, and Ω_p represents the surface area of the domain of influence of P. For the convective fluxes, this procedure is equivalent to the vertex finite-volume formulation described in [2,3]. For a smoothly varying regular triangulation, the above formulation is second-order accurate.

Additional artificial dissipation terms are required to ensure stability and to capture shocks without producing numerical oscillations. This is necessary for both inviscid and viscous flow computations, since in the later case, large regions of the flow-field behave essentially inviscidly and the physical viscosity is not sufficient to guarantee numerical stability for the type of mesh spacings typically employed. Artificial dissipation terms are thus constructed as a blend of a Laplacian and a biharmonic operator in the conserved flow variables. The Laplacian term represents a strong formally first-order accurate dissipation which is turned on only in the vicinity of a shock, and the biharmonic term represents a weaker second-order accurate dissipation which is employed in regions of smooth flow [4,5,6].

The spatially discretized equations are integrated in time to obtain the steady-state solution. For inviscid flow calculations, a five-stage Runge-Kutta scheme is employed for the time integration, where the convective terms are evaluated at each stage in the time-stepping scheme, and the dissipative terms are only evaluated at the first two stages and then frozen for the remaining stages. A complete multistage time-step, in which the solution is advanced from time level n to level n+1, can be written as

$$\begin{aligned}
w^{(0)} &= w^n \\
w^{(1)} &= w^{(0)} - \alpha_1 \Delta t \left[Q(w^{(0)}) - D(w^{(0)}) \right] \\
w^{(2)} &= w^{(0)} - \alpha_2 \Delta t \left[Q(w^{(1)}) - D(w^{(1)}) \right] \\
w^{(3)} &= w^{(0)} - \alpha_3 \Delta t \left[Q(w^{(2)}) - D(w^{(1)}) \right] \\
w^{(4)} &= w^{(0)} - \alpha_4 \Delta t \left[Q(w^{(3)}) - D(w^{(1)}) \right] \\
w^{(5)} &= w^{(0)} - \alpha_5 \Delta t \left[Q(w^{(4)}) - D(w^{(1)}) \right] \\
w^{(n+1)} &= w^{(5)}
\end{aligned} \quad (7)$$

with

$$\alpha_1 = 1/4 \qquad \alpha_2 = 1/6 \qquad \alpha_3 = 3/8 \qquad \alpha_4 = 1/2 \qquad \alpha_5 = 1$$

where w represents the conserved flow variables, Q is the convective residual, D denotes the dissipative operator, and Δt represents the discrete time-step. For viscous flow computations, a variant of this scheme is employed, where the dissipative terms are evaluated at the first, third and fifth stages, and frozen at alternate stages. These particular schemes have been designed to rapidly damp out high frequency error components [4,5], which is a necessary characteristic for a multigrid driving scheme. Convergence to steady-state is accelerated by employing local time-stepping and implicit residual averaging [2,3,4], which have previously been described in the context of unstructured meshes.

4. MULTIGRID STRATEGY

The idea of a multigrid strategy is to perform time steps on coarser meshes to calculate corrections to a solution on a fine mesh. The advantages of time stepping on coarse meshes are twofold: first, the permissible time-step is much larger, since it is proportional to the cell size, and secondly, the work is much less because of the smaller number of grid points. On the finest grid of the sequence, the flow variables are updated by the 5-stage scheme as shown in equations (7). The residuals and flow variables are then transferred to the next coarser grid. If R' represents the transferred residuals and w' the transferred flow variables, a forcing function on the coarse grid can be defined as

$$P = R' - R(w') \quad (8)$$

Now on the coarse grid, time stepping proceeds as shown below:

$$w^{(q)} = w^{(q-1)} - \alpha_q \Delta t \left(R(w^{(q-1)}) + P \right) \quad (9)$$

for the q-th stage. In the first stage, $w^{(q-1)}$ reduces to the transferred flow variable w'. Thus, the calculated residuals on the coarse grid are canceled by the second term in the forcing function P, leaving only the R' term. This indicates that the driving force for the solution on the coarse grid is provided by the fine grid residuals. Thus we are ensured that, when the fine grid solution is fully converged, no further corrections will be generated by the coarser grids. This procedure is repeated on successively coarser grids. When the coarsest grid is reached, the corrections are transferred back to the finer grids. The use of a multigrid method with unstructured meshes presents an additional challenge. Consistent coarse tetrahedral grids can no longer be formed by simply considering subsets of the fine grid vertices. An alternative would be to generate the fine mesh by repeatedly subdividing an initial coarse mesh in some manner. However, generally poor topological control of the fine mesh results from such a procedure. Another approach, known as the agglomeration technique, reconstructs coarse grids from a given fine unstructured grid by grouping neighboring elements together to form large polyhedral coarse-grid cells [7,8]. In the present work, it has been decided to pursue an unstructured multigrid approach in which a sequence of completely unrelated coarse and fine meshes are employed. This approach provides great flexibility in determining the configuration of the coarsest and finest meshes. Coarse meshes may be designed to optimize the speed of convergence, whereas fine meshes may be constructed based on solution accuracy considerations. In general, beginning from a fine grid, a coarser level is constructed which contains roughly half the resolution in each coordinate direction throughout the domain (about 1/8 the number of vertices in three dimensions, or 1/4 in two dimensions). This process is repeated until the coarsest grid capable of representing the geometry topology is obtained. In the context of adaptive meshing, new finer meshes may be added to the multigrid sequence, using any given adaptive refinement technique, since no relation is assumed between the various meshes of the sequence.

The key to the success of such a strategy lies in the ability to efficiently transfer variables, residuals and corrections back and forth between unrelated unstructured meshes. In the present context, this is performed using linear interpolation. For each vertex of a given grid, the tetrahedron which contains this vertex on the grid to which variables are to be interpolated is determined. The variable at this node is then linearly distributed to the four vertices of the enclosing tetrahedron (three vertices of the enclosing triangle in two dimensions). The main difficulty lies in efficiently determining the enclosing cell for each grid point. A naive search over all cells would lead to an $O(N^2)$ complexity algorithm, where N is the total number of grid points, and would be more expensive than

the flow solution itself. In this work, a graph traversal search routine with best case complexity of $O(N)$ is employed. The search begins by choosing a node on one grid, and locating the enclosing tetrahedron on the other grid. This can usually be determined a priori, for example, by choosing the minimum x-y-z node and the minimum x-y-z tetrahedron for the respective grids. We next chose a new node for which the enclosing cell is to be searched, and this node is taken as a neighbor of the previous node. As a starting guess we choose the tetrahedron which was previously found to enclose the first node, which is in the same vicinity as the new node. If this cell is not found to enclose the new node, we search the four neighbors of this cell, and then the neighbors of these neighbors, thus traversing through the mesh until the enclosing cell is located, at which point the process is repeated for a new node.

The interpolation patterns between the various meshes are completely determined by assigning to each mesh vertex four interpolation addresses and four interpolation weights, which are all computed in a preprocessing phase. In practice, this preprocessing has been found to require an amount of CPU time roughly equivalent to one or two flow solution cycles on the finest grid.

5. ADAPTIVITY

One of the most efficient adaptive mesh enrichment techniques consists of sequential point insertion and local grid restructuring. This can be achieved using Bowyer's algorithm for Delaunay triangulation. A Delaunay triangulation is a unique triangulation (tetrahedrization in 3-D) of a given set of points which exhibits certain desirable properties (maximizes small angles, provides a discrete maximum principle for Laplaces equation [9] etc ...). One of these properties, the empty circumcircle property, states that no vertex from any other triangle/tetrahedron can be contained in the circumcircle/sphere of a given triangle/tetrahedron. This property has often been employed as the basis for an algorithm known as Bowyer's method for the generation of unstructured meshes [10,11]. Bowyer's algorithm is also useful for adaptive mesh refinement. Assuming we have discretized the geometry with a Delaunay triangulation/tetrahedrization, and have solved the flow on this grid, we seek to refine the mesh in regions of high local truncation error. The first undivided differences of some key flow variable (density for example) are examined along every edge of the mesh. When this difference is larger than some fraction of the average differences across all edges of the mesh, a new point is added midway along that edge. Each new point must be inserted into the mesh, which must then be locally restructured accordingly. Following Bowyer's algorithm, we first locate all triangles/tetrahedra whose circumcircles/spheres are intersected by this new point. The union of these cells are removed, as this determines the region of the mesh which must be restructured. A new structure is then formed by joining the new point to all vertices of the polygonal/hedral cavity formed by the cell removal operation, as shown for the two-dimensional case in Figure 2. This has been proven to result in a consistent Delaunay triangulation provided the original mesh is a Delaunay construction [11]. In cases where a non-Delaunay triangulation is employed for the original mesh, a consistency check must be executed after each new point is inserted. If negative volume cells are created, the new point must either be rejected or displaced and reinserted [12]. When new boundary points are introduced, they are repositioned onto the analytic surface-patch definition (or spline curve definition in two dimensions) of the geometry by recomputing the physical coordinates of the new point based on the assigned parametric patch coordinates, s and t, which are taken as the average of the parametric coordinates of the two vertices at either end of the generating boundary edge.

6. TWO-DIMENSIONAL RESULTS

6.1. An Inviscid Case

In order to illustrate the effectiveness of the simultaneous use of adaptive meshing and the multigrid strategy, the inviscid flow through a two-dimensional turbine blade cascade geometry has been computed. The particular blade geometry has been the subject of an experimental and computational investigation at the occasion of a VKI lecture series [13]. A total of seven meshes were used in the multigrid algorithm, with the last three meshes generated adaptively, using the undivided density difference criterion. The coarsest mesh of the sequence contains only 51 points, while the finest mesh, depicted in Figure 3, contains 9362 points. Extensive mesh refinement can be seen to occur in the neighborhood of shocks, and in other regions of high gradients. The inlet flow incidence is 30 degrees, and the average inlet Mach number is 0.27. The flow is turned 96 degrees by the blades, and the average exit isentropic Mach number is 1.3. At these conditions, the flow becomes supersonic as it passes through the cascade, and a complex oblique shock wave pattern is formed. These are evident from the computed Mach contours depicted in Figure 4. All shocks are well resolved, including some of the weaker reflected shocks, which non-adapted mesh computations often have difficulty resolving. Details of the flow in the rounded trailing edge region of the blade, where the flow separates inviscidly and forms a small recirculation region, are also well reproduced. Once the first four globally generated meshes were constructed, the entire flow solution - adaptive mesh enrichment cycle was performed three times, executing 25 multigrid cycles at each stage. This entire operation required 40 CPU seconds on a single processor of a Cray-YMP supercomputer. The residuals on the finest mesh were reduced by two and a half orders of magnitude, which should be adequate for engineering calculations.

6.2. Viscous Flows

While the discretization of the viscous terms for the Navier-Stokes equations as outlined in Section 2 is relatively straightforward, the main difficulties involved in computing high-Reynolds-number viscous flows relate to the grid generation and turbulence modeling requirements. In order to efficiently resolve the thin viscous layers encountered in such flows, highly stretched grids with very high resolution in the direction normal to the flow must be employed. Standard unstructured grid generation techniques (i.e. advancing front methods [14,15], or Delaunay triangulations [11,16]) generally break down when attempting to generate such highly stretched grids (normal to streamwise resolution ratios of 100 to 1000 are typically required). The procedure adopted in this work is to employ one of these standard techniques (in this case, the Delaunay construction) in a locally mapped space, as opposed to physical space [17]. A suitable mesh-point distribution with the required normal and streamwise resolution must first be obtained. This is achieved by generating a structured hyperbolic mesh about each geometry component, and employing the union of the points of these overlapping local structured meshes as the basis of a Delaunay triangulation. However, a Delaunay triangulation of a given set of points tends to produce the most equiangular triangles possible, and therefore in general, is not well suited for the generation of highly stretched mesh elements. Thus, an alternate triangulation procedure must be employed. The approach taken consists of defining a stretching vector (stretching magnitude and direction) at each node of the initial point distribution throughout the flow field. Assuming an initial triangulation has been obtained, when a new mesh point is to be inserted, the associated stretching vector is employed to construct a locally mapped space such that,

within this mapped space, the local point distribution appears isotropic. A Delaunay triangulation is then performed to triangulate the new point into the mesh in this mapped space, and the resulting triangulation is mapped back into physical space, thus resulting in the desired stretched triangulation. Hence, a fully unstructured mesh with highly stretched elements in the boundary layer and wake regions, nearly equilateral triangles in the inviscid regions of flow, and a smooth variation of elements throughout the transition regions is obtained. The use of fully unstructured meshes for viscous flow calculations has been pursued, as opposed to the hybrid structured-unstructured meshes often advocated in the literature [18,19], due to the increased generality they afford in dealing with geometries with close tolerances between neighboring bodies, where confluent boundary layers may occur, and due to the ease with which adaptive meshing may be incorporated throughout the viscous and inviscid regions of flow.

The use of a turbulence model is required for the practical solution of high-Reynolds number viscous flows. The most common turbulence models employed for aerodynamic flows are of the algebraic type. Such models typically require information concerning the distance of each point from the wall. Turbulence length scales are determined by scanning appropriate flow variables along specified streamwise stations. In the context of unstructured meshes, such information is not readily available and hence, the implementation of algebraic turbulence models on such meshes introduces additional complexities. The approach adopted in this work [20] consists of generating a set of background turbulence mesh stations. These are constructed by generating a hyperbolic structured mesh about each geometry component, based on the boundary-point distribution of the original unstructured mesh, and extracting the normal lines of the mesh. When performing adaptive meshing, new turbulence mesh stations must be constructed for each new adaptively generated boundary point, as illustrated in Figure 5. Each time the turbulence model is executed, the flow variables are interpolated onto the normal turbulence stations, the turbulence model is executed on each station, and the resulting eddy viscosity is interpolated back to the unstructured mesh. The method employed for interpolating variables back and forth between the unstructured mesh and the turbulence mesh stations is similar to that previously described for the unstructured multigrid algorithm.

Figures 6 through 9 illustrate a calculation which makes use of these various techniques to compute a complicated two-dimensional viscous flow over a high-lift multi-element airfoil. The final mesh employed is depicted in Figure 6, and contains a total of 48,691 points. This mesh was obtained using the stretched Delaunay triangulation technique previously described, followed by two levels of adaptive refinement. The height of the smallest cells at the wall is of the order of 2×10^{-5} chords and cell aspect ratios up to 500:1 are observed. The computed Mach number contours for this case are depicted in Figure 7. The freestream Mach number is 0.1995, the chord Reynolds number is 1.187 million, and the corrected incidence is 16.02 degrees. At these conditions, the flow remains entirely subcritical. Compressibility effects are nevertheless important due to the large suction peaks generated about each airfoil. For example, in the suction peak on the upper surface of the leading-edge slat, the local Mach number achieves a value of 0.77. The computed surface pressure coefficients are compared with experimental wind tunnel data [21] in Figure 8, and good overall agreement is observed, including the prediction of the height of the suction peaks. This case provides a good illustration of the importance of adaptive meshing in practical aerodynamic calculations. Adequate resolution of the strong suction peak on the upper surface of the slat can only be achieved with a very fine mesh resolution in this region. Failure to adequately capture this large suction peak results in the generation of numerical entropy which is then convected downstream, thus contaminating the solution in the downstream regions, and degenerating the global accuracy of the solution. Because these suction peaks are very localized, they are efficiently resolved with adaptive techniques. In order to obtain a similar resolution using global mesh refinement, of the order of 200,000 mesh points would be required, greatly increasing the cost of the computation. The convergence history for this case, as measured by the density residuals and the total lift coefficient versus the number of multigrid cycles, is depicted in Figure 9. A total of 400 multigrid cycles were executed, which required roughly 35 minutes of single processor CRAY-YMP time, and 14 Mwords of memory.

The discrepancy between the computed and experimental pressure coefficients on the trailing edge flap is due to a separated flow condition which is not reproduced by the algebraic turbulence model. Figure 10 compares computed and experimental lift coefficients at various angles of attack for a three-element high-lift airfoil [22]. The failure of the computations to predict the maximum lift point are directly attributable to the inability of the turbulence model to predict the onset of separation. These results strongly indicate the need for more sophisticated turbulence modeling. The use of single or multiple field-equation models appears to be the most appropriate choice for turbulent unstructured mesh computations. Such models can be discretized in a straight-forward manner on unstructured meshes. However, the task is now to ensure that such models adequately represent the flow physics, and that they can be solved in an efficient and robust manner. In this work, the implementation of a standard high-Reynolds-number $k-\varepsilon$ turbulence model with low-Reynolds-number modifications proposed by Speziale, Abid and Anderson [23], has been pursued. The main effort was focused on devising a technique for efficiently solving the two turbulence equations in the context of the unstructured multigrid strategy [24]. The four flow equations and the two turbulence equations are solved as a loosely coupled system. The flow equations are solved explicitly, and the turbulence equations point-implicitly, using a time-step limit which ensures stability and positivity of k and ε. In the context of the unstructured multigrid algorithm, the turbulence eddy viscosity is assumed constant on all but the finest grid level where it is recomputed at each time-step. The transonic flow over a two-element airfoil configuration has been computed using this implementation of the model. For this case, the freestream Mach number is 0.5, the incidence is 7.5 degrees, and the Reynolds number is 4.5 million. Figures 11 and 12 depict the mesh and the solution obtained with the current implementation of the $k-\varepsilon$ turbulence model. Four meshes were employed in the multigrid sequence, with the finest mesh containing a total of 28,871 points. The convergence rates of the various equations for this case are plotted in Figure 13. As can be seen, the turbulence equations and flow equations converge at approximately the same rates. The computed flow field exhibits regions of transonic flow with a small region of separated flow at the foot of the shock. These features appear to be well reproduced by the turbulence model. Future efforts will concentrate on computationally predicting flows with large regions of separation, such as that inferred by Figure 8, and on modifying the model to better represent the flow physics.

7. THREE DIMENSIONAL RESULTS

Due to the limitations of present day supercomputers, and the difficulties associated with generating highly stretched tetrahedral meshes, three-dimensional computations have presently been confined to inviscid flows. The techniques described in the context of two-dimensional inviscid flows extend readily to three dimensions. In particular, the unstructured multigrid algorithm and the adaptive meshing strategy have been found to be particular effective for three-dimensional computations [12]. As an example, an adaptive multigrid calculation of transonic flow about an ONERA M6 wing is illustrated in Figures 14 through 16. The

final mesh, depicted in Figure 14, contains a total of 174,412 points and just over 1 million tetrahedral volumes. This represents the fourth mesh in the multigrid sequence and the second adaptive refinement level. Mesh refinement was based on the undivided gradient of density. The freestream Mach number and incidence for this case are 0.84 and 3.06 degrees respectively. The well known double shock pattern for this case is reproduced in the computed Mach contours of the solution in Figure 15. The leading edge expansion and shocks are well resolved due to the extensive mesh refinement in these regions. A globally refined mesh of this resolution would result in roughly 600,000 points and would thus require 3 to 4 times more computational resources. The multigrid convergence rate for this case is depicted in Figure 16, where 50 cycles were performed on the original grid, prior to adaptation, 50 cycles on the first adapted mesh, and 100 cycles on the finest adapted mesh. On this final mesh, the residuals were reduced by 5 orders of magnitude over 100 cycles, requiring a total of 35 CRAY-YMP single CPU minutes and 22 MW of memory.

7.1. Parallel Computing Results

As mentioned previously, due to their homogeneous (although random) nature, unstructured mesh data-sets are particularly well suited for parallel processing. An unstructured mesh solver typically consists of a single (indirect addressed) loop over all interior mesh elements, and another similar loop over all boundary elements. On a vector machine, each loop may be split into groups (colors) such that within each group, no recurrences occur. Each group can then be vectorized. A simple parallelization strategy for a shared memory machine is to further split each group into n subgroups, where n is the number of available processors. Each subgroup can then be vectorized and run in parallel on its associated processor. Because the original number of groups is not large (usually 20 to 30), the vector lengths within each subgroup are still long enough to obtain the full vector speedup of the machine, for a moderate number of processors. For more massively parallel distributed-memory scalar machines, the entire mesh must be subdivided and each resulting partition associated with a single processor. On each processor, the single scalar interior and boundary loops are then executed, with inter-processor communication occurring at the beginning and end of each loop. The mesh partitioning strategy must ensure good load balancing on all processors while minimizing the amount of inter-processor communication required.

7.2. CRAY-YMP-8 Results

Figure 17 illustrates an unstructured mesh generated over a three-dimensional aircraft configuration. This mesh contains a total of 106,064 points and 575,986 tetrahedra. This represents the second finest mesh employed in the multigrid sequence. The finest mesh, which is not shown due to printing resolution limitations, contains a total of 804,056 points and approximately 4.5 million tetrahedra. This is believed to be the largest unstructured grid problem attempted to date. The inviscid flow was solved on this mesh using all eight processors running in parallel on the CRAY-YMP supercomputer. A total of 4 meshes were used in the multigrid sequence. The convergence rate for this case is depicted in Figure 19. In 100 multigrid cycles, the residuals were reduced by almost 6 orders of magnitude. This run required a total of 16 minutes wall clock time running in dedicated mode on the 8 processor CRAY-YMP, including the time to read in all the grid files, write out the solution, and monitor the convergence by summing and printing out the average residual throughout the flow field at each multigrid cycle. The total memory requirements for this job were 94 million words. The ratio of CPU time to wall clock time was 7.7 on 8 processors, and the average speed of calculation was 750 Mflops, as measured by the CRAY hardware performance monitor [25]. For this case, the freestream Mach number is 0.768 and the incidence is 1.116 degrees. The computed Mach contours are shown in Figure 18, where good resolution of the shock on the wing is observed, due to the large number of mesh points employed.

7.3. Intel iPSC 860 Results

The implementation of the unstructured multigrid Euler solver on the Intel iPSC 860 distributed memory scalar multiprocessor machine, has been pursued using a set of software primitives designed to ease the porting of scientific codes to parallel machines [26]. The present implementation was undertaken as part of a more general project aimed at designing and constructing such primitives with experience gained from various implementations. The net effect of the use of such primitives is to relieve the programmer of most of the low level machine dependent software programming tasks. The mesh was partitioned using a spectral partioning algorithm which had previously been shown to produce good load balancing and minimize inter-processor communication [27]. At present, the partitioning of the mesh is done in a preprocessing stage on a sequential machine. At the time of writing, the fine aircraft mesh (804,056 vertices) has not been run on the Intel machine. Thus, results with coarser meshes are quoted. Table 1 gives an overview of the results obtained to date. A small 3600 point mesh was found to run at about 4.1 Mflops on a single Intel iPSC 860 processor. The largest case tried to date, a 210,000 point mesh, resulted in a 144 Mflop rate on 64 processors, which represents an efficiency of about 55% percent, based on the single processor results. It is anticipated that the fine 804,056 point grid, when implemented on 512 processors, will achieve an equivalent or greater computational speed than that observed with the full CRAY-YMP 8-processor machine.

8. CONCLUSION

This paper has illustrated the application of unstructured mesh techniques to various types of aerodynamic flows, and emphasized the advantages which can be obtained for complex geometries using adaptive meshing and parallelization. In two dimensions, a viscous flow solution capability has been demonstrated, while in three dimensions, efficient Euler solutions are possible. The main problems associated with three-dimensional viscous solutions are related to the development of reliable grid generation strategies, particularly with regards to the generation of highly stretched tetrahedral elements for capturing thin viscous layers. Turbulence modeling is also a limiting factor, although this difficulty is not particular to the field of unstructured meshes. Future work should also concentrate on more complete parallelization of the entire solution process, including items such as grid generation, partitioning, and adaptive meshing.

ACKNOWLEDGEMENTS

Much of this work has been possible due to the generous assistance of a large number of people. Among those, Daryl Bonhaus is acknowledged for his work in helping validate the 2-D viscous flow solver. The help of Shahyar Pirzadeh and Clyde Gumbert in providing assistance with the generation of three-dimensional unstructured grids using the VGRID program is acknowledged. Rob Vermeland and CRAY Research Inc. are thanked for providing dedicated time on the CRAY-YMP-8 machine, and the National Aerodynamic Simulation Facility (NAS), for providing the computational facilities which have made most of this work possible. Finally, Raja Das, Joel Saltz and Ravi Ponnusamy are acknowledged for actually having done most of the work on the distributed memory parallel implementation.

REFERENCES

1. Dannenhoffer, J. F., "Grid Adaptation for Complex Two-Dimensional Transonic Flows", *Sc.D Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology*, August 1987.

2. Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow over a Complete Aircraft", *AIAA paper 86-0103*, January, 1986.

3. Mavriplis, D. J., "Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes", *Ph.D. Thesis, Department of Mechanical and Aerospace Engineering, Princeton University*, June 1987.

4. Jameson, A., "Transonic Flow Calculations" *Princeton University Report MAE 1751*, 1984

5. Martinelli, L., "Calculations of Viscous Flows with a Multigrid Method" *Ph.D Thesis*, Department of Mechanical and Aerospace Engineering, Princeton University, October, 1987.

6. Mavriplis, D. J., and Jameson, A., "Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes", *AIAA Journal, Vol 28, No. 8, pp. 1415-1425*, August 1990.

7. Lallemand, M. H., Dervieux, A., A Multigrid Finite-Element Method for Solving the Two-Dimensional Euler Equations", *Proceedings of the Third Copper Mountain Conference on Multigrid Methods, Lecture Notes in Pure and Applied Mathematics*, Ed S. F. McCormick, Marcel Dekker Inc., April 1987, pp. 337-363.

8. Smith, W. A., "Multigrid Solution of Transonic Flow on Unstructured Grids", *Recent Advances and Applications in Computational Fluid Dynamics, Proceedings of the ASME Winter Annual Meeting*, Ed. O. Baysal, November 1990.

9. Barth, T. J., "Numerical Aspects of Computing Viscous High-Reynolds Number Flows on Unstructured Meshes", *AIAA Paper 91-0721* January, 1991

10. Bowyer, A., "Computing Dirichlet Tessalations", *The Computer Journal*, Vol. 24, No. 2, 1981, pp. 162-166

11. Baker, T. J., "Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets", *Proc. of the AIAA 8th Comp. Fluid Dyn. Conf.*, AIAA paper 87-1124, June, 1987.

12. Mavriplis, D. J., "Three Dimensional Unstructured Multigrid for the Euler Equations", *Proc. of the AIAA 10th Comp. Fluid Dyn. Conf.*, AIAA paper 91-1549, June, 1991.

13. Sieverding, C. H., "Experimental Data on Two Transonic Turbine Blade Sections and Comparisons with Various Theoretical Methods", *Transonic Flows in Turbomachinery*, VKI Lecture Series 59, 1973.

14. Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations", *J. Comp. Phys.*, Vol 72, October, 1987, pp. 449-466

15. Gumbert, C., Lohner, R., Parikh, P., and Pirzadeh, S., "A Package for Unstructured Grid Generation and Finite Element Flow Solvers", *AIAA paper 89-2175* June, 1989.

16. George, P. L., Hecht, F., Saltel, E., "Maillage Automatique de Domaines Tridimensionels Quelconques", *INRIA Report No. 1021, Institut National de Recherche en Informatique et en Automatique, Rocquencourt, France*, April 1989.

17. Mavriplis, D. J., "Unstructured and Adaptive Mesh Generation for High Reynolds Number Viscous Flows", *Proc. of the 3rd Int. Conf. on Numerical Grid Generation in Comp. Fluid Dyn.*, Eds. A. S. Arcilla, J. Hauser, P. R. Eisman, and J. F. Thompson, Pineridge Press Ltd., 1988, pp. 79-91.

18. Nakahashi, N., "FDM-FEM Zonal Approach for Viscous Flow Computations Over Multiple Bodies", *AIAA paper 87-0604*, January, 1987.

19. Holmes D. G., and Connell, S., "Solution of the 2-D Navier-Stokes Equations on Unstructured Adaptive Grids", *AIAA paper 89-1932, Proc. of the AIAA 9th Computational Fluid Dynamics Conference, Buffalo, NY*, June, 1989.

20. Mavriplis, D. J., "Algebraic Turbulence Modeling for Unstructured and Adaptive Meshes", *AIAA Paper 90-1653*, June, 1990.

21. Wigton, L. B., *Private Communication*, The Boeing Company

22. Bonhaus, D., "Computed Flows of Sulfur Hexafluoride Over Mulu-Element Airfoils" *NASA TP (Technincal Paper)*, To Appear.

23. Speziale, C. G., Abid, R., and Anderson, E. C., "A Critical Evaluation of Two-Equation Models for Near Wall Turbulence", *ICASE Report 90-46, NASA CR 182068, AIAA paper 90-1481*, June, 1990

24. Mavriplis, D. J., "Multigrid Solution of Compressible Turbulent Flow on Unstructured Meshes Using a Two-Equation Model", *AIAA Paper 91-0237*, January 1991.

25. UNICOS Performance Utilities Reference Manual, *SR-2040 6.0* Cray Research Inc.

26. Das, R., Mavriplis, D. J., Saltz, J., Ponnusamy, R., "The Design and Implementation of a Parallel Unstructured Euler Solver Using Software Primitives", *AIAA paper 92-0562*, January 1992.

27. Simon, H., "Partitioning of Unstructured Mesh Problems for Parallel Processing", *Proceedings of the Conference on Parallel Methods on Large Scale Structural Analysis and Physics Applications*, Permagon Press, 1991.

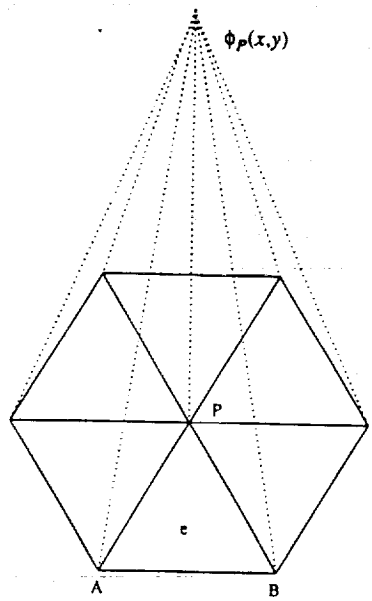
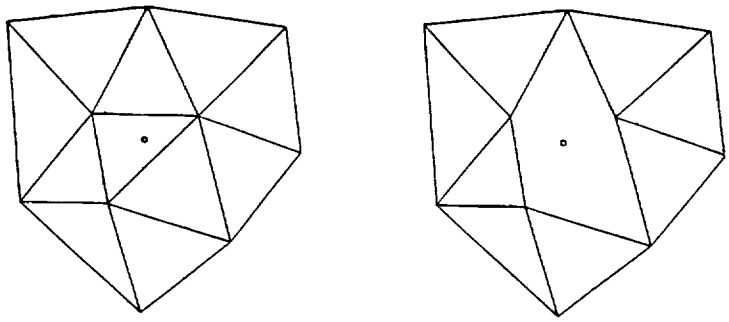
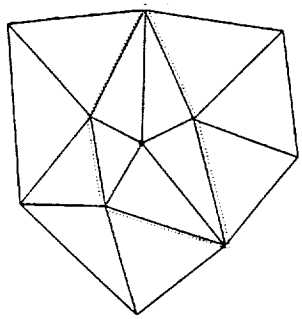


Figure 1: Domain of Influence of Finite-Element Basis Function and Equivalent Finite-Volume Control Volume



a)
Insertion of New Point

b)
Determination of Intersected Triangles
and Removal of Mesh Structure in this Region



c)
Local Restructuring of Mesh in Affected Region

Figure 2: Illustration of Bowyer's Algorithm for Delaunay Triangulation

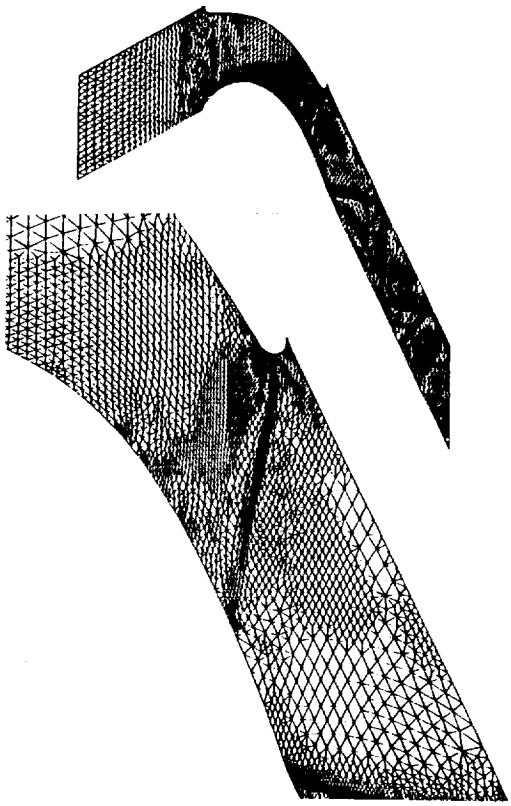


Figure 3: Adaptive Mesh Employed for Computing Transonic Inviscid Flow Through a Periodic Turbine Blade Cascade Geometry; Number of Nodes = 9362

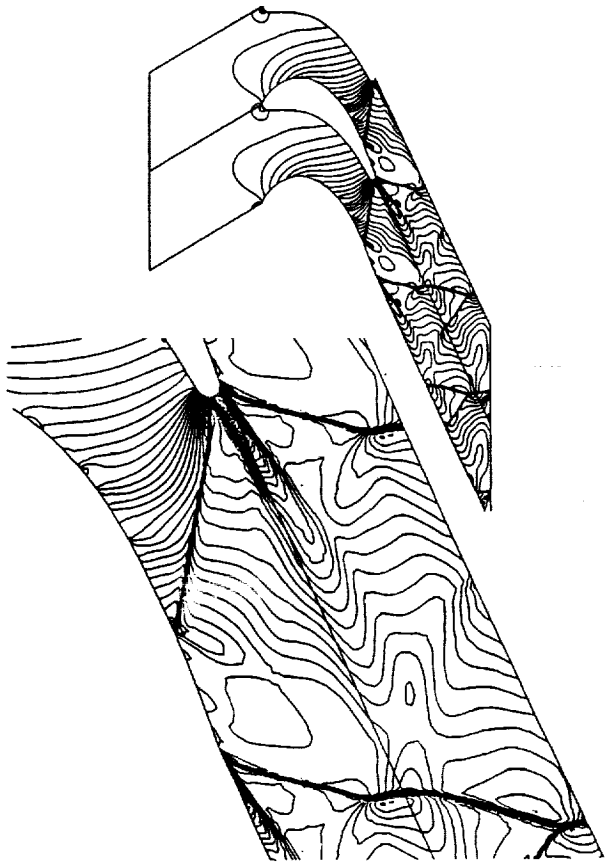


Figure 4: Computed Mach Contours for Flow Through a Periodic Turbine Blade Cascade Geometry

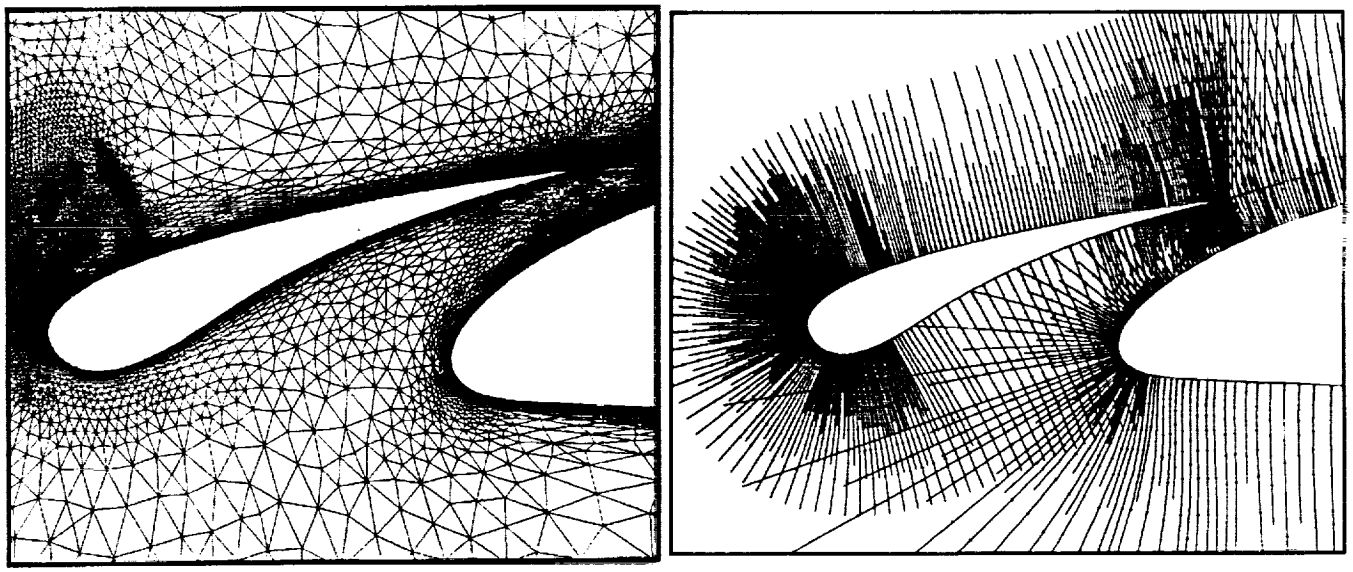


Figure 5: Illustration of Turbulence Mesh Stations Employed in Algebraic Model for an Adaptively Generated Mesh

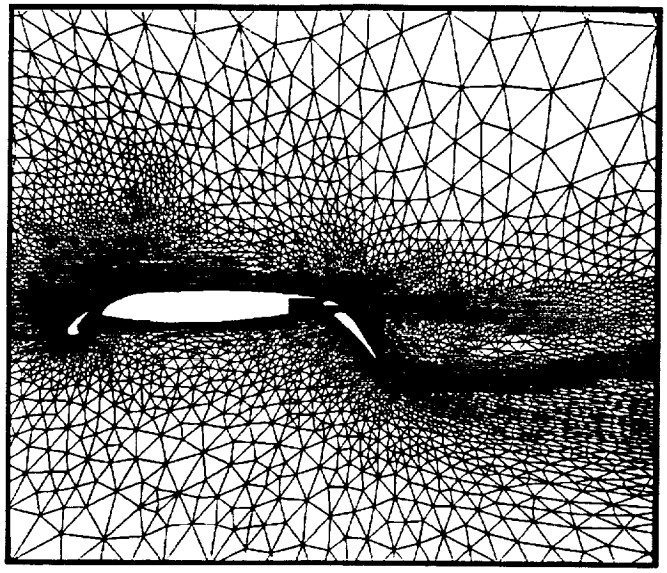


Figure 6: Adaptively Generated Unstructured Mesh about Four-Element Airfoil; Number of Nodes = 48,691

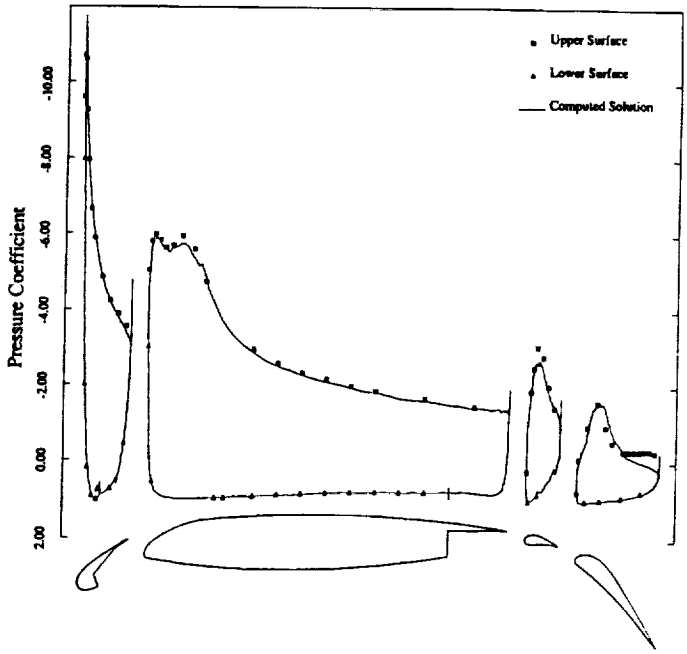


Figure 8: Comparison of Computed Surface Pressure Distribution with Experimental Wind-Tunnel Data for Flow Over Four-Element Airfoil Configuration; Mach = 0.1995, Reynolds Number = 1.187 million, Incidence = 16.02 degrees

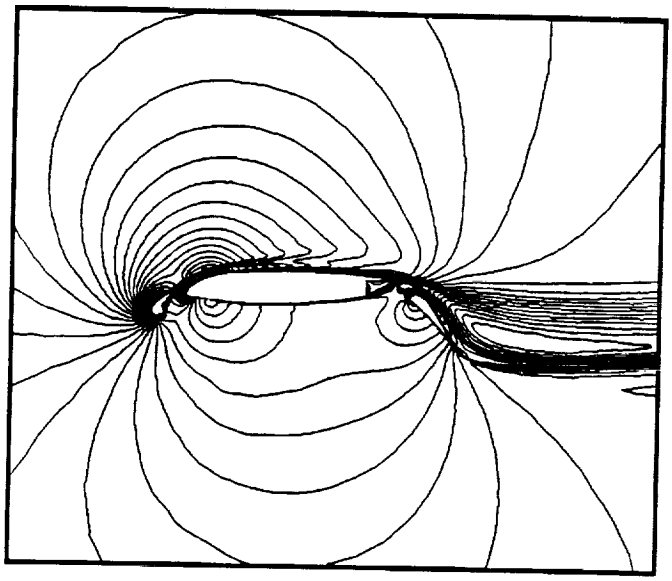


Figure 7: Computed Mach Contours for Flow over Four-Element Airfoil; Mach = 0.1995, Reynolds Number = 1.187 million, Incidence = 16.02 degrees

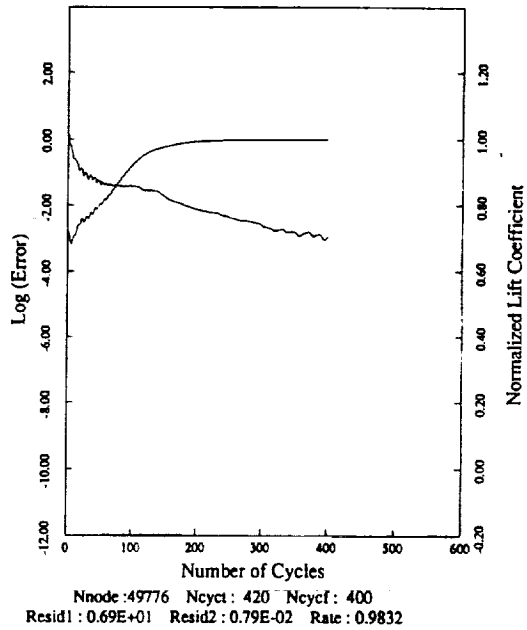


Figure 9: Convergence as Measured by the Computed Lift Coefficient and the Density Residuals Versus the Number of Multigrid Cycles for Flow Past a Four-Element Airfoil

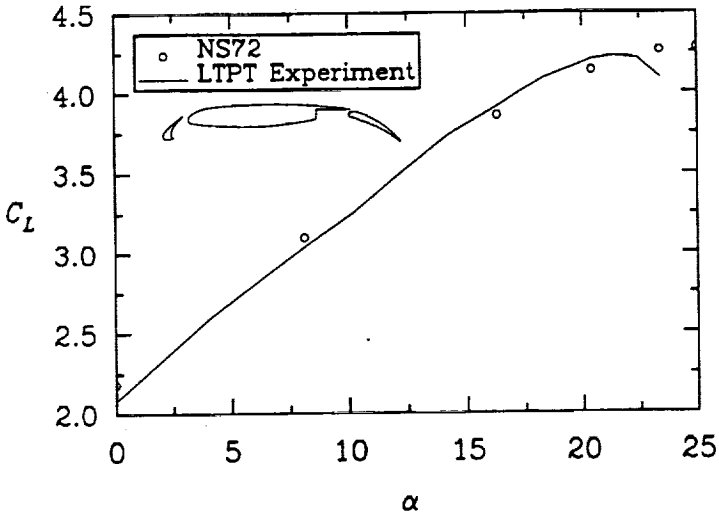


Figure 10: Comparison of Computed and Experimental Lift Coefficients as a Function of Incidence for a Three-Element Airfoil Configuration

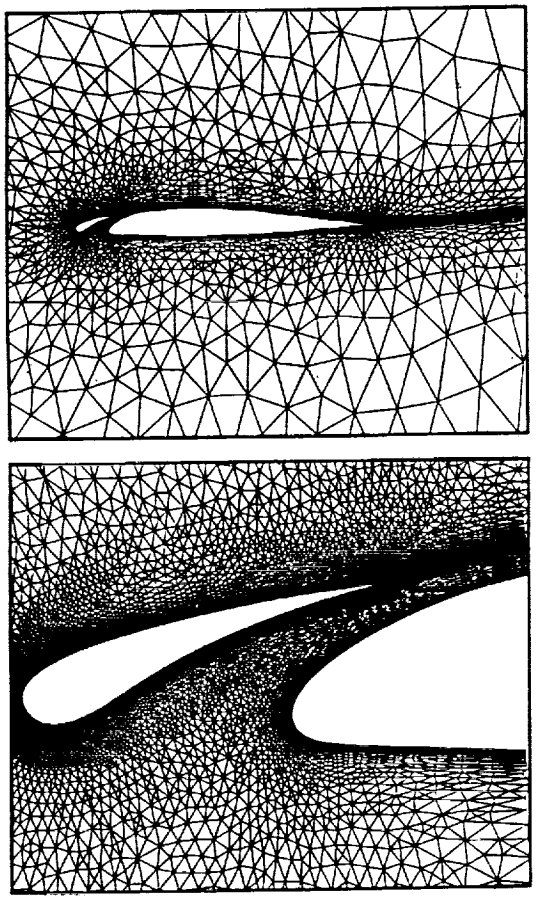


Figure 11: Global View of Coarse Unstructured Mesh and Close-Up View of Fine Unstructured Mesh Employed for Computing Flow Past a Two-Element Airfoil (Coarse Mesh Points = 7272, Fine Mesh Points = 28871)

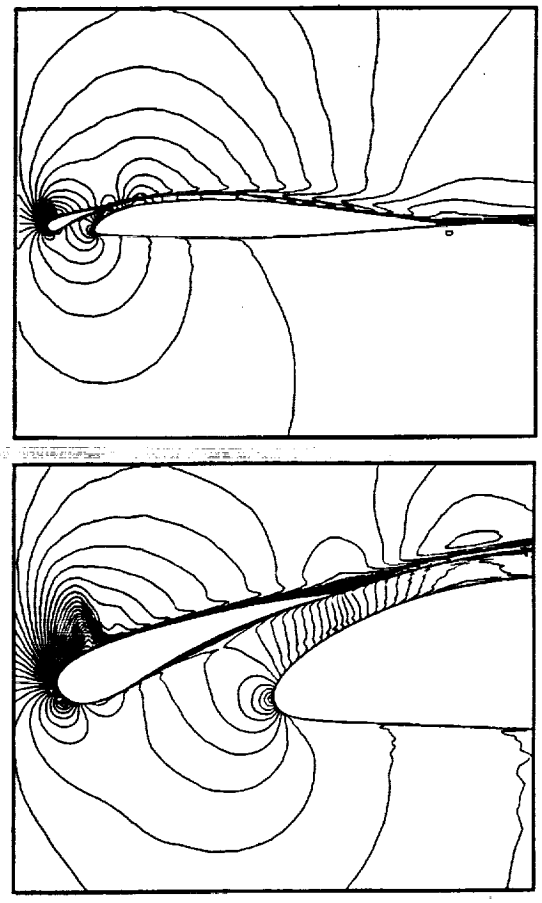


Figure 12: Computed Mach Contours Using Low-Reynolds Number Modification for Turbulence Equations for Supercritical Flow over a Two-Element Airfoil (Mach = 0.5, Re = 4.5 million, Incidence = 7.5 degrees)

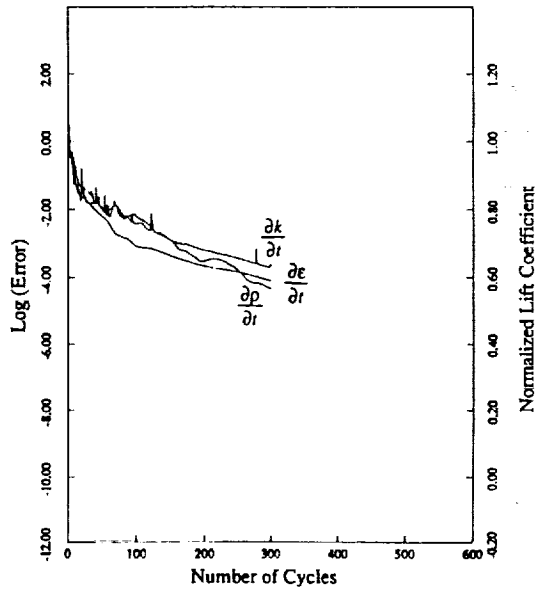


Figure 13: Multigrid Convergence Rate of the Density Equation and the Two Turbulence Equations Using Low-Reynolds Number Modifications for Flow Over Two-Element Airfoil (Mach = 0.5, Re = 4.5 million, Incidence = 7.5 degrees)

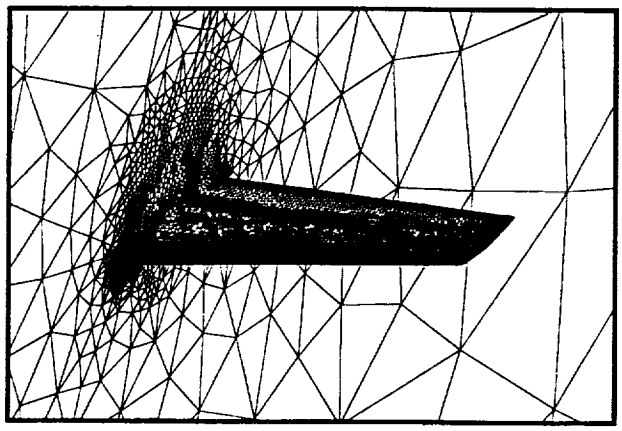


Figure 14: Finest Adapted Mesh Generated About ONERA M6 Wing (Number of Nodes = 173,412 Number of Tetrahedra = 1,013,718)

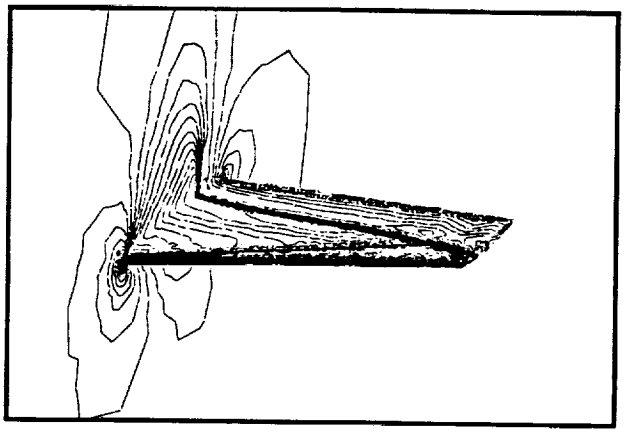


Figure 15: Computed Mach Contours on the Adaptively Generated Mesh About the ONERA M6 Wing (Mach = 0.84, Incidence = 3.06 degrees)

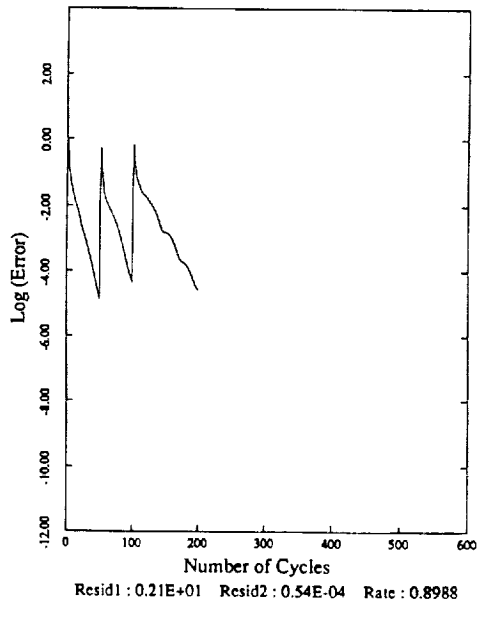


Figure 16: Convergence Rate of the Unstructured Multigrid Algorithm on the Adaptively Generated Sequence of Meshes about the ONERA M6 Wing as Measured by the Average Density Residuals Versus the Number of Multigrid Cycles

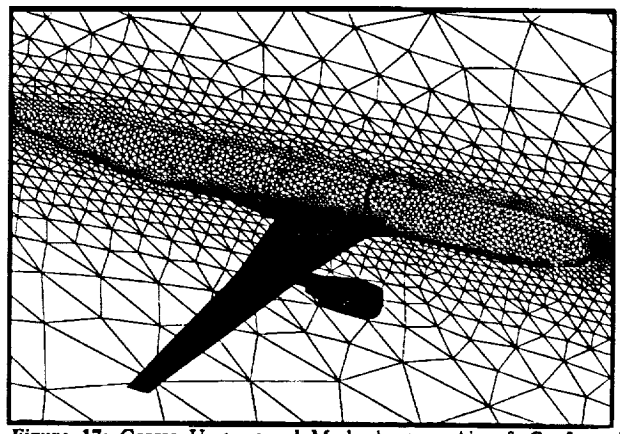


Figure 17: Coarse Unstructured Mesh about an Aircraft Configuration with Single Nacelle; Number of Points = 106,064, Number of Tetrahedra = 575,986 (Finest Mesh Not Shown)

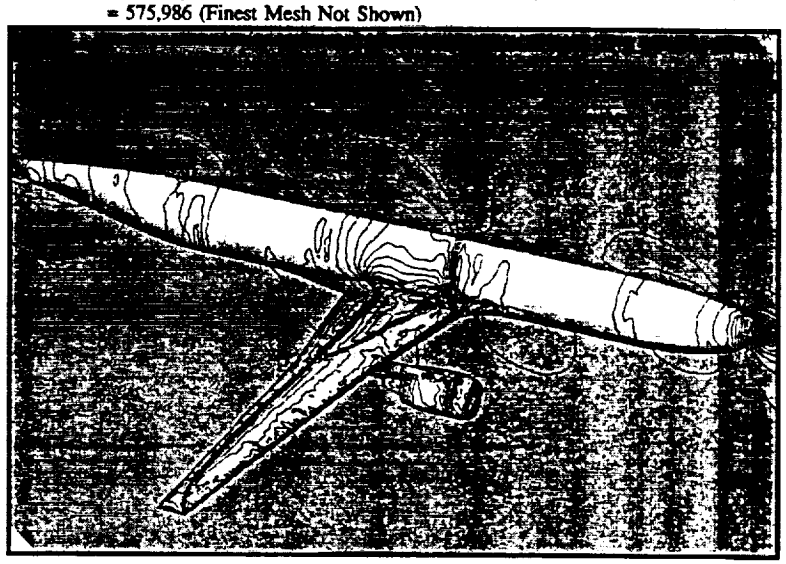


Figure 18: Mach Contours for Flow over Aircraft Configuration Computed on Fine Mesh of 804,056 Vertices and 4.5 million Tetrahedra (Mach = 0.768, Incidence = 1.116 degrees)

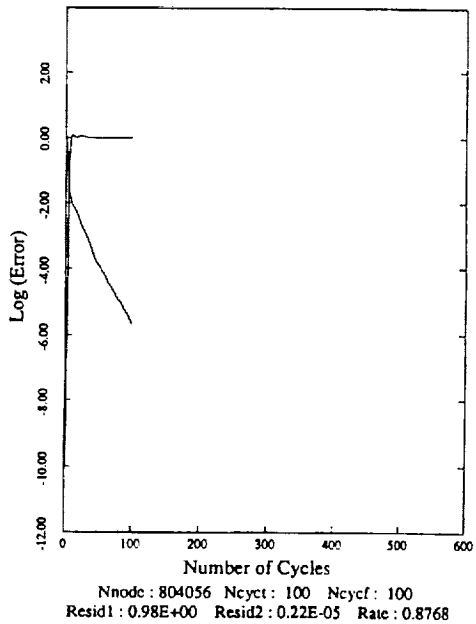


Figure 19: Multigrid Convergence Rate on Finest Mesh of the Multigrid Sequence for Transonic Flow over Aircraft-with-Nacelle Configuration

Size Mesh		Number of Processors				
		1	2	8	16	64
3600	Mflops	4.1	7.1	16.9	17.4	-
	comp/iter(s)	4.6	2.4	0.6	0.34	-
	comm/iter(s)	-	0.25	0.48	0.73	-
26K	Mflops	-	-	23.8	38.8	
	comp/iter(s)	-	-	4.5	2.3	
	comm/iter(s)	-	-	1.1	1.1	
210K	Mflops	-	-	-	-	144.3
	comp/iter(s)	-	-	-	-	4.75
	comm/iter(s)	-	-	-	-	2.3

Table 1: Observed Computational Rates and Timings per Iteration of Computational Work and Communication Overhead for Various Sizes of Unstructured Meshes on Intel iPSC/860