ADAPTIVE UNSTRUCTURED TRIANGULAR MESH GENERATION AND FLOW SOLVERS FOR THE NAVIER-STOKES EQUATIONS AT HIGH REYNOLDS NUMBER

Gregory A. Ashford* and Kenneth G. Powell†

W. M. Keck Foundation Laboratory for Computational Fluid Dynamics
Department of Aerospace Engineering
The University of Michigan
Ann Arbor, MI 48109-2118, U.S.A.

Abstract

A method for generating high quality unstructured triangular grids for high Reynolds number Navier-Stokes calculations about complex geometries is described. Careful attention is paid in the mesh generation process to resolving efficiently the disparate length scales which arise in these flows. First the surface mesh is constructed in a way which ensures that the geometry is faithfully represented. The volume mesh generation then proceeds in two phases thus allowing the viscous and inviscid regions of the flow to be meshed optimally. A solution-adaptive remeshing procedure which allows the mesh to adapt itself to flow features is also described. The procedure for tracking wakes and refinement criteria appropriate for shock detection are described. Although at present it has only been implemented in two dimensions, the grid generation process has been designed with the extension to three dimensions in mind. An implicit, higher-order, upwind method is also presented for computing compressible turbulent flows on these meshes. Two recently developed one-equation turbulence models have been implemented to simulate the effects of the fluid turbulence. Results for flow about a RAE 2822 airfoil and a Douglas three-element airfoil are presented which clearly show the improved resolution obtainable.

1 Introduction

The desire in the engineering community to simulate numerically flows about increasingly complex geometries has fueled interest in the development of unstructured grid methods. These methods provide great flexibility in dealing with the complex geometries encountered in practice and offer a natural framework for solution-adaptive mesh refinement. As attention has turned to solutions of the full Navier-Stokes equations, several methods have been recently developed to address the special requirements imposed on the grid generator. At the high Reynolds numbers encountered in typical aerodynamic applications, the viscous effects are felt predominantly in the very thin boundary layers adjacent to solid surfaces and in the wakes. In these regions, the normal length scale can be many orders of magnitude smaller than the tangential length scale. For

*Doctoral Candidate, Aerospace Engineering and Scientific Computing
†Associate Professor, Aerospace Engineering

efficiency this requires that the mesh be highly stretched in the viscous regions while local isotropy is desired in the mainly inviscid regions of the flow.

The generation of an unstructured triangular mesh for a complex geometry is usually accomplished in two phases. In the first, the boundaries of the domain are discretized to form the surface mesh. During this phase, one desires a method which renders a faithful discretization of the geometry by taking into account effects such as curvature and proximity to nearby bodies [1]. In the next phase of the process, the volume mesh is generated filling the domain with triangles. A number of special techniques have been developed to generate the highly stretched elements which are desired in the thin boundary layers and wakes encountered in high Reynolds number flows. Some of these methods use portions of a structured or semi-structured mesh in the viscous regions which are then matched up with an unstructured mesh in the inviscid regions [2]. Another approach is to generate the mesh using prismatic elements [3]. Others are based on modifications to the Delaunay triangulation [4, 5] or to the advancing front method [6, 7].

A variety of algorithms have appeared for solving the compressible Navier-Stokes equations on unstructured meshes. They range from implicit solvers employing upwind methods [8, 9] to multigrid solvers based on a Galerkin finite element technique [10]. Progress has also recently been made in the development of turbulence models which are well-suited to implementation on unstructured meshes [11, 12].

In this work, a method for generating unstructured meshes suitable for high Reynolds number Navier-Stokes flows is described. First the viscous regions are meshed using a node lifting procedure which is a node-based advancing front method. The inviscid regions are then meshed with the traditional face-based advancing front method. A remeshing strategy is also described in which the solution on the current mesh is analyzed and regions which are found to be under-resolved are flagged for refinement when the new mesh is generated. In particular, shock waves and wakes can be well captured in only a few remeshings. An implicit upwind solver is also described for computing solutions to the Reynolds-averaged Navier-Stokes equations.

2 Mesh Generation

The mesh generation process for high Reynolds number Navier-Stokes flows is driven by the need to capture efficiently the thin boundary layers and wakes which occur in these flows. In order to take advantage of the fact that in these regions normal gradients can be many orders of magnitude larger than tangential gradients, the mesh needs to be highly stretched. In addition to control over the stretching, control over the element shape is also desirable. In particular, elements with very large obtuse angles can lead to accuracy problems and should be avoided [13]. It is here that techniques which introduce stretching by means of a mapping [4, 14] can have difficulty.

Ideally, in the anisotropic viscous regions of the flow the elements should be high aspect ratio nearly right triangles while in the mainly isotropic inviscid regions the elements should be nearly equilateral triangles. In this work, this is accomplished by first meshing the viscous regions with a node lifting algorithm and then meshing the remainder of the domain with an advancing front algorithm. A smooth transition between the two regions is assured by dividing the viscous mesh into two layers: a viscous layer and a transition layer. The user has full control over the thickness, the number of points, and the mesh stretching in the viscous layer. The transition layer then serves as a buffer between the edge of the viscous layer and local isotropy at the edge of the viscous mesh. The mesh generation process is detailed in the following sections.

2.1 Geometry Description

The mesh generation procedure begins with a definition of the domain boundaries. In this work a very flexible approach has been taken whereby the boundary curves are given as the union of parametric splines. The particular parameterization chosen here is that of chord length along the spline. Parts of the domain boundary may also be curves describing the locations of wake centerlines in the flow. For example, these may be obtained by streamline traces from an initial solution on a coarse mesh. Each spline is C^2 on the interior while slope discontinuities are permitted at the endpoints. This approach allows for the easy description of very complex multiply-connected domains.

2.2 Surface Mesh Generation

Once the geometry has been defined, the next step is the construction of the surface mesh. This involves the triangulation of the domain boundaries. In 2-D this means the creation of edges and nodes on the boundary curves while in 3-D this would involve the construction of a surface triangulation. Initially each boundary curve is divided into a user-specified number of edges of nearly equal length by specifying a uniform discretization in parameter space. This serves to control the maximum spacing which will be allowed. Next each curve is refined based on a curvature criterion. If the angle formed by the two edges incident to an interior node exceeds a user-specified tolerance (typically 5°), then these edges and their neighbors are flagged for refinement. Flagging neighbors as well as the offending edges expands the region of refinement slightly and produces smoother discretizations. Upon examination of all the nodes, each of the flagged edges is subdivided into two by placing a new node (on the spline) near the edge midpoint. The new distribution of nodes is then smoothed by applying a few sweeps of a Laplacian filter in parameter space. The process is then repeated until the angle criterion is satisfied at every interior node. The process is guaranteed to converge since the splines are at least C^1.

Next the discretization is refined further based on proximity to nearby bodies. The object here is to avoid situations in which the local tangential spacing along the curve is large compared to the distance to a nearby body. In general, if such a situation is allowed to persist, the mesh generator has no choice but to produce badly shaped elements. In fact, in extreme situations, it may fail entirely. Edges which are longer than five times the distance to a nearby body are detected and refinement proceeds recursively as described above. A provision is also made for the user to specify the maximum tangential spacing which can be tolerated at specific locations on the boundary curves. This is useful for clustering points in regions where increased activity is anticipated but is not otherwise apparent from the geometry alone (e.g., trailing edges).

At this point, since each curve has been discretized independently, the tangential spacings on either side of a node at which two curves join may differ substantially. This is then remedied by refining near the endpoint of the curve with the larger spacing until the tangential spacings are comparable. The surface mesh generation concludes with a final smoothing sweep.

2.3 Volume Mesh Generation

The first step in the volume mesh generation procedure is the construction of the background grid. The purpose of the background grid is to specify the local (isotropic) element size throughout the domain. It is constructed by first performing a Delaunay triangulation [15, 16] of the surface nodes and then converting this to a (local) MinMax triangulation via edge swapping [17]. The spacing value at each node is taken to be the average length of the incident boundary edges. Linear

interpolation then provides the spacing function over the whole domain. In most regions of the domain this works quite well resulting in linear variation in element size from the (usually) finely discretized inner boundaries to the coarsely discretized outer boundary. However, this triangulation sometimes produces connections between two widely separated regions of very fine discretization. This then tends to produce an overly fine mesh in regions where it is not desired. This situation is easily remedied by inserting a small number of additional control points into the Delaunay triangulation prior to the edge swapping to break up these unwanted connections. At present this is performed by the user upon examination of the background grid, but a procedure to automate this is under development. This would allow the background grid to be generated automatically from the surface mesh. We believe this is preferable to the traditional approach of first requiring the user to provide the background grid from which the surface mesh is then generated. This is especially true in 3-D where the specification of a background grid which will yield the desired surface resolution can be difficult.

A preliminary step toward the generation of the viscous mesh is the computation of an average surface normal for each boundary node. The surface normals at the nodes are computed by looping over the edges and scattering the contribution due to the edge to each of its two nodes. In order to handle wake cuts, the edges (and nodes) on the wake cut are first duplicated and added to the list of edges but with opposite orientation. This in effect creates a two-sided surface which can then be treated in the standard way. The surface normals are then smoothed with several passes of a Laplacian filter. This smoothing tends to produce better meshes in regions near surface slope discontinuities. At the very end of the mesh generation procedure a clean-up utility is called which removes the duplicate edges and nodes by fusing them with their parents.

With each surface node is associated a local viscous layer thickness, δ_{vl}. When constructing the initial mesh, this is computed by assuming Blasius boundary layer growth (at the given Reynolds number) along the surface of each body starting from a user-specified stagnation point location. Also along any wake curves a similar scaling analysis is used to prescribe an appropriate wake thickness as a function of position. When incorporated within an adaptive remeshing process, the viscous layer thickness can be determined from the existing solution. In particular, since turbulence models are frequently sensitive to the initial y^+ spacing of the first node off the wall, this information can be easily incorporated. Three additional global parameters are asked of the user: n_{vl}, the number of points in the viscous layer, r_{vl}, the stretching ratio in the viscous layer, and r_{tl}, the stretching ratio in the transition layer. Since the stretching ratio is the ratio of the heights of the cells at successive levels, these form a geometric series whose sum is the local viscous layer thickness

$$\delta_1(1 + r_{vl} + r_{vl}^2 + ... + r_{vl}^{n_{vl}-1}) = \delta_{vl}. \tag{1}$$

From this the initial height, δ_1, can then be determined. The heights of cells at successive levels are then given by

$$\delta_n = \delta_{n-1} \begin{cases} r_{vl}, & n \leq n_{vl} \\ r_{tl}, & n > n_{vl} \end{cases} \tag{2}$$

Typical ranges for values of the parameters are $n_{vl} = 10 - 20$, $r_{vl} = 1.3 - 1.7$, and $r_{tl} = 1.4 - 1.8$.

A node lifting process is then used to generate the viscous mesh. This is an advancing front method where the advancement is node-based rather than face-based. The front is initialized to consist of the boundary nodes and edges. These nodes are designated as being at level 0. Advancement begins by selecting a node on the front and marching it out along the local surface

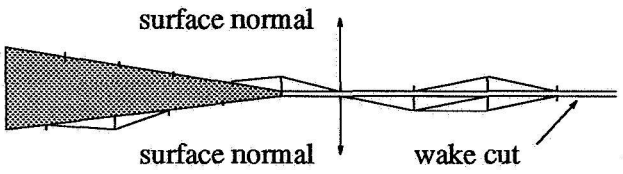

Figure 1: Node lifting example near the trailing edge of an airfoil showing a wake cut.

normal a distance δ_1. In the process, two cells, three edges (two of which form faces on the front), and one node (now at level 1) are created while two faces and one node formally on the front are deleted (see Figure 1). Once all nodes at the current level have been advanced, the process continues with nodes on the next level. Advancement terminates at a node if

- a cell of less than unit aspect ratio would be produced, or
- δ_n exceeds the local background grid spacing, or
- a node forming the base of a new triangle is at a lower level than the node being lifted and the angle between the base and the normal exceeds a threshold (taken to be 120°), or
- an intersection would occur with another edge on the front.

The viscous mesh about the slat for a Reynolds number of 9 million is shown in Figure 2. Note how the transition mesh provides a smooth transition from the highly stretched cells in the boundary layer and wake to local isotropy at the edge of the viscous mesh.

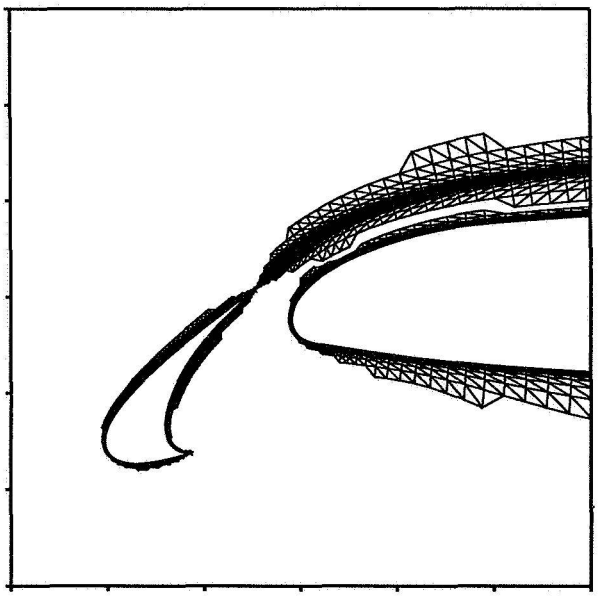

Figure 2: Viscous mesh about a leading-edge slat with a wake.

With the viscous mesh complete, the remainder of the domain is meshed with a traditional face-based advancing front method [14]. This method builds the mesh one element at a time by advancing the boundary of the domain inward. Briefly, the procedure is

- initialize the front
- while(there are faces on the front) do
  - select a face on the front to be the base of the new element
  - obtain the local spacing value from the background grid
  - determine the location of a new 'ideal' point
  - find the nearby nodes and faces on the front
  - decide whether to connect with an existing node on the front or introduce the 'ideal' point
  - form the new element by updating the data structures
- smooth the mesh with a Laplacian filter.

Efficient implementation of this procedure requires the use of dynamic data structures. Typically the front is advanced from its shortest face to help prevent larger cells from overlapping smaller ones. A priority queue of faces which allows efficient insertion and deletion is implemented using a heap. An alternating digital tree [18] is used to locate nearby nodes on the front. Nearby faces on the front are then found using node to face pointers. With these data structures the mesh can be generated in $O(N \log N)$ time while incurring minimal storage overhead. Typical volume mesh generation times are about 35 seconds for a 30,000 node mesh on a 24 MIPS DECstation 5000/200.

The complete mesh consisting of about 31,300 nodes for a Douglas three-element airfoil at a Reynolds number of 9 million is shown in Figure 3. In this example, the locations of the wakes off the leading-edge slat and the main element have been determined by streamline traces from a coarse grid solution. Both wakes have been tracked to slightly downstream of the airfoil at which point the wake grids end.

3 Adaptive Remeshing

One of the advantages of using an unstructured mesh approach is that it provides a natural framework for the incorporation of solution-adaptive mesh refinement. In this process, the mesh is refined locally based on an estimate of the solution error. Since numerical errors tend to be largest in regions where the solution is changing most rapidly, these are generally good candidates for refinement. Conversely, in regions showing little activity, the mesh can often be coarsened with little degradation in solution accuracy. By concentrating mesh points where they are most needed, high quality solutions can be obtained at reasonable computational cost. An unstructured approach facilitates this process because its data structure can easily support local refinement and coarsening of the mesh.

In this work, an adaptive remeshing procedure has been adopted. Guided by the solution on the current mesh, a new mesh is generated which is better suited to capturing the flow features. At the heart of this procedure is the construction of the background grid which will specify the desired spacing throughout the domain. Once this has been established, the new surface mesh can be generated by traversing each spline and discretizing it into line segments whose lengths are given by the background grid. Note that the surface mesh must be derived from the background grid

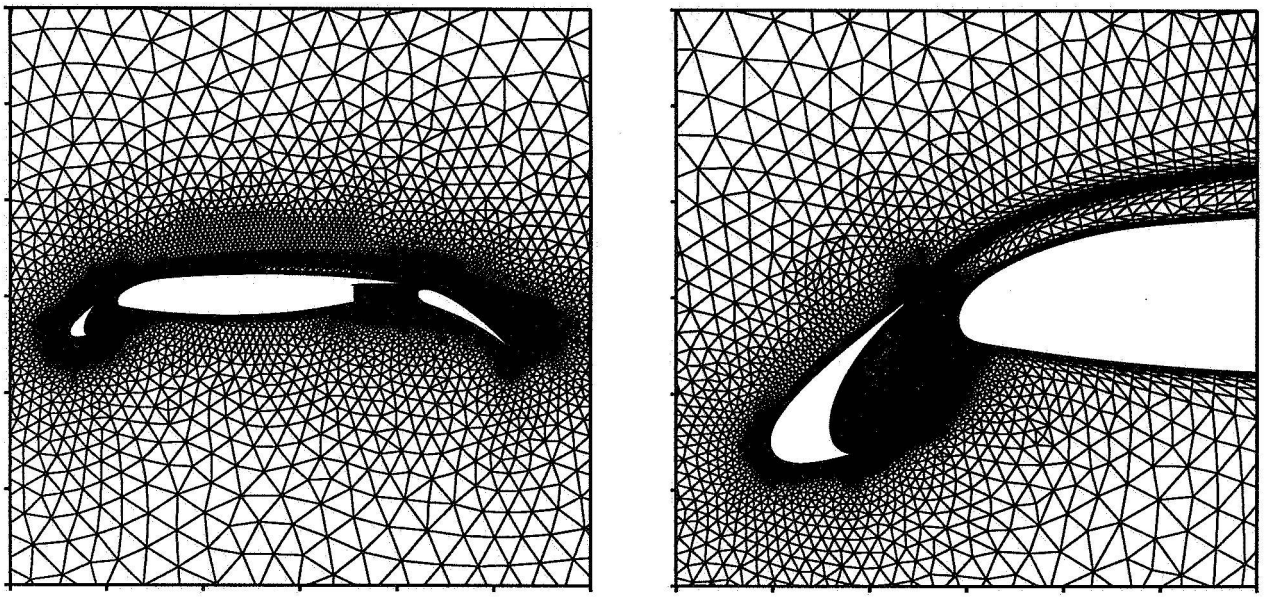

Figure 3: Complete mesh about a Douglas airfoil.

otherwise an inconsistency in the spacing will occur at the domain boundaries. The generation of the volume mesh may then proceed via the node lifting/advancing front algorithm described earlier. The current solution is interpolated onto the new mesh as the initial condition for the flow solver. The solution on the new mesh is then computed and the process repeated if the desired resolution has not been obtained.

By defining appropriate spacing values at the vertices, the current mesh can serve as the background grid. By analyzing the current solution, regions requiring more (or less) resolution are identified. For this purpose, a variety of refinement criteria which detect specific flow features can be used. Next the current mesh spacing is determined by assigning to each mesh vertex a value which reflects the average local element size. This spacing value is then modified in accordance with the selected refinement criteria to produce the background grid. For example, in regions requiring more resolution the spacing value would be reduced to reflect the fact that smaller elements are needed. The adaptive remeshing procedure can be summarized as:

while(solution quality is less than desired) do

- construct the spacing function at the vertices of the current mesh
- compute the refinement parameters
- create the background grid by modifying the spacing function in accordance with the refinement parameters
- generate the new surface mesh using the background grid
- generate the new volume mesh using the background grid
- interpolate the current solution onto the new mesh
- compute the solution on the new mesh.

For inviscid flows a spacing function on the current mesh can be constructed by assigning to each of its vertices the average length of the incident edges. This spacing function can then be smoothed by averaging the value at a vertex with the values at its first-order neighbors. Next the parameters on which the refinement is to based are computed at the mesh vertices. For inviscid flow, two parameters which have been found to work well are related to the local divergence and curl of the velocity field by

$$\tau_{1,i} = |\text{div } \mathbf{u}| \; h_i^p \tag{3}$$

$$\tau_{2,i} = \|\text{curl } \mathbf{u}\| \; h_i^p \tag{4}$$

where h_i is the local spacing value at vertex i and p (taken here to be 1.5) determines how strongly the element size influences the refinement. The divergence criterion measures the local compressibility of the flow and is effective in locating shock waves while the curl criterion measures the local rotationality of the flow and performs well in locating slip layers. The standard deviations from zero of the refinement parameters are then calculated from

$$\sigma_k^2 = \frac{1}{N} \sum_{i=1}^{N} \tau_{k,i}^2 \qquad k = 1, 2 \tag{5}$$

where N is the number of nodes in the mesh.

Refinement can then be effected by decreasing the spacing value wherever $\tau_{k,i}/\sigma_k$ is large. Conversely, the mesh can be coarsened by increasing the spacing value wherever $\tau_{k,i}/\sigma_k$ is small. In general, the desired spacing value can be written

$$h'_{k,i} = h_i \; g_k\left(\frac{\tau_{k,i}}{\sigma_k}\right) \tag{6}$$

where g_k is a non-increasing function which is 1 for intermediate values of $\tau_{k,i}/\sigma_k$ and is bounded away from very large and very small values as $\tau_{k,i}/\sigma_k \to 0$ and ∞ respectively. These bounds are necessary in order to prevent the spacing function, and hence the mesh, from changing too abruptly from one meshing to the next. Typically, g_k can be taken to be a simple piecewise linear function bounded so that $1/4 \leq g_k \leq 2$. Obviously different choices of refinement parameter lead to different values of the desired spacing. It is usually best to take the modified spacing value, h'_i, to be the smallest of these values

$$h'_i = \min_k h'_{k,i}. \tag{7}$$

This ensures that all the selected flow features are detected and resolved by the adaption. Linear interpolation on the triangles is then used to define the modified spacing distribution over the entire domain.

A few modifications are necessary when applying this remeshing procedure to a viscous flow. The spacing value from the current mesh must be constructed so that it is indicative of the local inviscid mesh scale. For example, in the highly stretched cells in the boundary layers and wakes of a high Reynolds number flow it is the local tangential mesh scale which the spacing value should reflect and not the very small normal mesh scale. The reason for this is that near a body or a wake centerline the purpose of the background grid is to specify the local tangential length scale; the local normal length scale is specified explicitly by δ_{vl}. An approach that has been found to work well is to take the spacing to be the average median side length of the triangles incident to the vertex. In the highly stretched cells in the boundary layers and wakes, this results in an appropriate

'streamwise' length scale. In nearly isotropic portions of the mesh, this reverts to a measure of the local average side length. While this has performed quite well, other choices are clearly possible.

Due to the highly anisotropic nature of the flow in the boundary layers and wakes, the refinement criteria must also be modified. In these regions, the refinement criteria should only detect the need for local tangential refinement (e.g., along the surface in the case of shock-boundary layer interaction); the proper normal length scale is accounted for through the specification of the local viscous layer thickness. A parameter which has been found to work well in detecting shocks in these situations is

$$\tau_{3,i} = \frac{p_{max,i} - p_{min,i}}{p_{max,i}} h_i^q \tag{8}$$

where $p_{min,i}$ and $p_{max,i}$ are the minimum and maximum values respectively of the pressure at the vertex and its first-order neighbors and q (taken here to be 0.5) controls the degree to which the local element size influences the refinement.

Other flow features of interest can be captured by similar means. For example, the large vortices which occur in the separated flow behind the slat and in the flap well of the three-element airfoil (and aft of the flap at high angles of attack) are regions where refinement would be beneficial. Since one distinguishing characteristic of these vortices is that they are regions of isotropic rotational flow, one might try using the curl criterion. However, although the curl is relatively large in these areas (on the order of 50), it is much larger in the boundary layer (where it can exceed 100,000). As it stands, refinement based on the curl would tend to flag the entire airfoil surface for refinement while leaving the regions of vortical flow undetected. This can be rectified by replacing h_i in the curl refinement criterion with a measure of the smallest local length scale. This effectively removes the highly stretched cells in the boundary layers and wakes from the computation of the refinement criterion.

4 Solution Algorithm

The Reynolds-averaged Navier-Stokes equations are discretized in space using a finite volume scheme in which the unknowns are associated with the mesh vertices and the median dual mesh is used to define the control volumes. The convective fluxes are evaluated using Roe's flux-difference splitting [19]. Higher-order accuracy is achieved by using a piecewise linear reconstruction within each control volume. A least-squares procedure is used to compute the solution gradients. This procedure is exact whenever the solution varies linearly over the support of the reconstruction. On the highly stretched meshes employed in Navier-Stokes computations, the least-squares procedure is preferable to a Green-Gauss path integration since it appears to be better conditioned. For flows involving discontinuities, it is necessary to limit the reconstructed gradient so that new extrema are not created. Experience has shown that it is sufficient to satisfy this condition at the Gauss points of the flux quadrature (i.e., the edge midpoints). For this purpose the limiter proposed by Barth and Jespersen is used [20]. The viscous terms are evaluated using a Galerkin finite element approximation with piecewise linear elements. On a uniform subdivided quadrilateral mesh this would result in central differencing the viscous terms.

A fully implicit scheme based on a backward Euler linearization of the equations is used to march to the steady state. The backward Euler method is

$$\frac{\Delta \mathbf{u}}{\Delta t} = \mathbf{R}(\mathbf{u}^{n+1}) \tag{9}$$

where $\Delta\mathbf{u} = \mathbf{u}^{n+1} - \mathbf{u}^n$ and $\mathbf{R}(\mathbf{u})$ is an operator representing the spatial discretization. The right hand side is then linearized about $\mathbf{u}^n$ resulting in

$$\left(\frac{I}{\Delta t} - \frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right) \Delta\mathbf{u} = \mathbf{R}(\mathbf{u}^n). \tag{10}$$

This produces a large sparse system of linear equations which needs to be solved at each time step. Since the support of the higher-order scheme is quite large, consisting of the node and its first- and second-order neighbors, typically only the first-order scheme is linearized. This also circumvents the difficulty of linearizing the inherently highly nonlinear limiting procedure. Also due to the complexity of linearizing Roe's flux, generally this is only done approximately. Notice that these approximations do not affect the accuracy at steady state; only the (pseudo) time history is altered. The no-slip and isothermal wall boundary conditions are made implicit by altering appropriate rows in the matrix.

In order to try to reduce memory requirements, a Gauss-Seidel relaxation scheme has been implemented to solve the linear system. This has the advantage of requiring no additional storage beyond that of the matrix itself and is completely vectorizable by using a coloring scheme. In fact, if one is willing to recalculate the matrix each subiteration, the matrix need not be stored at all. However, since the calculation of the matrix is expensive, the required CPU time would increase substantially. Usually 20 subiterations are performed each time step with a CFL number of 300. Typical performance is about 280 MFLOPS on one processor of a Cray Y-MP C90 leading to solution times of about 10-15 minutes for a medium-sized (40,000 node) mesh. The implicit code with the Gauss-Seidel solver currently requires about 300 words of memory per node. This could be reduced further by more frugal memory management.

To simulate the effects of fluid turbulence at high Reynolds numbers, the Baldwin-Barth [11] and Spalart-Allmaras [12] turbulence models have been implemented. These are both one-equation transport models which solve for a working variable related to the eddy viscosity throughout the domain. The turbulence model equation is integrated in time using an implicit method similar to that of the mean flow equations. In order to facilitate the incorporation of different turbulence models, the mean flow equations and turbulence model equation are decoupled in the time integration.

5 Results

The results after one remeshing for a computation at $M_\infty = 0.20$, $\alpha = 16°$, and $Re = 9 \times 10^6$ about the Douglas three-element airfoil are shown in Figure 4. For this calculation the Spalart-Allmaras turbulence model has been used. The wake emanating from the slat is well captured and in fact remains distinct from the boundary layer on the main element over almost all of its length. Although not shown, on the initial mesh (which did not employ wake grids) the boundary layer which develops on the slat merely ends at its trailing edge without any hint of the wake which naturally exists downstream. This is a good example of why a remeshing approach to solution adaption is favored here for these types of flows. If one were to attempt an adaption strategy based on h-refinement for this case, it is likely that many iterations would be required as the wake refinement is gradually propagated downstream from the trailing edge of the slat. Also mesh adaption strategies based on local enrichment have difficulty producing the well-shaped, highly stretched elements desired for the efficient capture of viscous features. A remeshing strategy,

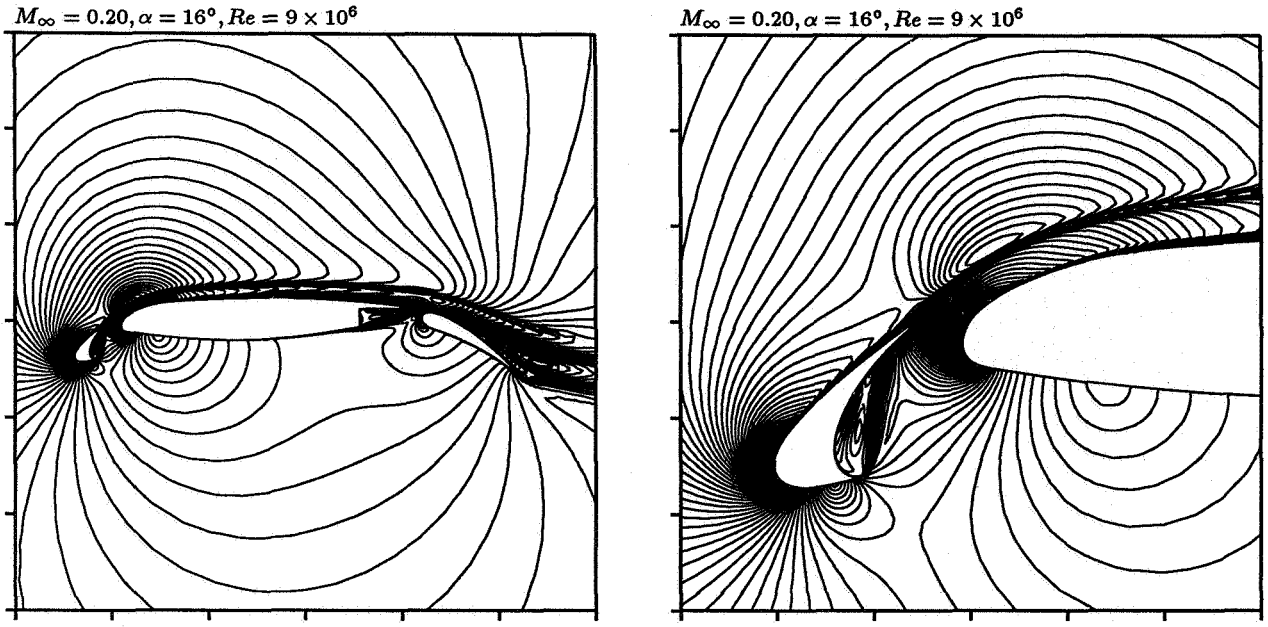

Figure 4: Iso-Mach number contours ($\Delta M = 0.01$) for flow about a Douglas airfoil.

on the other hand, offers the possibility of capturing efficiently many of the flow features in very few iterations.

Figure 5 show the results after two remeshings for flow at $M_\infty = 0.73$, $\alpha = 2.80°$, and $Re = 6.5 \times 10^6$ about the RAE 2822 airfoil. The Baldwin-Barth turbulence model has been used in the computation. The wake centerline has been determined by a streamline trace from the coarse grid solution. The shock-boundary layer interaction region has been well resolved by the two levels of solution adaption based on the local relative pressure change. The calculation predicts slight separation at the base of the shock in agreement with what others have seen for this case with the Baldwin-Barth turbulence model. The surface pressure distribution also shows good agreement with experimental data.

6 Conclusions

A method for generating high quality unstructured triangular grids for high Reynolds number Navier-Stokes calculations about complex geometries has been described. Careful attention has been paid to resolving efficiently the disparate length scales which arise in these problems. By dividing the mesh generation task into two phases, both the viscous and inviscid regions of the flow can be meshed optimally. A solution-adaptive remeshing strategy which allows the mesh to adapt itself to solution features such as wakes and shock waves has also been described. Although at present it has only been implemented in two dimensions, the grid generation process has been designed to be readily extendible to three dimensions. An implicit, higher-order, upwind method has also been presented for computing compressible turbulent flows on these meshes. Two recently developed one-equation turbulence models have been implemented to simulate the effects of the fluid turbulence. High Reynolds number flows about single- and multi-element airfoils have been presented which clearly demonstrate the improved resolution provided by the solution-adaptive remeshing.

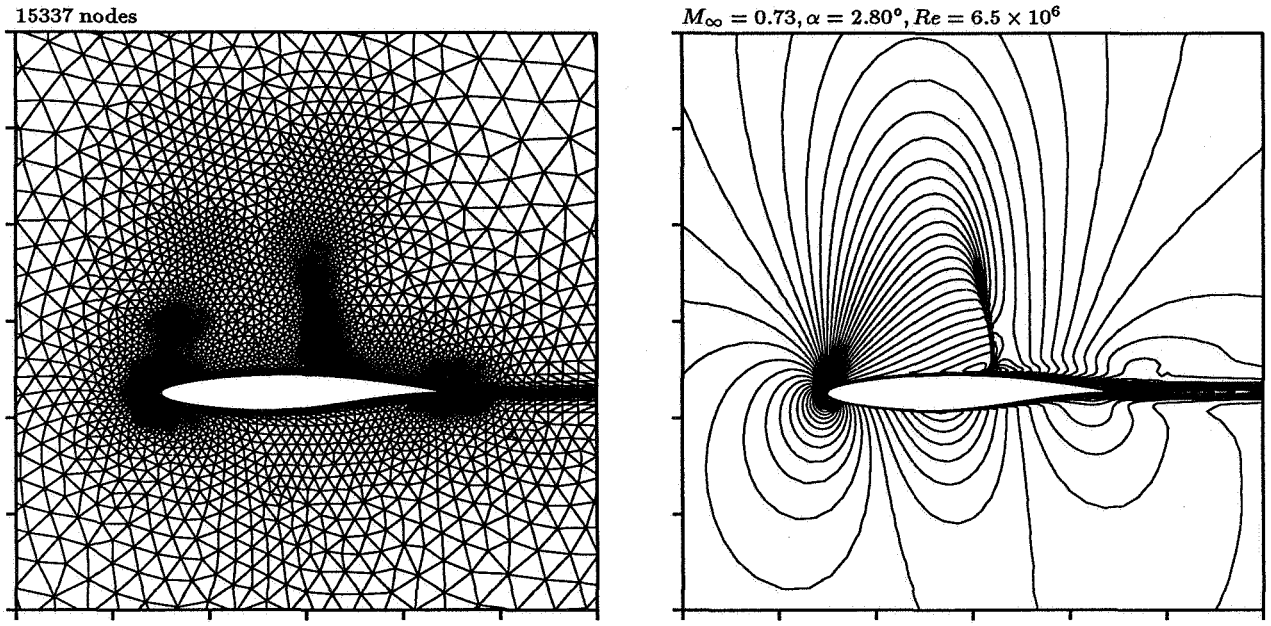

Figure 5: Mesh and Iso-Mach number contours ($\Delta M = 0.02$) about a RAE 2822 after two remeshings.

7 Acknowledgments

This work was funded through an IBM Fellowship in Scientific Computing administered by the Laboratory for Scientific Computation at the University of Michigan.

References

[1] J. Y. Trépanier, M. Reggio, and R. Camarero, "Automated geometric-based mesh requirements for adaptive flow computations," AIAA Paper 93-0674, 1993.

[2] R. Löhner, "Matching semi-structured and unstructured grids for Navier-Stokes calculations," AIAA Paper 93-3348, 1993.

[3] Y. Kallinderis and S. Ward, "Prismatic grid generation for three-dimensional complex geometries," *AIAA Journal*, vol. 31, pp. 1850–6, October 1993.

[4] D. J. Mavriplis, "Adaptive mesh generation for viscous flows using Delaunay triangulation," *Journal of Computational Physics*, vol. 90, pp. 271–291, October 1990.

[5] J.-D. Müller, "Quality estimates and stretched meshes based on Delaunay triangulations," *AIAA Journal*, to appear.

[6] S. Pirzadeh, "Unstructured viscous grid generation by advancing-layers method," AIAA Paper 93-3453, 1993.

[7] S. Pirzadeh, "Viscous unstructured three-dimensional grids by the advancing-layers method," AIAA Paper 94-0417, 1994.

[8] T. J. Barth, "Numerical aspects of computing viscous high Reynolds number flows on unstructured meshes," AIAA Paper 91-0721, 1991.

[9] W. K. Anderson and D. L. Bonhaus, "An implicit upwind algorithm for computing turbulent flows on unstructured grids," *Computers and Fluids*, vol. 23, 1994.

[10] D. J. Mavriplis and L. Martinelli, "Multigrid solution of compressible turbulent flow on unstructured meshes using a two-equation model," AIAA Paper 91-0237, 1991.

[11] B. S. Baldwin and T. J. Barth, "A one-equation turbulence transport model for high Reynolds number wall-bounded flows," NASA TM 102847, 1990.

[12] P. Spalart and S. Allmaras, "A one-equation turbulence model for aerodynamic flows," AIAA Paper 92-0439, 1992.

[13] I. Babuška and A. K. Aziz, "On the angle condition in the Finite Element Method," *SIAM Journal on Numerical Analysis*, vol. 13, no. 2, 1976.

[14] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, "Adaptive remeshing for compressible flow computations," *Journal of Computational Physics*, vol. 72, pp. 449–466, 1987.

[15] A. Bowyer, "Computing Dirichlet tessellations," *The Computer Journal*, vol. 24, no. 2, pp. 162–166, 1981.

[16] D. F. Watson, "Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes," *The Computer Journal*, vol. 24, no. 2, pp. 167–171, 1981.

[17] T. J. Barth, "Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations," in *Computational Fluid Dynamics*, Von Kármán Institute for Fluid Dynamics, Lecture Series 1994-05, 1994.

[18] J. Bonet and J. Peraire, "An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems," *International Journal for Numerical Methods in Engineering*, vol. 31, 1991.

[19] P. L. Roe, "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, vol. 43, 1981.

[20] T. J. Barth and D. C. Jespersen, "The design and application of upwind schemes on unstructured meshes," AIAA Paper 89-0366, 1989.