AIAA-94-0061

Recent Progress Toward a Three-Dimensional Unstructured Navier-Stokes Flow Solver

N. T. Frink
NASA Langley Research Center
Hampton, VA

32nd Aerospace Sciences Meeting
January 10-13, 1994/Reno, Nevada

AIAA 94-0061

Recent Progress Toward a Three-Dimensional Unstructured Navier-Stokes Flow Solver

Neal T. Frink†
NASA-Langley Research Center

Abstract

Progress has been made toward solving three-dimensional viscous flow problems with unstructured grids. Enhancements have been added to an established unstructured grid flow solver that upgrade its formal accuracy to second order, improve the code efficiency and speed, and accurately solve the Navier-Stokes equations on tetrahedral cells. Accuracy is increased by a pseudo-Laplacian weighted averaging algorithm which produces more robust convergence, and permits higher-order boundary conditions to be used, which is important on highly-stretched cells. An existing time-implicit Gauss-Seidel algorithm is installed which results in a six-fold reduction in total computer time over that of an explicit algorithm for inviscid-flow calculations. Finally, an unstructured laminar solution on the ONERA M6 wing has been validated against a structured-grid solution with encouraging results.

Introduction

Unstructured grid methods provide a viable alternative to the structured grid approach for solving the Euler equations for flows over complex aerodynamics shapes. With current grid generation technology,[1-3] isotropic tetrahedral grids can now be constructed around complex aircraft configurations in a matter of days. The construction of a comparable block-structured grid generally requires much more time. As a result, unstructured grid methodology has become widely used throughout industry and government for rapidly solving complex aerodynamic problems which are amenable to inviscid assumptions.

While many practical problems of interest can be addressed with inviscid methodology, the inclusion of viscous effects is critical for the correct analysis of many complex configurations. The numerical solution of the Navier-Stokes equations on tetrahedral elements presents a formidable challenge. However, considerable progress is being made toward achieving such solutions,[4-9] particularly on two-dimensional triangular elements. The primary obstacle to progress in 3D viscous unstructured grid methodology been the absence of a highly stretched unstructured grid generation capability. Advances are being made in that area which may soon produce the much needed capability[10-12]. Pirzadeh[12] recently demonstrated the generation of a 3-D tetrahedral viscous grid on a multielement wing by the Advancing-Layer Method. Once this technology is matured, it will only be a matter of time before viscous unstructured grid solutions become routine.

This paper reports on recent enhancements made to the unstructured grid flow solver described in Refs. 13 to 15 toward solving viscous-flow problems, as well as to provide a general upgrade of the base methodology. The accuracy and utility of the base Euler code has been well documented in Refs. 16 to 21. The subject enhancements were made to address four issues, 1) to improve some peculiar convergence properties encountered periodically through extensive application of the inviscid method to complex configurations, which may potentially inhibit convergence of viscous-flow solutions, 2) to increase the formal accuracy of the method, and 3) to improve code efficiency and speedup the overall convergence rate, and 4) to develop an accurate algorithm for solving the viscous shear fluxes for the Navier-Stokes equations on tetrahedral cells. Toward these goals, a new weighted averaging scheme has been implemented which upgrades the algorithm for reconstructing cell-averaged data to the cell vertices to full second-order accuracy from one that is slightly less accurate. The increased accuracy of the new scheme permits higher-order boundary conditions to be employed. A face-colored data structure has also been implemented resulting in significant improvements in code efficiency over the prior cell-based data structure. The implicit time integration scheme from Ref. 22 has been installed in the code to increase the support within the computational domain during time stepping. The Navier-Stokes terms have been added and validated against structured grid results for laminar attached flow over an ONERA M6 wing.

Governing Equations

The fluid motion is governed by the time dependent Navier-Stokes equations for an ideal gas which express mass, momentum, and energy for a compressible Newtonian fluid in the absence of external forces.

†Senior Research Engineer, Senior member AIAA

The equations are given below in integral form for a bounded domain Ω with a boundary $\partial\Omega$

$$\frac{\partial}{\partial t}\int\int\int_{\Omega}\mathbf{Q}dV+\int\int_{\partial\Omega}\mathbf{F}(\mathbf{Q})\cdot\hat{\mathbf{n}}dS=\int\int_{\partial\Omega}\mathbf{G}(\mathbf{Q})\cdot\hat{\mathbf{n}}dS \tag{1}$$

where

$$\mathbf{Q}=\begin{Bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ e_o\end{Bmatrix}$$

and

$$\mathbf{F}(\mathbf{Q})\cdot\hat{\mathbf{n}}=(\mathbf{V}\cdot\hat{\mathbf{n}})\begin{Bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ e_o+p\end{Bmatrix}+p\begin{Bmatrix}0\\ \hat{n}_x\\ \hat{n}_y\\ \hat{n}_z\\ 0\end{Bmatrix}$$

and

$$\mathbf{G}(\mathbf{Q})\cdot\hat{\mathbf{n}}=\frac{M_\infty}{R_{e_L}}(\hat{n}_x\mathbf{G_1}+\hat{n}_y\mathbf{G_2}+\hat{n}_z\mathbf{G_3})$$

$$\mathbf{G_1}=\begin{Bmatrix}0\\ \tau_{xx}\\ \tau_{xy}\\ \tau_{xz}\\ u\tau_{xx}+v\tau_{xy}+w\tau_{xz}-\dot{q}_x\end{Bmatrix}$$

$$\mathbf{G_2}=\begin{Bmatrix}0\\ \tau_{xy}\\ \tau_{yy}\\ \tau_{yz}\\ u\tau_{xy}+v\tau_{yy}+w\tau_{yz}-\dot{q}_y\end{Bmatrix}$$

$$\mathbf{G_3}=\begin{Bmatrix}0\\ \tau_{xz}\\ \tau_{yz}\\ \tau_{zz}\\ u\tau_{xz}+v\tau_{yz}+w\tau_{zz}-\dot{q}_z\end{Bmatrix}$$

where

$$\tau_{xx}=2\mu u_x-\frac{2}{3}\mu(u_x+v_y+w_z)$$

$$\tau_{yy}=2\mu v_y-\frac{2}{3}\mu(u_x+v_y+w_z)$$

$$\tau_{zz}=2\mu w_z-\frac{2}{3}\mu(u_x+v_y+w_z)$$

$$\tau_{xy}=\mu(u_y+v_x),\quad \tau_{xz}=\mu(u_z+w_x),$$

$$\tau_{yz}=\mu(v_z+w_y)$$

$$\dot{q}_x=-\frac{\mu}{Pr(\gamma-1)}\frac{\partial T}{\partial x},\quad \dot{q}_y=-\frac{\mu}{Pr(\gamma-1)}\frac{\partial T}{\partial y},$$

$$\dot{q}_z=-\frac{\mu}{Pr(\gamma-1)}\frac{\partial T}{\partial z}$$

The equations are nondimensionalized with a free-stream reference values for density ρ_∞ and a speed of sound a_∞. Here $\hat{n}_x, \hat{n}_y$, and $\hat{n}_z$ are the Cartesian components of the exterior surface unit normal $\hat{\mathbf{n}}$ on the boundary $\partial\Omega$. The Cartesian velocity components are u, v, and w in the x, y, and z directions, respectively. The term e_o is the total energy per unit volume. The Prandtl number, Pr, is assigned a value of 0.72. Viscosity, μ, is computed by Sutherland's law. With the ideal gas assumption, the normalized values for pressure and temperature can be expressed as

$$p=(\gamma-1)(e_o-\frac{1}{2}\rho(u^2+v^2+w^2))$$

and

$$T=\gamma p/\rho$$

where γ is the ratio of specific heats and is prescribed as 1.4 for air.

Spatial Discretization

A finite-volume discretization is applied to Eq. 1 which results in a consistent approximation to the conservation laws where the time rate of change of the state vector $\mathbf{Q}$ within the domain Ω is balanced by the net fluxes of $\mathbf{F}$ and $\mathbf{G}$ across the boundary surface $\partial\Omega$. The spatial domain is divided into a finite number of tetrahedral cells, with each element serving as a computational cell. Thus, the discretized solution to Eq. 1 results in a set of volume-averaged state variables $\mathbf{Q}$ which are in balance with the area-averaged fluxes (inviscid and viscous) across the cell faces.

Inviscid Fluxes

The inviscid flux quantities are computed across each cell face using Roe's[23] flux-difference-splitting (FDS). The implementation of this methodology in the present approach is described in Refs. 13 to 15. Roe's FDS technique reconstructs the fluxes by determining an approximate solution to an underlying set of Riemann problems which still describe the important nonlinear behavior of the interacting waves. For that, discontinuous states are assumed to exist on either side of a cell interface. The accuracy with which these states are determined directly impacts the accuracy of the flux computation.

The following two subsections describe the present reconstruction approach that is based on a Taylor series expansion of the cell-averaged solution. A key component of the scheme is the reconstruction of surrounding cell-averaged data to a common vertex by a weighted

averaging procedure. The prior averaging procedure of Ref. 13 has proven to be robust and accurate, but is known to be less than second-order accurate[15] and has periodically exhibited some convergence difficulties. A new averaging procedure is presented below which resolves these issues.

Higher-Order Cell Reconstruction

A simple, universal formula for reconstructing data on tetrahedral cells was introduced in Ref. 14

$$\mathbf{q}_{f_{1,2,3}} = \mathbf{q}_c + 1/4\,[1/3(\mathbf{q}_{n_1} + \mathbf{q}_{n_2} + \mathbf{q}_{n_3}) - \mathbf{q}_{n_4}] \quad (2)$$

where

$$\mathbf{q} \equiv [\rho, u, v, w, p]^T$$

and as illustrated in Fig. 1, the subscripts n_1, n_2, n_3 denote the nodes comprising face $f_{1,2,3}$ of cell c and n_4 corresponds to the opposite node. The derivation of this formula has only appeared in Ref. 15 which received limited distribution, and is thus included in the Appendix of this paper. This formula represents the analytical solution to a Taylor series expansion of the primitive variables from the centroid of a tetrahedral cell to the centroids of its triangular faces. The state at the nodes is assumed to be known and is determined by the reconstruction process described in the next subsection.

Weighted Averaging Procedure

Estimates of the solution are determined at each node by a weighted average of the surrounding cell-centered solution quantities:

$$\mathbf{q}_n = \Big(\sum_{i=1}^{N} \mathrm{w}_{c,i}\mathbf{q}_{c,i}\Big)/\Big(\sum_{i=1}^{N} \mathrm{w}_{c,i}\Big). \quad (3)$$

The subscripts n and c, i refer to the node and surrounding cell-centered values, respectively. The method of Ref. 13 employs an inverse-distance weighting factor, i.e.

$$\mathrm{w}_{c,i} = \frac{1}{r_i} \quad (4)$$

where

$$r_i = [(x_{c,i} - x_n)^2 + (y_{c,i} - y_n)^2 + (z_{c,i} - z_n)^2]^{\frac{1}{2}}.$$

This weighting has been used quite successfully for computing the flow over a number of complex configurations. However, its accuracy is known to be less than second-order.

A fully second-order accurate averaging procedure was presented by Rausch, et. al.[24] for two-dimensional triangular cells which was an extension of work by Holmes[6]. That approach is extended for three-dimensional tetrahedral cells in the following. The method is based on deriving weight factors in Eq. (3) which satisfy the Laplacians

$$\begin{aligned} L(x_n) &= \sum_{i=1}^{N} \mathrm{w}_{c,i}(x_{c,i} - x_n) = 0 \\ L(y_n) &= \sum_{i=1}^{N} \mathrm{w}_{c,i}(y_{c,i} - y_n) = 0 \\ L(z_n) &= \sum_{i=1}^{N} \mathrm{w}_{c,i}(z_{c,i} - z_n) = 0 \end{aligned} \quad (5)$$

This is a desirable property since the Laplacian of a linear function is exactly zero. The weights are determined by defining

$$\mathrm{w}_{c,i} = 1 + \Delta\mathrm{w}_{c,i} \quad (6)$$

where

$$C = \sum_{i=1}^{N} (\Delta\mathrm{w}_{c,i})^2 \quad (7)$$

is a cost function. The cost function is minimized by solving an optimization problem subject to the constraints of Eqs. (5). This optimization problem is solved by the method of Lagrange multipliers were $\Delta\mathrm{w}_{c,i}$ is given by

$$\Delta\mathrm{w}_{c,i} = \lambda_x(x_{c,i} - x_n) + \lambda_y(y_{c,i} - y_n) + \lambda_z(z_{c,i} - z_n). \quad (8)$$

The solution to the constrained optimization problem yields the Lagrange multipliers

$$\begin{aligned} \lambda_x = [&-R_x(I_{yy}I_{zz} - I_{yz}^2) \\ &+R_y(I_{xy}I_{zz} - I_{xz}I_{yz}) \\ &-R_z(I_{xy}I_{yz} - I_{yy}I_{xz})]/D \end{aligned} \quad (9a)$$

$$\begin{aligned} \lambda_y = [&R_x(I_{xy}I_{zz} - I_{xz}I_{yz}) \\ &-R_y(I_{xx}I_{zz} - I_{xz}^2) \\ &+R_z(I_{xx}I_{yz} - I_{xy}I_{xz})]/D \end{aligned} \quad (9b)$$

$$\begin{aligned} \lambda_z = [&-R_x(I_{xy}I_{yz} - I_{yy}I_{xz}) \\ &+R_y(I_{xx}I_{yz} - I_{xy}I_{xz}) \\ &-R_z(I_{xx}I_{yy} - I_{xy}^2)]/D \end{aligned} \quad (9c)$$

where

$$\begin{aligned} D = &I_{xx}(I_{yy}I_{zz} - I_{yz}^2) \\ &-I_{xy}(I_{xy}I_{zz} - I_{xz}I_{yz}) \\ &+I_{xz}(I_{xy}I_{yz} - I_{yy}I_{xz}) \end{aligned}$$

and

$$R_x = \sum_{i=1}^{N}(x_{c,i} - x_n)$$

$$R_y = \sum_{i=1}^{N}(y_{c,i} - y_n)$$

$$R_z = \sum_{i=1}^{N}(z_{c,i} - z_n)$$

$$I_{xx} = \sum_{i=1}^{N}(x_{c,i} - x_n)^2$$

$$I_{yy} = \sum_{i=1}^{N}(y_{c,i} - y_n)^2$$

$$I_{zz} = \sum_{i=1}^{N}(z_{c,i} - z_n)^2$$

$$I_{xy} = \sum_{i=1}^{N}(x_{c,i} - x_n)(y_{c,i} - y_n)$$

$$I_{xz} = \sum_{i=1}^{N}(x_{c,i} - x_n)(z_{c,i} - z_n)$$

$$I_{yz} = \sum_{i=1}^{N}(y_{c,i} - y_n)(z_{c,i} - z_n).$$

These weights are constructed entirely from geometric information. For some severely distorted grids even these 'optimum' weights can differ significantly from unity. Holmes[6] proposed clipping all weights to the range (0,2). The weights were only modified in the present work for the viscous solutions.

Boundary points are treated in the same manner as interior points through the use of ghost cells, described in a subsequent section. In contrast, the scheme of Ref. 13, which employed the inverse-distance weighting of Eq. (4), constructs the state at boundary nodes from the surrounding face-centered boundary conditions and respective distances.

The weighting factor of Eqs. 6 and 8 produces an exact interpolation of the nodal quantity $\mathbf{q}_n$ for a linear variation of $\mathbf{q}_{c,i}$ on an arbitrary tetrahedral grid. The entire procedure of 1) averaging to nodes via. Eqs. 3, 6, and 8, and 2) interpolating to cell face via Eq. (2) has been tested for a linear function on an arbitrary grid. The linear solution is reconstructed to within machine accuracy, both at the vertices and at the cell faces. Thus, for a nonlinear variation of the cell-averaged state, this pseudo-Laplacian weighting procedure is second-order accurate in space.

Viscous Fluxes

The viscous fluxes $\mathbf{G}(\mathbf{Q})$ are approximated at the cell-face centroids by first computing the viscous stresses at the cell vertices, then averaging each component from the three vertices comprising the face. This approach alleviates the need to store the set of nine stress terms for each tetrahedral face.

As is evident in Eq. 1, the viscous stress and heat flux terms are constructed from the Cartesian derivatives of u, v, w, and T. These derivatives are computed directly at the nodes by applying the midpoint trapezoidal rule to the gradient theorem:

$$\nabla\phi_n = \frac{1}{V_\Omega}\oint_{\partial\Omega}\phi\hat{\mathbf{n}}dS. \tag{10}$$

where Ω is the volume of the domain. A compact stencil is formed by prescribing for Ω an interior subset of the tetrahedra surrounding a node as illustrated in Fig. 2 for a 2D simplification. The domain is defined by scaling down the tetrahedra from f to f', where the new face centroid coincides with the centroid of cell c. Because of the invariant distances noted in Fig. 1, the face areas scale in 3D as $S_{f'} = (3/4)^2 S_f$. Similarly, the reduced volume is defined by $V_{c'} = (3/4)^3 V_c$. Applying the midpoint rule to Eq. 10 for the scaled domain results in

$$\begin{aligned}\nabla\phi_n &= \frac{1}{V_{n'}}\sum_{i=1}^{\kappa(n)}\phi_{f',i}\hat{\mathbf{n}}S_{f',i} \\ &= \frac{4}{3}\frac{1}{V_n}\sum_{i=1}^{\kappa(n)}\phi_{c,i}\hat{\mathbf{n}}S_{f,i}\end{aligned} \tag{11}$$

where V_n denotes the accumulated volume of the unscaled tetrahedra $V_{c,i}$ surrounding node n, $\hat{\mathbf{n}}S_{f,i}$ is the directed area of face i on the unscaled tetrahedra, and $\kappa(n)$ are the outer faces f of cells surrounding node n.. The parameter $\phi_{c,i} \equiv \phi_{f',i}$ is the cell-averaged value for ϕ in tetrahedral cell c, i.

Figure 3 illustrates for a 2D example how boundary faces provide the final closure of the domain surrounding a boundary node n_b. For boundary faces encountered in Eq. 11, define $S_f = S_{f,b}$ and

$$\phi_{f',i} = {}^3\!/_4\phi_b + {}^1\!/_4\phi_n$$

where ϕ_b is the face-centered boundary-condition state value of face b and ϕ_n is the value of ϕ at node n as determined from the weighted averaging of Eq. 3.

Improved Data Structure

The FORTRAN computer code reported in Refs. 13 to 15 utilized a cell-based data structure. The flux calculation and residual accumulation were performed

in two steps. The fluxes were first computed and stored by cycling over the cell faces. Then the residuals were accumulated by cycling through the tetrahedra and summing the stored fluxes from the four faces. This process required an additional connectivity array to scatter the information. One advantage of this approach was the ease with which it vectorized on current supercomputers without further special treatment.

A face-based data structure[25] was explored in an attempt to eliminate the need to store the fluxes for each face, and thereby reduce storage requirements. The code was modified to sum the residuals by scattering the fluxes to the two adjacent cells during the flux calculation. This code modification not only eliminated the need to store the face fluxes, but also eliminated an unnecessary connectivity array.

A similar strategy was used to reconstruct the cell-averaged state quantities at the nodes. The previous approach cycled through the nodes and performed a weighted summation of the surrounding cell-centered quantities. The routine was modified to cycle through the faces and scatter the adjacent weighted cell-averaged quantities to the appropriate nodes.

In order for this scheme to vectorize properly, the faces must be reordered into groups or "colors" such that no two faces share the same cell and opposing node. The result of this modification is a 30-percent reduction in memory from 64 words/cell to 45 words/cell and a 16-percent speedup from 21 μs/cell/cycle to 17.5 μs/cell/cycle on a CRAY Y-MP supercomputer for the explicit Euler code.

Time Integration

Explicit

The explicit scheme advances the solution in time using a 3-stage Runge-Kutta time stepping scheme with implicit residual smoothing and local time stepping. Pertinant details are reported in Refs. 13 through 15.

Implicit

Details of the implicit algorithm are described in Ref. 22. The scheme uses the linearized, backward Euler time differencing approach to update the solution at each time step for the set of equations

$$[\mathbf{A}]^n\{\Delta\mathbf{Q}\}^n = \{\mathbf{R}\}^n. \tag{12}$$

where

$$[\mathbf{A}]^n = \frac{V}{\Delta t}\mathbf{I} + \frac{\partial \mathbf{R}^n}{\partial \mathbf{Q}}$$

The linear system of equations are solved at each time step with a subiterative procedure where the tetrahedral cells are grouped into "colors" (different from face-coloring) such that no two cells share a common face. Thus, the solution is computed by solving for all the unknowns in a particular color by a point-Jacobi subiteration step before proceeding to the next color. Since the solution of the unknowns for each group can depend on those from previously computed groups, a Gauss-Seidel-like effect is realized. The method has the advantage of being completely vectorizable.

Because of the number of operations required to invert a matrix depends on the matrix bandwidth, the left-hand side of the system of linear equations is evaluated with first-order differencing to reduce both required storage and computer time. Convergence of the subiterations is further accelerated by using Van Leer's Flux Vector Splitting (FVS) on the left-hand side. Thus in the present study, first-order differencing and FVS are applied to the left-hand side, and higher order differencing and FDS to the right-hand side. The viscous terms are not included in the left-hand side of the equation at this time.

It is necessary to store $[\mathbf{A}]^n$, which is a 5×5 matrix for each cell. Thus, storage requirements are increased from 45 words/cell for the explicit code to 180 words/cell for the implicit code. The code requires 64 μs/cell/cycle on a CRAY Y-MP with 20 subiterations and higher-order differencing. The viscous terms in the right-hand side are absorbed within workspace in the code and do not require additional memory.

Boundary Conditions

The application of the pseudo-Laplacian averaging to boundary nodes requires the implementation of ghost cells. Ghost cells are produced by constructing an image cell across the exterior boundary of an adjacent interior cell. The geometric information is supplied by vector relations

$$\begin{aligned} x_{gh} &= (x_{c,i} - x_{n_b}) - 2Xn_x \\ y_{gh} &= (y_{c,i} - y_{n_b}) - 2Xn_y \\ z_{gh} &= (z_{c,i} - z_{n_b}) - 2Xn_z \end{aligned} \tag{13}$$

where

$$X = (x_{c,i} - x_{n_b})n_x + (y_{c,i} - y_{n_b})n_y + (z_{c,i} - z_{n_b})n_z$$

is a contravariant vector component of distance, and the subscript n_b denotes the a boundary node. These coordinates are used only in the initial generation and storing of the weighting factors and are not used thereafter.

The flow information within the ghost cells must be updated throughout the solution process. For solid boundaries, the flow tangency condition is imposed at

the nodes by constructing an image of the *interior* velocity vector within the adjacent ghost cell. As before, this is accomplished by subtracting twice the contravariant velocity from the adjacent interior cell velocity vector:

$$\begin{aligned} u_{gh} &= u_{c,i} - 2Un_x \\ v_{gh} &= v_{c,i} - 2Un_y \\ w_{gh} &= w_{c,i} - 2Un_z \end{aligned} \tag{14}$$

where

$$U = u_{c,i}n_x + v_{c,i}n_y + w_{c,i}n_z.$$

For viscous, no-slip boundaries, negative values of the cell-centered velocity components are assigned to the corresponding ghost cell. The density and pressure within the ghost cell are set equal to the values in the adjacent interior cell.

The face-centered boundary conditions can be defined by either a low-order or a higher-order approach. The low-order approach involves setting the five primitive variables on the boundary faces to their cell-centered values. The higher-order approach utilizes the Taylor series expansion formula of Eq. (2) to construct a more accurate estimate of the state on the boundary.

Flow tangency is enforced on solid inviscid boundary faces by subtracting the contravariant velocity component normal to the surface. A condition of zero mass and energy flux through the surface is ensured by setting the left and right states of solid boundary faces equal to the boundary conditions prior to computing the fluxes with Roe's approximate Riemann solver. This technique only permits a flux of the pressure terms of the momentum equations through a solid boundary.

The viscous, no-slip boundary conditions take advantage of inherent "structure" in the grid imposed by the Advancing-Layers Method (ALM) of Ref. 12. The ALM produces thin-layer tetrahedral grids with nodes in the "viscous region" distributed along predetermined surface vectors. Thus, as for structured grids, the pressure can be extrapolated to the boundary nodes by a two-point formula. Density is determined by either an adiabatic condition or a prescribed temperature, and velocity at the nodes is explicitly set to zero. Once the nodal quantities are set on the boundary, Eq. 2 is applied to determine the face boundary condition for pressure and density. The velocity is explicitly set to zero on the boundary faces.

Characteristic boundary conditions are applied to the far-field subsonic boundary using the fixed and extrapolated Riemann invariants corresponding to the incoming and outgoing waves. The incoming Riemann invariant is determined from the freestream flow and the outgoing invariant is extrapolated from the interior domain. The invariants are used to determine the locally normal velocity component and speed of sound. At an outflow boundary, the two tangential velocity components and the entropy are extrapolated from the interior, while at an inflow boundary they are specified as having far-field values. These five quantities provide a complete definition of the flow in the far field.

Implicit boundary conditions are employed with the implicit time integration scheme as described in Ref. 22.

Results

Results are presented in this section to 1) demonstrate improved convergence properties of the second-order averaging scheme for a representative "problem" case, 2) assess convergence characteristics of the implicit time integration scheme, and 3) validate the algorithm for solving the Navier-Stokes equations for laminar, attached flow. The inviscid results will focus on convergence characteristics since the code accuracy is well documented in literature[16-21].

Weighted Averaging Scheme

A surface grid is shown in Fig. 4 for the ONERA M6 wing which has a leading edge sweep of 30 degrees, an aspect ratio of 3.8, a taper ratio of 0.56, and symmetrical airfoil sections. The complete grid contains 288,869 cells and 53,791 nodes. An inviscid solution was obtained for $M_\infty = 0.84$ and $\alpha = 3.06°$ with inverse-distance averaging, Eq. (4), using Runge-Kutta time stepping. Convergence was accelerated by local time stepping and implicit residual smoothing with a CFL number of 4. The computational domain is bounded by a rectangular box with boundaries at $-6.5 \le x \le 11.0, 0.0 \le y \le 2.5$, and $-6.5 \le z \le 6.5$ relative to the wing with a semispan of 1. As can be noted in Fig. 5, the L_2-norm enters a limit cycle. This type of convergence behavior does not occur often with Eq. (4), but has surfaced occasionally during wide use of the code. The initial spike at cycle 500 is caused by the code automatically shifting from first-order to higher-order differencing.

A second run was made with the pseudo-Laplacian averaging scheme of Eq. (6) with the same low-order boundary conditions. The result in Fig. 5 is a significant improvement in L_2 convergence and a slight increase in lift coefficient (plotted on an expanded scale). The disturbances in the L_2-norm at cycles 2000 and 4000 are due to initialization problems during restart, and are not characteristic of the algorithm.

A third run is shown in Fig. 5 with Eq. (6) and higher-order boundary conditions. (Attempts to apply higher-order boundary conditions with Eq. (4) were unsuccessful.) There is little difference in the L_2

convergence properties, but a slight difference in lift coefficient. The conclusion is that the second-order pseudo-Laplacian averaging scheme substantially improves the robustness and convergence properties of the flow solver.

Implicit Time-Integration

The implicit time integration scheme is investigated for inviscid flow using the Low-Wing Transport (LWT) configuration shown in Fig. 6 which contained 135,207 cells and 25,400 nodes. A study is performed to determine combinations of CFL number and numbers of subiterations for the Jacobian matrix which produce rapid convergence. Computations are performed at both subsonic and transonic speeds using the the implicit Gauss-Seidel (GS) algorithm and compared with computations reported in Ref. 26 using the explicit three-stage Runge-Kutta (RK) time stepping scheme. Convergence of the RK scheme is accelerated to steady state with implicit residual smoothing, while both RK and GS employ local time stepping.

Subsonic. Figure 7 depicts the effect of CFL number on solution convergence for $M_\infty = 0.498$ and $\alpha = 1.615°$. Convergence to four-orders of magnitude is most rapid for CFL=75. Note that convergence deteriorates for CFL=100 as a result of decreased diagonal dominance of [A] in Eq. (12). Figure 8 shows the effect of the number of subiterations, NSUB, with CFL fixed at 75. The number of global time steps to achieve 4-orders of residual reduction, NCYC, are shown for reference. Note that 20 subiterations produce the most rapid convergence which is consistent with results from Ref. 22.

The convergence history from the implicit scheme with CFL=75 and 20 subiterations is compared with the explicit RK history from Ref. 26 in Figure 9. The solution converges to four orders of residual reduction with 5 to 6 times less computer time for the implicit scheme. The lift coefficient for the implicit result converges more rapidly and has better damping of low frequency oscillations than the explicit scheme. Figure 10 verifies that the surface pressures computed from the implicit algorithm are identical to those from the explicit scheme, which are in reasonable agreement with experimental data from Ref. 27.

Transonic. Similar results are presented for the LWT at the transonic flow conditions $M_\infty = 0.768$ and $\alpha = 1.116°$ in Figs. 11 through 14. Figure 11 shows that CFL=100 yields the most rapid convergence of the L_2-norm. The effect of the number of Jacobian subiterations is shown in Figure 12 where the best value is difficult to determine. However, the plots suggest that 20 subiterations would be a good compromise.

The convergence history for the implicit scheme with CFL=100 and 20 subiterations is compared in Fig. 13 with the explicit RK history from Ref. 26. The implicit solution converges in over 6-times less computer time than the explicit scheme. Figure 14 again confirms that the two solutions yield identical surface pressure distributions. The poor agreement in Fig. 14 between the computed solutions and the experimental data from Ref. 27 is unfortunately typical of that produced by inviscid methods on supercritical airfoils. At the test Reynolds number of 2.5 x 10^6, the boundary layer has a pronounced effect of decreasing the aft-camber which dramatically alters the circulation and resulting shock location.[28]

Laminar Viscous Solution

The viscous unstructured algorithm is validated for laminar attached flow over the ONERA M6 wing by comparing with structured-grid solutions from CFL3D[29] on a common base grid. A tetrahedral viscous grid was constructed by subdividing a structured $97 \times 41 \times 37$ O-O grid (chord-radial-span) into tetrahedral cells (Fig. 15). Thus, each hexahedral cell is divided into 6 tetrahedra, producing an unstructured grid with 829,440 cells and 143,705 nodes. A normal grid spacing of 1.5×10^{-5} root chord was prescribed at the surface and stretched geometrically to yield 11 to 12 points in the mid-chord laminar boundary layer for a Reynolds number of 1×10^6

Dual computations were performed with the unstructured and structured codes for $M_\infty = 0.5$, $\alpha = 3°$, and $R_e = 1 \times 10^6$. The CFL numbers were ramped linearly from 0.1 to 5 over the first 1000 cycles, and maintained at 5 for the remaining cycles. The unstructured algorithm used 20 subiterations of the Jacobian matrix. Figure 16 compares the structured and unstructured surface grid, upper surface pressure contours, and normalized velocity vectors. Reasonable agreement is noted in the flow parameters. The normalized velocity vectors were taken from the first node off the surface and reflect the directionality of the flow. An aberration in flow direction is present near the apex for the unstructured case which will be addressed in future work.

A comparison of skin-friction coefficients is shown in Fig. 17 at 0.44 and 0.90 semispan which shows good agreement between unstructured and structured solutions. The differences in resolution at the leading edge may be due to the higher spatial resolution of the tetrahedral grid. However, a grid sensitivity study is needed before drawing firm conclusions. A comparison of pressure coefficients at the same span stations is shown in Fig. 18 with excellent agreement. Some anomilies are present in the leading-edge recompression region of C_p

at $\eta = 0.90$ in Fig. 18, and in the unstructured skin-friction coefficients near the trailing edge in Fig. 17. Work is underway to investigate the source of these features.

A comparison of convergence histories is shown in Fig. 18. The lift coefficient converges to 0.210 for the unstructured algorithm and 0.202 for CFL3D. Peculiar convergence behavior is noted for the unstructured solution after 3500 iterations which needs further investigation. The rate of convergence may improve with the inclusion of the viscous terms in $[\mathbf{A}]^n$ of Eq. (12).

Concluding Remarks

Several improvements to an established three-dimensional unstructured grid flow solver have been reported which result in a more efficient and robust flow analysis tool with viscous-flow capability. A new weighted averaging algorithm is developed which increases the formal accuracy of the base method to second-order and leads to improved inviscid solution convergence for complex configurations. A new data structured based on tetrahedral face coloring has been implemented within the code, resulting in a 30-percent reduction in memory usage and a 16-percent reduction in required computer time per cycle for the time-explicit option of the code. An existing time-implicit Gauss-Seidel algorithm has been installed which results in a factor 6 decrease in total computer time needed to obtain a steady state solution, with a factor of four increase in memory. Finally, a 3D unstructured laminar solution on the ONERA M6 wing has been validated against a structured-grid solution.

Appendix - Derivation of Extrapolation Formula

In the following, a universal expression for the Taylor series expansion within a triangular cell will be derived. A similar derivation could be carried out for a tetrahedral cell. The resulting expression requires knowledge of the state at the vertices or nodes, which is obtained by a weighted average from the state at the centers of the cells surrounding the node.

The Taylor series expansion within a cell can be written as

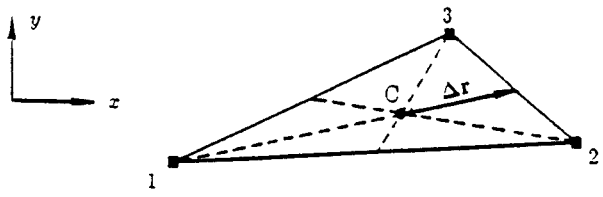
$$\begin{aligned}\mathbf{q}(x,y,z) &= \mathbf{q}(x_c,y_c,z_c) + \nabla\mathbf{q}_c \cdot \Delta\mathbf{r} \quad + \quad O(\Delta r^2) \\ &\approx \mathbf{q}(x_c,y_c,z_c) + \mathbf{q}_x|_c \cdot (x - x_c) \\ &\qquad + \mathbf{q}_y|_c \cdot (y - y_c).\end{aligned} \tag{A.1}$$

The area-averaged solution gradient is estimated by evaluating the closed integral over the boundary $\partial\Omega$ enclosing the domain Ω

$$\nabla\mathbf{q}_c = \frac{1}{A_\Omega}\oint_{\partial\Omega} \mathbf{q}\hat{\mathbf{n}}dl. \tag{A.2}$$

In two-dimensions, Eq. (A.2) represents a line integral around some closed path surrounding an area. In three-dimensions, it represents a surface integral over a surface enclosing a volume. The unit normal $\hat{\mathbf{n}}$ is assumed to point outward from the domain.

The components of the gradient in Eq. (A.1) can be computed as in Ref. 13 by coalescing the surrounding cell-averaged information to the vertices of the candidate cell, and then approximating the solution to Eq. (A.2) with the midpoint trapezoidal rule. Consider the arbitrary triangular cell in sketch a.



Sketch a

The goal is to expand the solution from the cell centroid to the centroids of the edges, e.g. to edge 2-3. Note that the vector $\Delta\mathbf{r}$ is defined along the line passing from node 1 through the cell centroid to the centroid of the opposite edge 2-3. When the midpoint trapezoidal rule is applied to the integral in Eq. (A.2) around the perimeter of the triangle, then the x-component is written as

$$\begin{aligned}\mathbf{q}_x|_c &= \frac{1}{2A}[(\mathbf{q}_{n_1} + \mathbf{q}_{n_2})n_{x_{12}}l_{12} \\ &\quad +(\mathbf{q}_{n_2} + \mathbf{q}_{n_3})n_{x_{23}}l_{23} \\ &\quad +(\mathbf{q}_{n_3} + \mathbf{q}_{n_1})n_{x_{31}}l_{31}] \\ &= \frac{1}{A}[\langle\mathbf{q}_{12}\rangle\Delta y_{12} \\ &\quad +\langle\mathbf{q}_{23}\rangle\Delta y_{23} \\ &\quad +\langle\mathbf{q}_{31}\rangle\Delta y_{31}]\end{aligned} \tag{A.3}$$

where A is the area of the triangle, l_{12}, l_{23}, l_{31} are edge lengths and $n_{x_{12}}$, $n_{x_{23}}$, $n_{x_{31}}$ the x-component of the direction cosines for edges 1-2, 2-3, and 3-1,

$$\begin{aligned}\langle\mathbf{q}_{12}\rangle &= \tfrac{1}{2}(\mathbf{q}_{n_1} + \mathbf{q}_{n_2}) \\ \langle\mathbf{q}_{23}\rangle &= \tfrac{1}{2}(\mathbf{q}_{n_2} + \mathbf{q}_{n_3}) \\ \langle\mathbf{q}_{31}\rangle &= \tfrac{1}{2}(\mathbf{q}_{n_3} + \mathbf{q}_{n_1})\end{aligned}$$

and

$$\Delta y_{12} = y_2 - y_1, \quad \Delta y_{23} = y_3 - y_2, \quad \Delta y_{31} = y_1 - y_3.$$

Similarly, the y-component can be written as

$$\mathbf{q}_y = \frac{-1}{A}[\langle\mathbf{q}_{12}\rangle\Delta x_{12} + \langle\mathbf{q}_{23}\rangle\Delta x_{23} + \langle\mathbf{q}_{31}\rangle\Delta x_{31}] \tag{A.4}$$

where

$$\Delta x_{12} = x_2 - x_1, \quad \Delta x_{23} = x_3 - x_2, \quad \Delta x_{31} = x_1 - x_3.$$

As shown in sketch a,

$$\begin{aligned}\Delta \mathbf{r} &= 1/3\{[1/2(x_2 + x_3) - x_1]\hat{\mathbf{i}} + [1/2(y_2 + y_3) - y_1]\hat{\mathbf{j}}\} \\ &= 1/6[(\Delta x_{12} - \Delta x_{31})\hat{\mathbf{i}} + (\Delta y_{12} - \Delta y_{31})\hat{\mathbf{j}}].\end{aligned} \tag{A.5}$$

Equation (A.5) applies to any arbitrary triangle due to the invariant features that 1) a line extending from a vertex through the cell centroid will always intersect the centroid of the opposing edge, and 2) the distance from the vertex to the centroid of the triangle is always two-thirds of that from the vertex to the opposing edge. Since $\nabla \mathbf{q} = \mathbf{q}_x\hat{\mathbf{i}} + \mathbf{q}_y\hat{\mathbf{j}}$, then the first order term in Eq. (A.1) becomes

$$\begin{aligned}\nabla \mathbf{q}_c \cdot \Delta \mathbf{r} = \frac{1}{6A}[&(\langle q_{12}\rangle \Delta y_{12} + \langle q_{23}\rangle \Delta y_{23} \\ &+ \langle q_{31}\rangle \Delta y_{31})(\Delta x_{12} - \Delta x_{31}) \\ &-(\langle q_{12}\rangle \Delta x_{12} + \langle q_{23}\rangle \Delta x_{23} \\ &+ \langle q_{31}\rangle \Delta x_{31})(\Delta y_{12} - \Delta y_{31})] \qquad (A.6) \\ = \frac{1}{6A}[&\langle q_{23}\rangle(\Delta x_{12}\Delta y_{23} - \Delta x_{23}\Delta y_{12}) \\ &+ \langle q_{23}\rangle(\Delta x_{23}\Delta y_{31} - \Delta x_{31}\Delta y_{23}) \\ &+ (\langle q_{12}\rangle + \langle q_{31}\rangle)(\Delta x_{12}\Delta y_{31} - \Delta x_{31}\Delta y_{12})].\end{aligned}$$

By defining the edges as vectors

$$\mathbf{L}_{12} = \Delta x_{12}\hat{\mathbf{i}} + \Delta y_{12}\hat{\mathbf{j}}$$

$$\mathbf{L}_{23} = \Delta x_{23}\hat{\mathbf{i}} + \Delta y_{23}\hat{\mathbf{j}}$$

$$\mathbf{L}_{31} = \Delta x_{31}\hat{\mathbf{i}} + \Delta y_{31}\hat{\mathbf{j}}$$

then area of the triangle can be computed as

$$\begin{aligned}2A &= |\mathbf{L}_{12} \times \mathbf{L}_{23}| = \Delta x_{12}\Delta y_{23} - \Delta x_{23}\Delta y_{12} \\ &= |\mathbf{L}_{23} \times \mathbf{L}_{31}| = \Delta x_{23}\Delta y_{31} - \Delta x_{31}\Delta y_{23} \qquad (A.7) \\ &= |\mathbf{L}_{31} \times \mathbf{L}_{12}| = \Delta x_{31}\Delta y_{12} - \Delta x_{12}\Delta y_{31}.\end{aligned}$$

Substitute Eq. (A.7) into Eq. (A.6) yields

$$\begin{aligned}\nabla \mathbf{q} \cdot \Delta \mathbf{r} &= 1/3[2\langle q_{23}\rangle - (\langle q_{12}\rangle + \langle q_{31}\rangle)] \\ &= 1/3[1/2(q_{n_2} + q_{n_3}) - q_{n_1}].\end{aligned} \tag{A.8}$$

Substituting Eq. (A.8) into Eq. (A.1) gives the simple relation

$$q_{e_{2,3}} = q_c + 1/3\,[1/2(q_{n_2} + q_{n_3}) - q_{n_1}]. \tag{A.9}$$

A similar derivation in three-dimensions on a tetrahedral cell yields the expression

$$q_{f_{1,2,3}} = q_c + 1/4\,[1/3(q_{n_1} + q_{n_2} + q_{n_3}) - q_{n_4}]. \tag{A.10}$$

Thus, the state at the edge or face centroid can be readily determined by Eq. (A.9) in two dimensions and Eq. (A.10) in three dimensions, respectively. Recall that the edge or face centroid locations are precisely where the averaged fluxes are evaluated in the finite-volume formulation.

The Eqs. (A.9) and (A.10) are formally second-order accurate. This becomes evident by observing that the Taylor series in Eq. (A.1) is second-order accurate and that the midpoint trapezoidal rule used to evaluate the integral in Eq. (A.2) is also second-order accurate. A second-order accurate nodal averaging procedure is presented in the main body of this paper.

Acknowledgements

The author would like to express appreciation to Dr. Kyle Anderson, NASA Langley Research Center, for providing subroutines and expertise for installing his implicit time integration scheme into the flow solver. Thanks is also extended to Dr. Tom Roberts, NASA Langley Research Center for providing the O-O grid used in the ONERA M6 wing viscous calculation, and to Dr. Paresh Parikh, ViGYAN, Inc for the inviscid M6 grid. Appreciation is expressed to Dr. David Halt of McDonnell Douglas Research Laboratories for bringing to the authors attention in the literature the second-order averaging scheme of Rausch et. al, and for suggesting the change in code data structure. The computations were performed on the CRAY Y-MP at NASA Langley Research Center, and on the Numerical Aerodynamic Simulation (NAS) facility at NASA Ames Research Center.

References

[1] Parikh, P., Pirzadeh, S., and Löhner, R., "A Package for 3-D Unstructured Grid Generation, Finite-Element Flow Solutions, and Flow-Field Visualization", NASA CR-182090, September 1990.

[2] Pirzadeh, S.,"Structured Background Grids for Generation of Unstructured Grids by Advancing Front Method", *AIAA Journal*, Vol. 31, No. 2, February 1993, pp. 257-265.

[3] Pirzadeh, S.,"Recent Progress in Unstructured Grid Generation ", AIAA Paper 92-0445, January 1992.

[4] Anderson, W. K., and Bonhaus, D. L.: "An implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids", *Computers Fluids*, Vol. 23, No. 1, pp. 1-21, 1994.

[5] Nakahashi, N., "FDM-FEM Zonal Approach for Viscous Flow computations Over Multiple Bodies", AIAA Paper 87-0604, January, 1987.

[6] Holmes, D. G., and Connell, S. D., "Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Grids", presented at the *AIAA 9th Computational Fluid Dynamics Conference*, June, 1989.

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

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

[9] Marcum, D. L.; and Agarwal, R. K.: "A Three-Dimensional Finite Element Navier-Stokes Solver with k-ϵ Turbulence Model for Unstructured Grids." AIAA Paper 90-1652, 1990.

[10] Ward, S., and Kallinderis, Y.: "Hybrid Prismatic/ Tetrahedral Grid Generation for Complex 3-D Geometries", AIAA Paper No. 93-0669, January 1993.

[11] Pirzadeh, S.: "Unstructured Viscous Grid Generation by Advancing-Layers Method", AIAA Paper No. 93-3453, August, 1993.

[12] Pirzadeh, S.: "Viscous Unstructured Three-Dimensional Grids by the Advancing-Layers Method", AIAA Paper No. 94-0417, January 1994.

[13] Frink, N. T.: "Upwind Scheme for Solving the Euler Equations on Unstructured Tetrahedral Meshes." *AIAA Journal*, Vol., No. 1, January, 1992, pp. 70-77.

[14] Frink, N. T., Parikh, P., and Pirzadeh, S., "A Fast Upwind Solver for the Euler Equations on Three-Dimensional Unstructured Meshes", AIAA 91-0102, Jan. 1991.

[15] Frink, N. T.,"Three-Dimensional Upwind Scheme for Solving the Euler Equations on Unstructured Tetrahedral Grids",Ph. D. Dissertation, Virginia Polytechnic Institute and State University, September 1991.

[16] Dodbele, S. S.: "Three Dimensional Aerodynamic Analysis of a High-Lift Transport Configuration", Paper No. AIAA 93-3536, August, 1993.

[17] Fouladi, K.: "Unstructured Grids for Sonic-Boom Analysis", AIAA Paper No. 93-2929, July, 1993.

[18] Kwon, O. J., and Hah, C.: "Three-Dimensional Unstructured Grid Euler Method Applied to Turbine Blades", AIAA Paper No. 93-0196, January, 1993.

[19] Parikh, P., Pirzadeh, S., and Frink, N. T.: "Unstructured Grid Solutions to a Wing/Pylon/Store Configuration Using VGRID3D/USM3D", AIAA Paper No. 92-4572, August, 1992.

[20] Potsdam, M. A., Intemann, G. A., Frink, N. T., Campbell, R. L., Smith, L. A., and Pirzadeh, S.: "Wing/Pylon Fillet Design Using Unstructured Mesh Euler Solvers", AIAA Paper No. 93-3500, August, 1993.

[21] Ghaffari, F.: "On The Vortical-Flow Prediction Capability Of An Unstructured Grid Euler Solver", AIAA Paper No. 94-0163, January, 1994.

[22] Anderson, W. K.: "Grid Generation and Flow Solution Method for Euler Equations on Unstructured Grids", NASA TM-4295, April 1992.

[23] Roe, P. L.: Characteristic Based Schemes for the Euler Equations. *Annual Review of Fluid Mechanics*, Vol. 18, 1986, pp. 337-365.

[24] Rausch, R. D., Batina, J. T., and Yang, H. T. Y., "Spatial Adaption Procedures on Unstructured Meshes For Accurate Unsteady Aerodynamic Flow Computation." AIAA Paper 91-1106, April, 1991.

[25] Jameson, A. and Baker, T. J.: "Improvements to the Aircraft Euler Method", AIAA Paper 87-0452, January 1987.

[26] Frink, N. T., Parikh, P., and Pirzadeh, S., "Aerodynamic Analysis of Complex Configurations using Unstructured Grids", AIAA Paper 91-3292, September 1991.

[27] Pendergraft, O. C.; Re, R. J.; and Kariya, T. T.: "Nacelle/Pylon Interference Study on a 1/17th-Scale, Twin-Engine, Low-Wing Transport Model", AIAA Paper No. 89-2480, July 1989.

[28] Melson, N. D. and Streett, C. L.: "Updated User's Guide for TAWFIVE With Multigrid", NASA TM-4109, 1989.

[29] Thomas, J. L., Taylor, S. L., and Anderson, W. K.: "Navier-Stokes Computations of Vortical Flows Over Low Aspect Ratio Wings," AIAA Paper No. 87-0207, 1987.

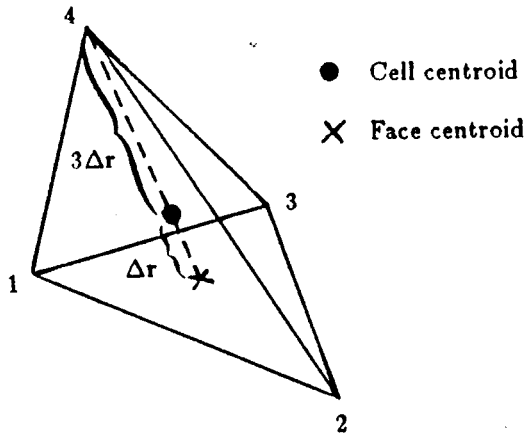


Figure 1.- Geometrically invariant features of tetrahedra.

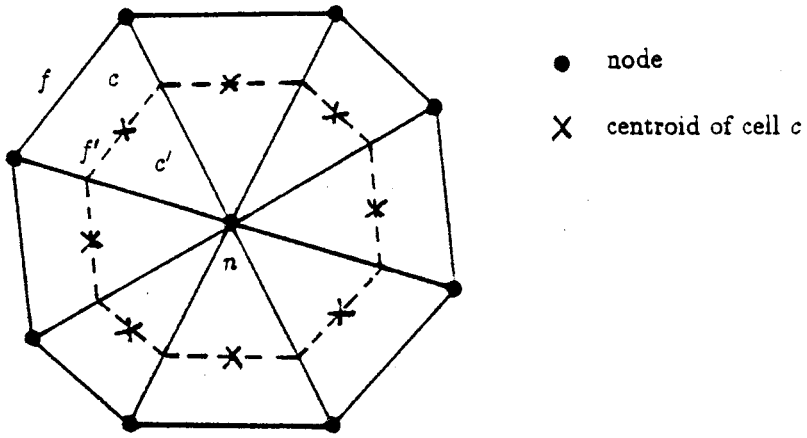


Figure 2.- Stencil for computing viscous stresses at interior nodes (2D simplification).

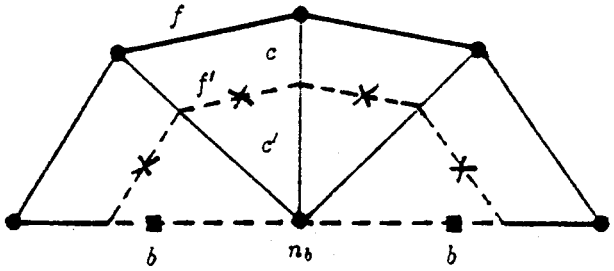


Figure 3.- Stencil for computing viscous stresses at boundary nodes (2D simplification).

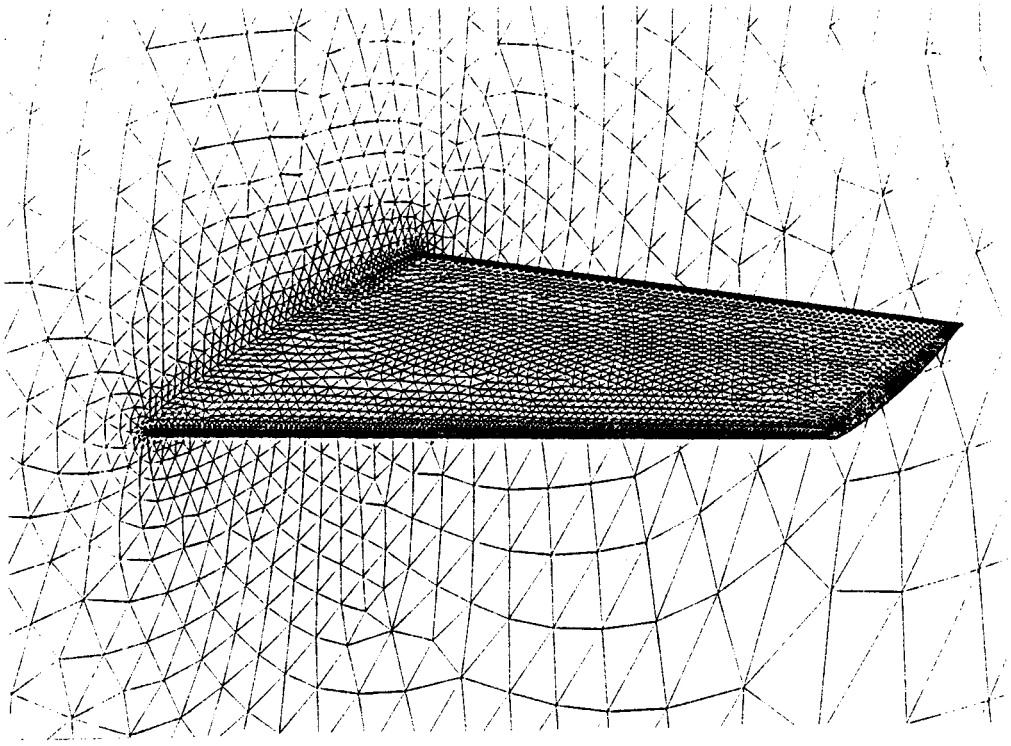
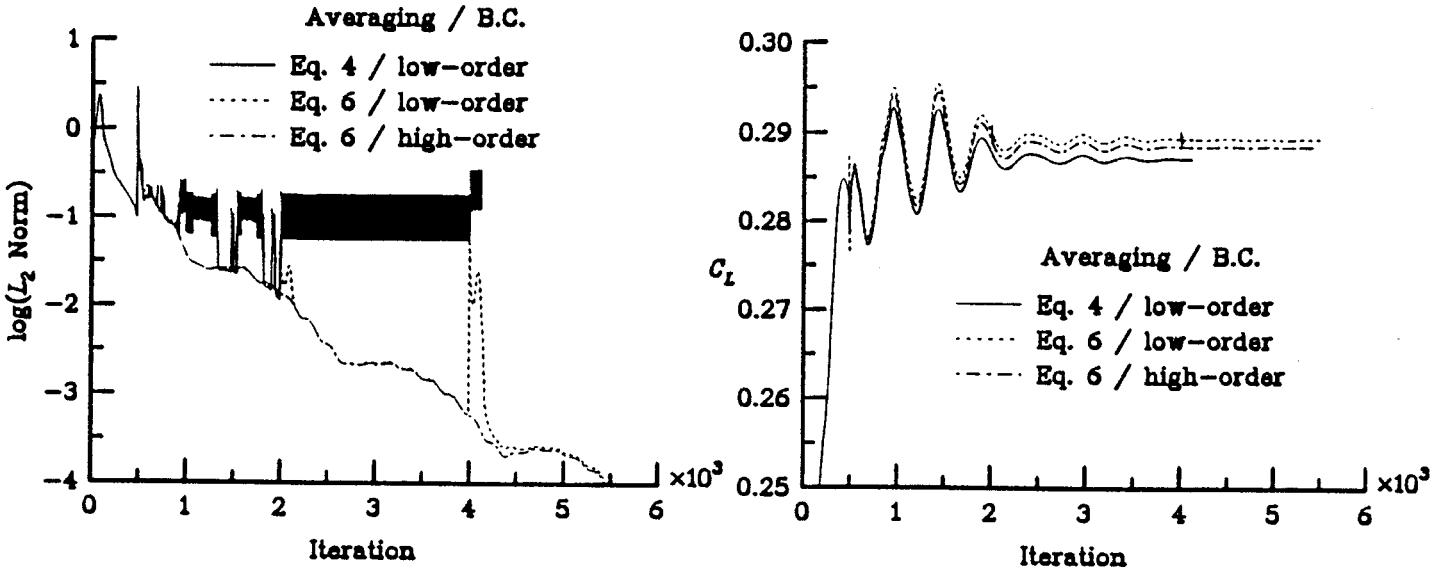


Figure 4.- Surface mesh for ONERA M6 wing.
288,869 cells, 53,791 nodes.



(a) L_2-norm

(b) Lift coefficient

Figure 5.- Effect of weighted-averaging scheme and boundary condition on convergence history for ONERA M6 wing, $M_\infty = 0.84$, $\alpha = 3.06°$.

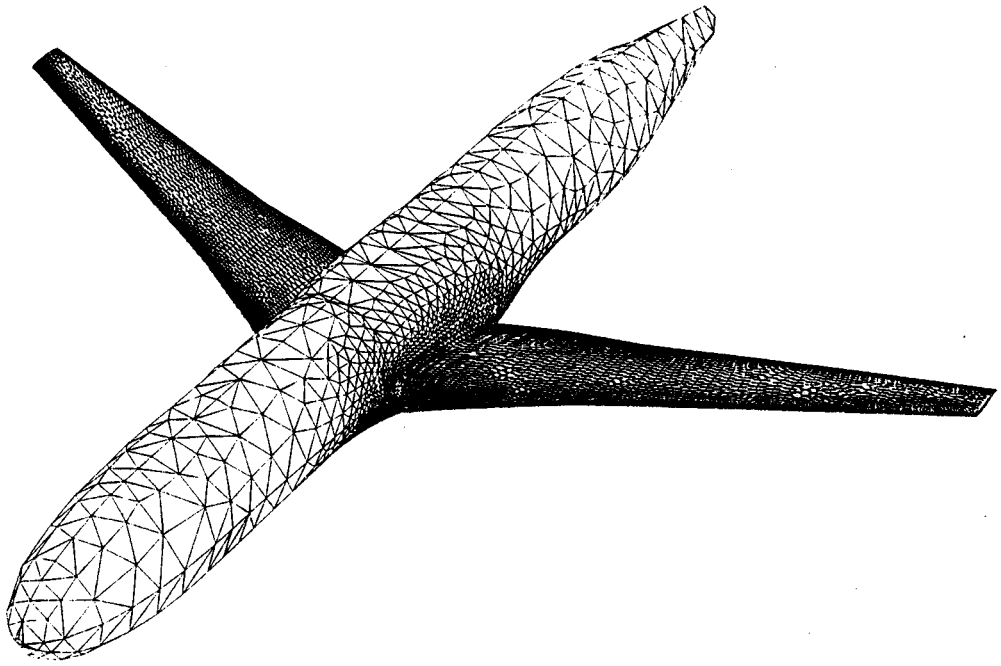
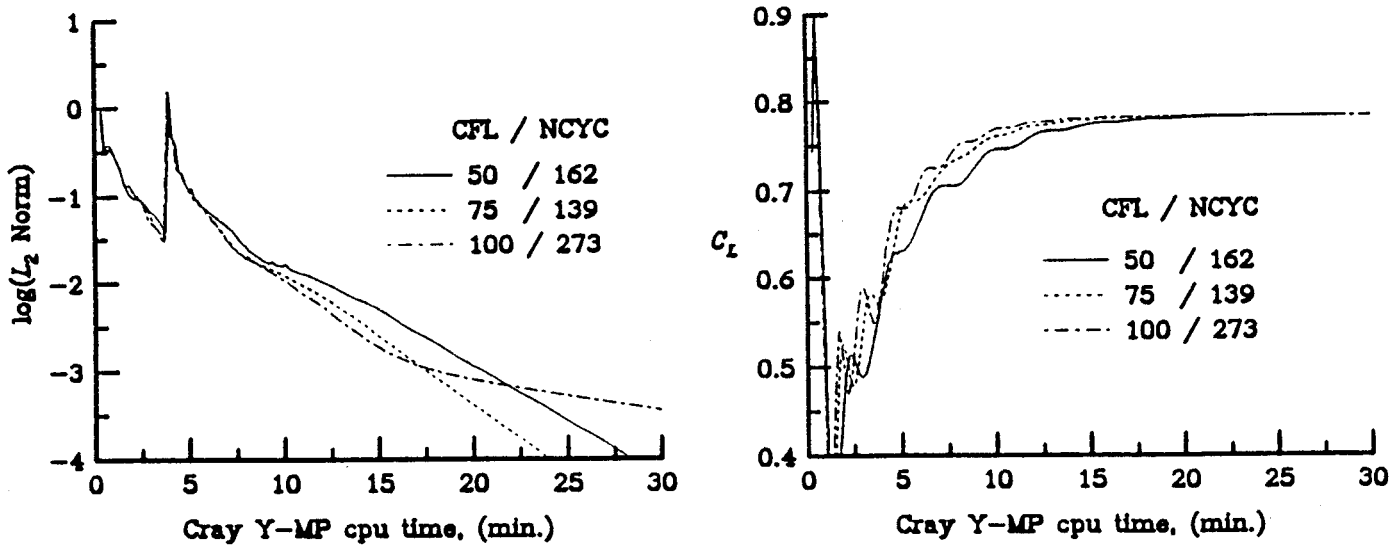


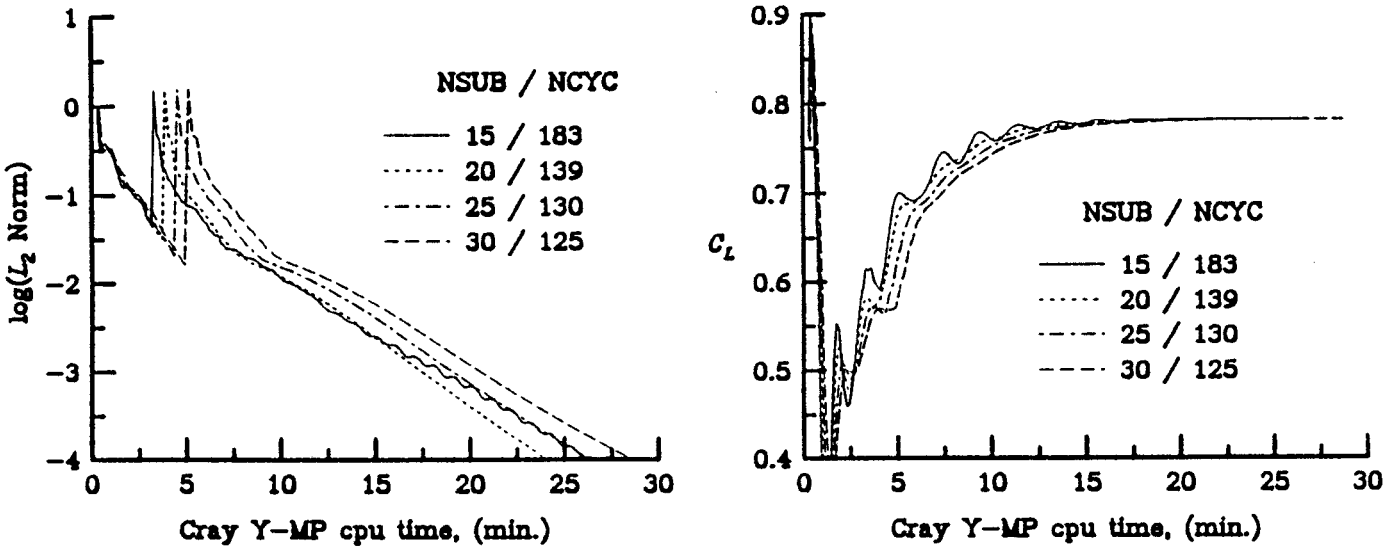
Figure 6.- Surface grid for Low-Wing Transport (LWT) configuration.
135207 cells, 25400 nodes



(a) L_2-norm

(b) Lift coefficient

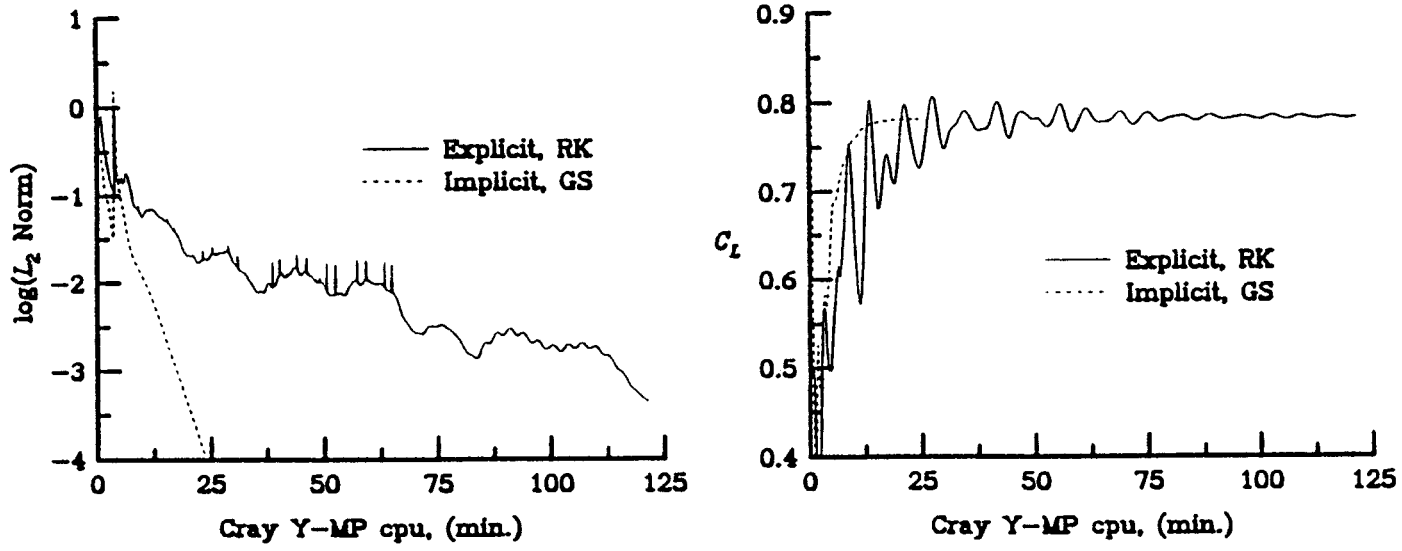
Figure 7.- Effect of CFL number on solution convergence history for LWT.
20 subiterations, $M_\infty = 0.5, \alpha = 1.62°$.



(a) L_2-norm

(b) Lift coefficient

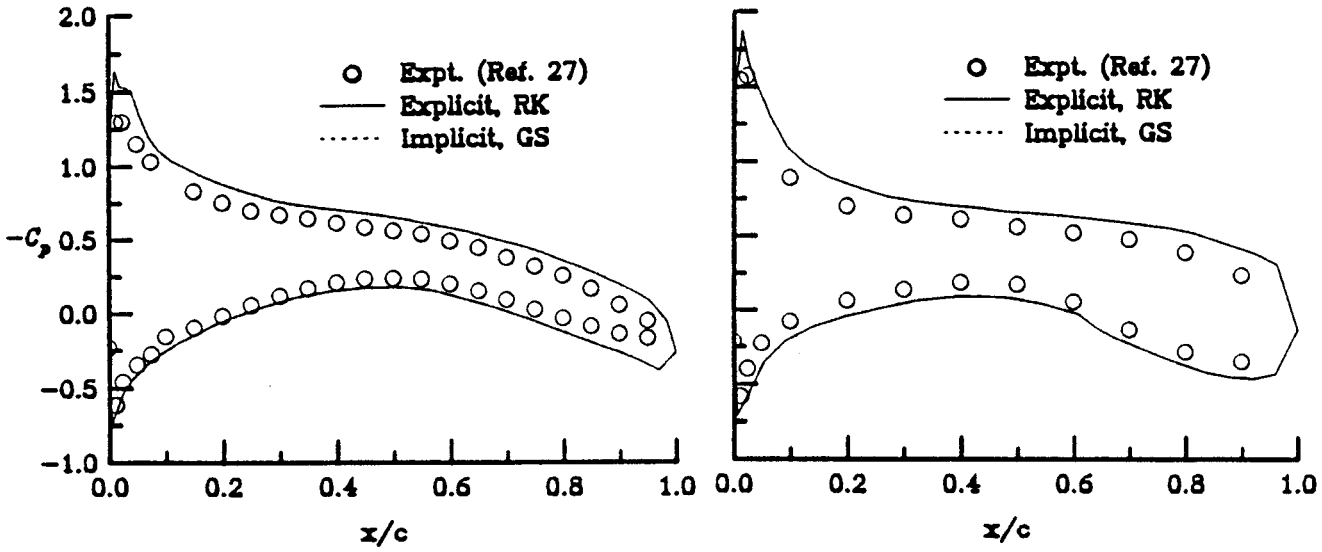
Figure 8.- Effect of Jacobian subiterations on convergence history for LWT.
CFL=75, $M_\infty = 0.5, \alpha = 1.62°$.



(a) L_2-norm

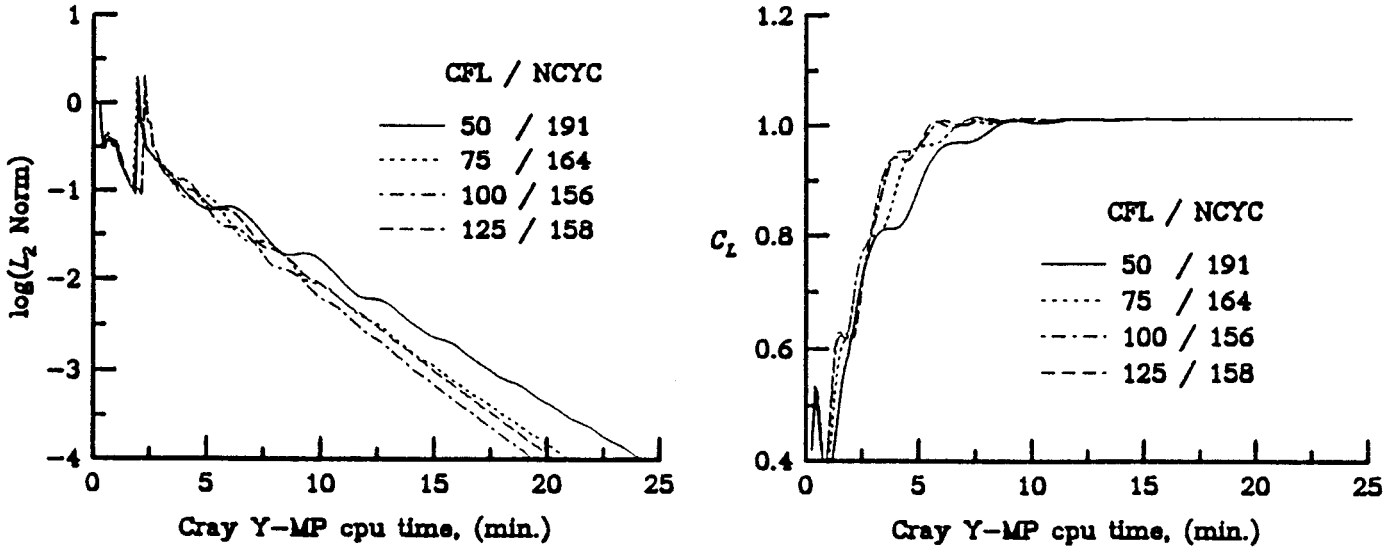
(b) Lift coefficient

Figure 9.- Effect of time-integration scheme on solution convergence history for LWT.
$M_\infty = 0.5, \alpha = 1.62°$.



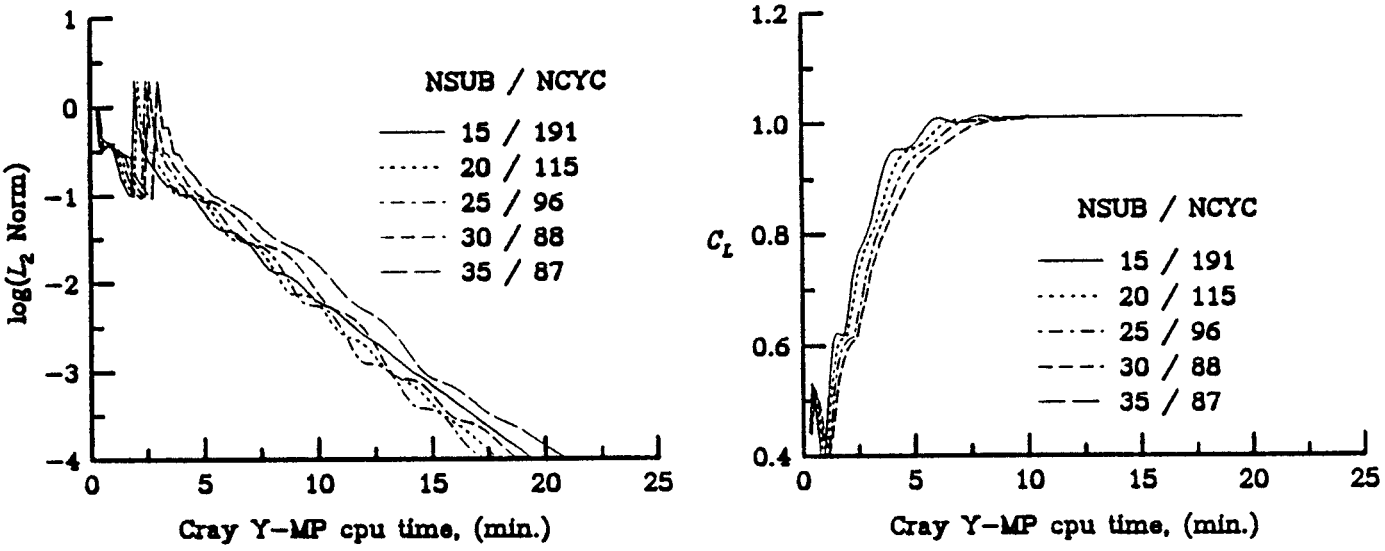
(a) $\eta = 0.277$ (b) $\eta = 0.550$

Figure 10. Effect of time-integration scheme on chordwise C_p distribution for LWT. $M_\infty = 0.5, \alpha = 1.62°$.



(a) L_2-norm (b) Lift coefficient

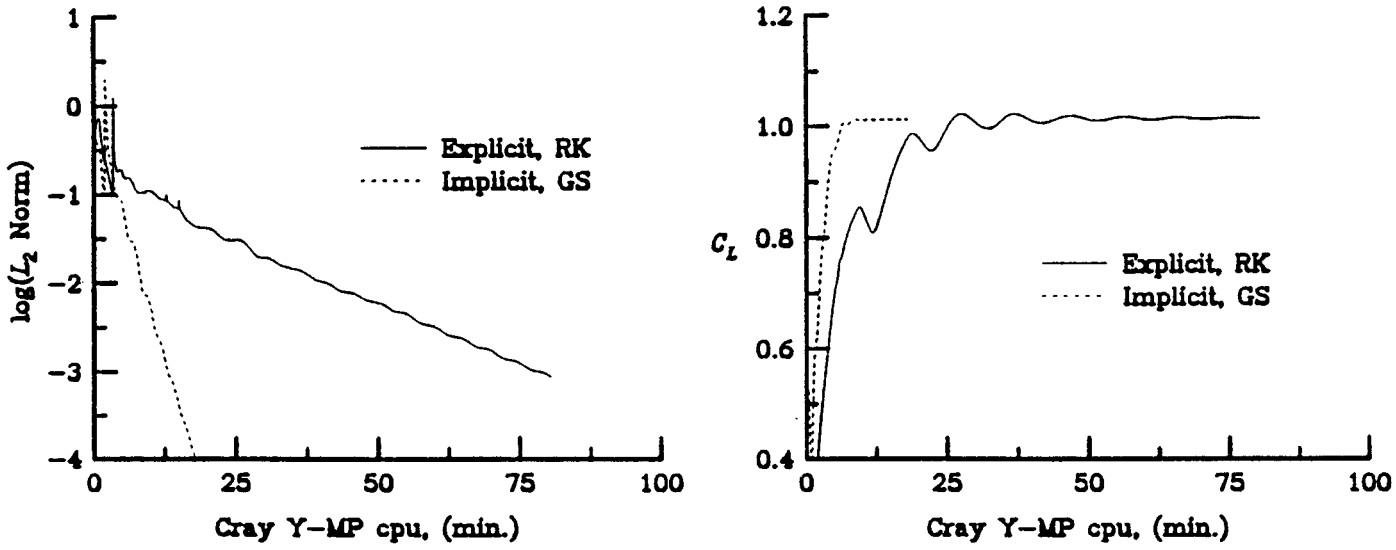
Figure 11.- Effect of CFL number on solution convergence history for LWT. 15 subiterations, $M_\infty = 0.77, \alpha = 1.12°$.



(a) L_2-norm

(b) Lift coefficient

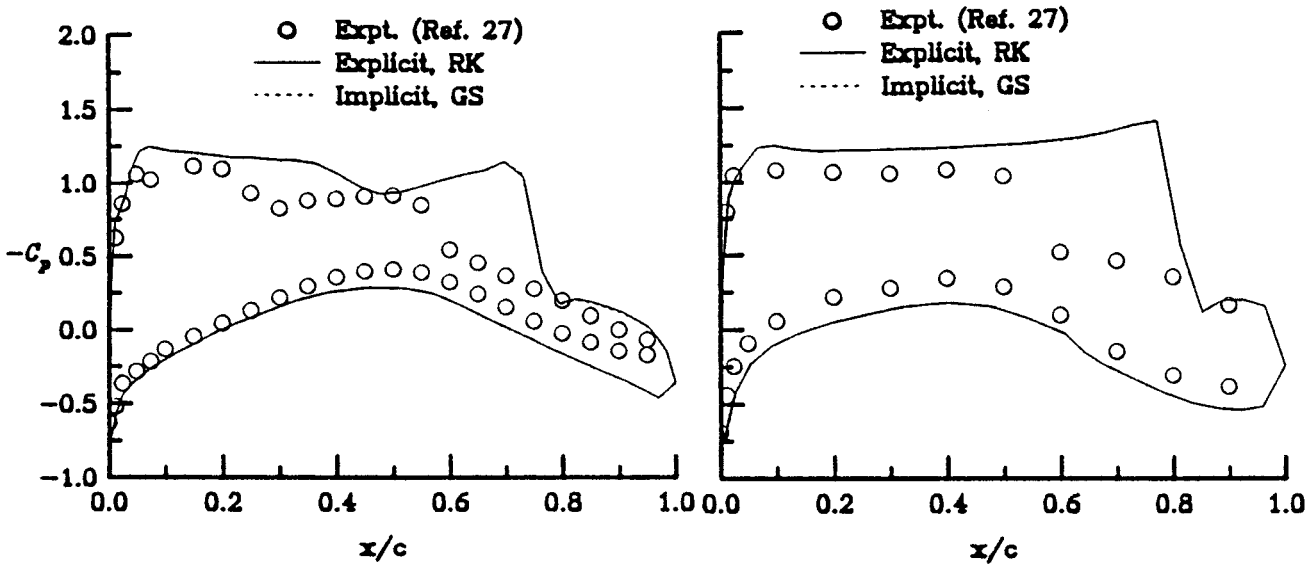
Figure 12.- Effect of Jacobian subiterations on convergence history for LWT.
CFL=100, $M_\infty = 0.77, \alpha = 1.12°$.



(a) L_2-norm

(b) Lift coefficient

Figure 13.- Effect of time-integration scheme on solution convergence history for LWT.
$M_\infty = 0.77, \alpha = 1.12°$.



(a) $\eta = 0.277$ (b) $\eta = 0.550$

Figure 14. Effect of time-integration scheme on chordwise C_p distribution for LWT. $M_\infty = 0.77, \alpha = 1.12°$.

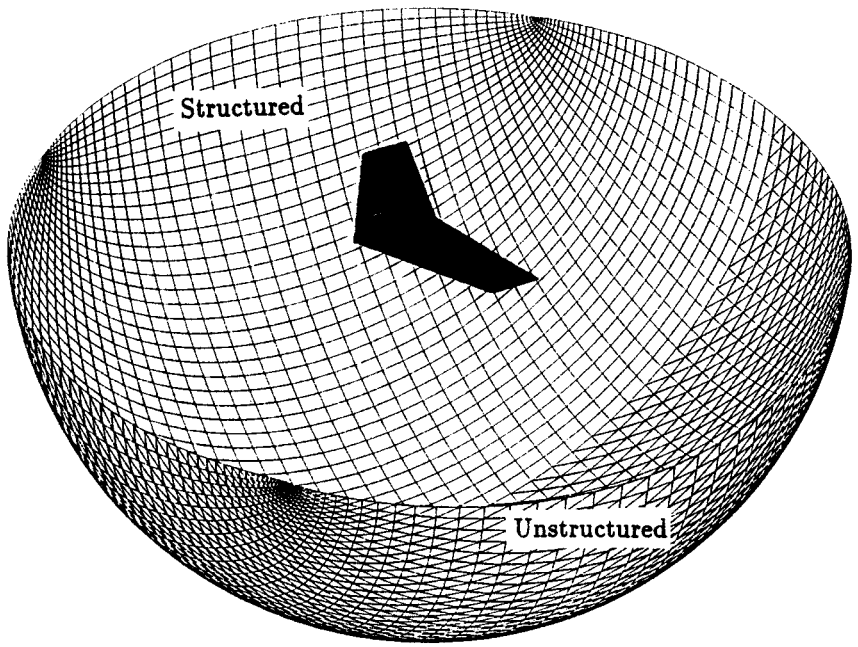
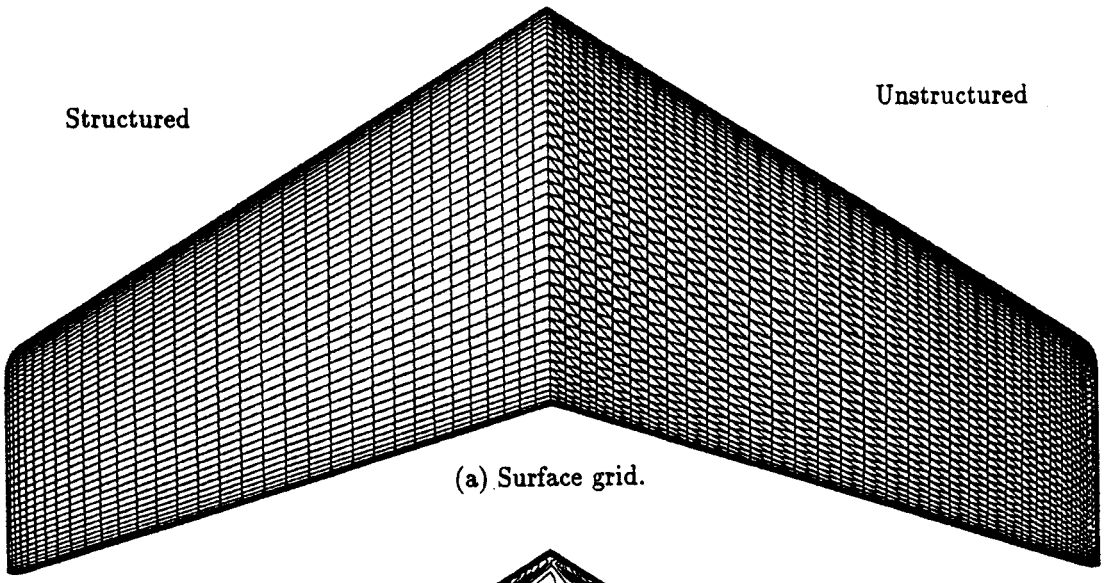
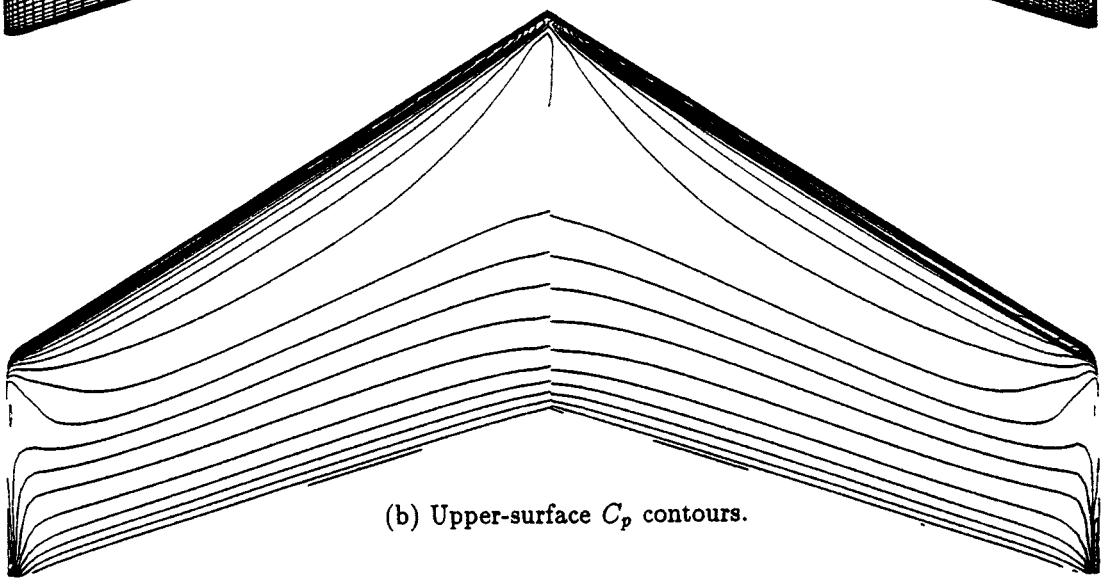


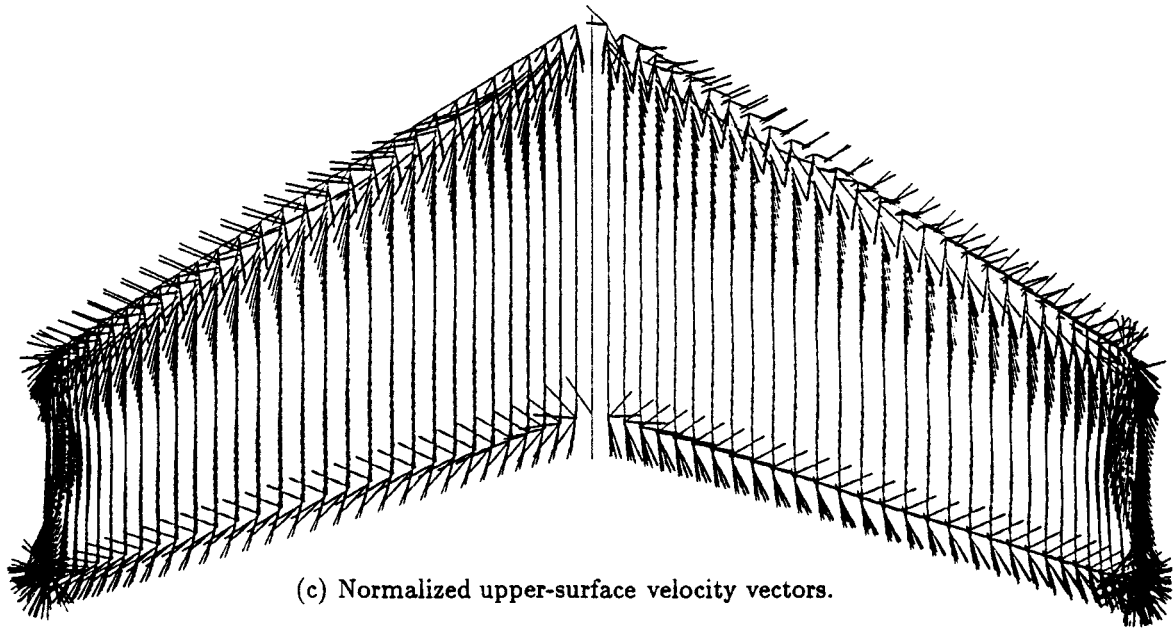
Figure 15.- Lower hemisphere of structured and unstructured 97 × 41 × 37 O-O grid for ONERA M6 wing.



(a) Surface grid.

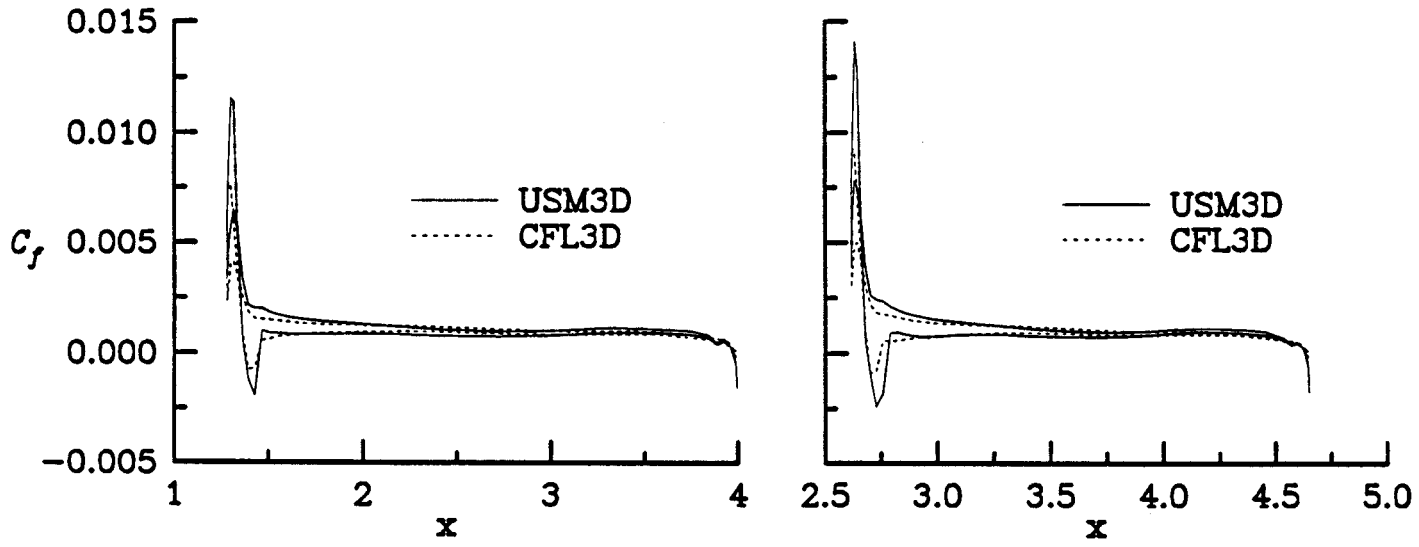


(b) Upper-surface C_p contours.



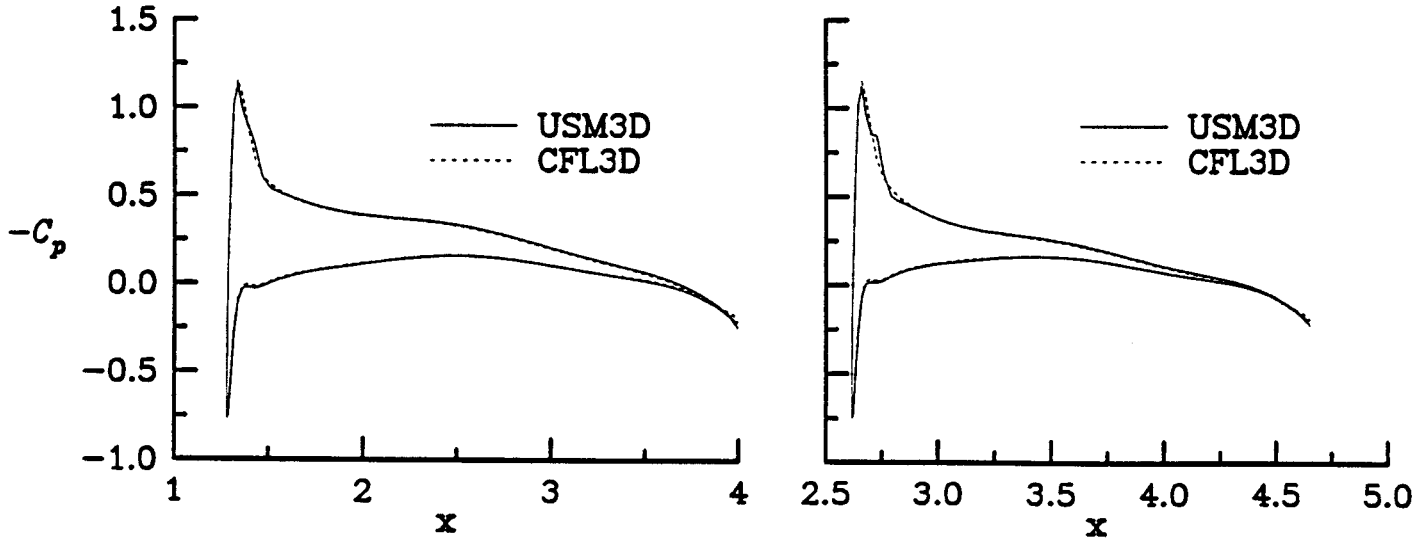
(c) Normalized upper-surface velocity vectors.

Figure 16.- Comparison of structured and unstructured grid and parameters for ONERA M6 wing. $M_\infty = 0.5, \alpha = 3°$, $R_e = 1 \times 10^6$, laminar.



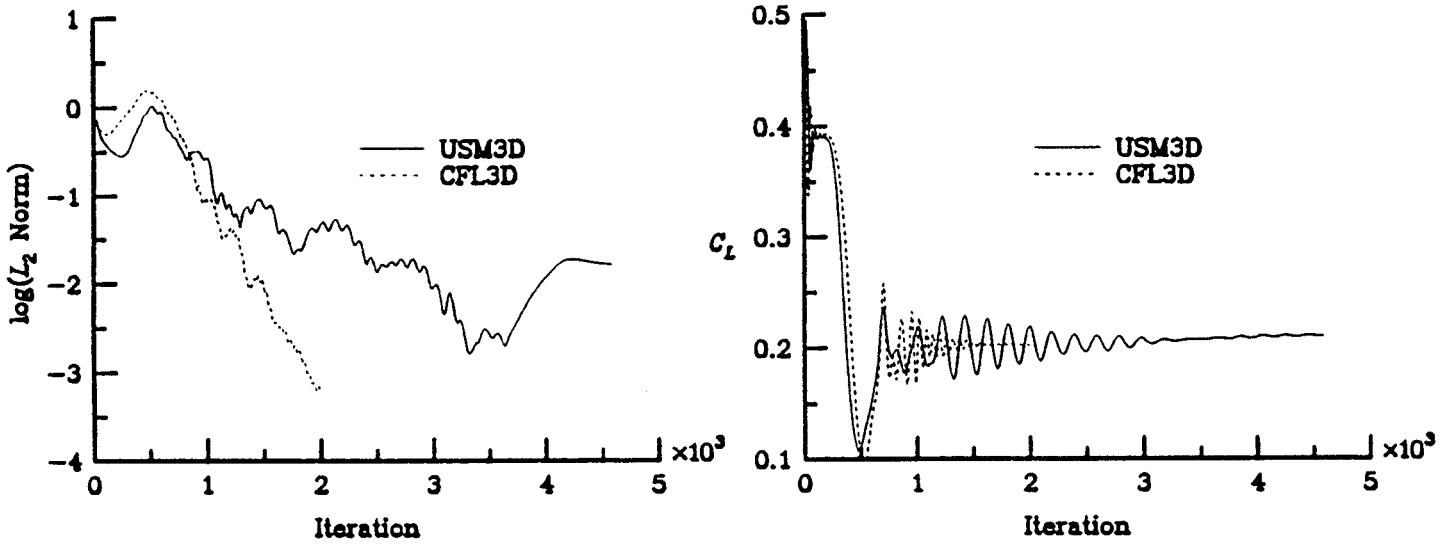
(a) $\eta = 0.44$ (b) $\eta = 0.90$

Figure 17.- Comparison of skin-friction distributions for structured and unstructured laminar solutions for ONERA M6 wing. $M_\infty = 0.5, \alpha = 3°$, $R_e = 1 \times 10^6$.



(a) $\eta = 0.44$ (b) $\eta = 0.90$

Figure 18.- Comparison of C_p distributions for structured and unstructured laminar solutions for ONERA M6 wing. $M_\infty = 0.5, \alpha = 3°$, $R_e = 1 \times 10^6$.



(a) L_2-norm

(b) Lift coefficient

Figure 19.- Comparison of convergence characteristics for structured and unstructured laminar solutions for ONERA M6 wing. $M_\infty = 0.5, \alpha = 3°$, $R_e = 1 \times 10^6$.