Efficient Method for Computing Transonic and Supersonic Flows About Aircraft

G. Volpe*
Grumman Corporate Research Center, Bethpage, New York 11714
and
A. Jameson†
Princeton University, Princeton, New Jersey 08544

A numerical method for computing flowfields about fighter-type aircraft is described in this paper. The time-dependent Euler equations are discretized on a single-block mesh in finite volume form and integrated to steady state via a Runge-Kutta scheme with a local time step. Convergence is accelerated by employing enthalpy damping and residual smoothing in addition to a multigrid sequencing of the computational mesh. While specifically designed for application in the transonic regime, the method can be used efficiently even at supersonic and low-subsonic speeds. Very realistic fighter configurations can be handled as demonstrated by the examples presented.

Introduction

THE advent of supercomputers and superminicomputers has brought the solution of the Euler equations into the toolbox of the aircraft designer. Indeed, methods of solving the Navier-Stokes equations are available. However, the current status of turbulence modeling lends a measure of unreliability to predictions based on Navier-Stokes formulations. In addition, such codes require very large running times, even on supercomputers, and, therefore, cannot be exercised frequently in a design environment. Currently, numerical methods based on the Euler equations offer an excellent compromise between reliability, speed, and faithful representation of the flowfield.

In recent years, several methods[1-11] have been proposed for the solution of the Euler equation. Key elements that determine the extent to which a code will be used in a project environment are its relative ease of use and its running time. All numerical methods are based on a discretization of the flow equations, which in turn relies on a discretization of the physical space about the configuration to be examined. Grid generation has matured to the point that meshes about very complex configurations can be generated. The grid generation step can be very complicated, however, and at times it can be carried out only by experienced users. It can, in addition, be time consuming, a feature that may be unacceptable in a preliminary design stage.

The space about relatively simple shapes can be represented quite well by a single-block structured mesh. In the present context, a structured mesh is one that has a definite topological structure. Single-block meshes are relatively easy to generate. For more complicated shapes (e.g., an aircraft with wings, nacelles, horizontal and vertical tails, etc.), composite meshes made up of several grids separately generated for each individual component would seem more appropriate. In such composite meshes, the individual grids, which may be topologically different, either meet at predefined interfaces[12,13] or overlap.[14] The generation of grids in the latter class is considerably easier than that of grids belonging to the former class. A recently introduced, alternative discretization is provided by unstructured grids.[15,16] However, methods based on unstructured grids have not yet reached the maturity level of methods based on structured grids. With a mesh that is fine enough to resolve a flowfield adequately, the storage requirements and running times of such methods are beyond the capacity of most current computers and the availability of most users. At present, composite grids offer the most versatile approach to the analysis of the flows about complex configurations. Unfortunately these, too, have drawbacks. First, as just mentioned, the grids are not generated easily. Second, the existence of interfaces or overlaps between the constituent grids requires special treatment of such artificial internal boundaries, inevitably leading to a deterioration of the convergence qualities of the underlying flow-solution scheme and, as a result, longer running times.

As will be illustrated in this paper, a single-block H-O mesh, which can be easily generated, can be used to represent the space around a large variety of configurations, including rather complex ones. With such a mesh, a fast numerical scheme can be designed to take advantage of the ordered sequence of mesh cells without the encumbrance of internal mesh boundaries. Such a numerical scheme will be described in this paper along with the grid generation process. The basis of the method rests on an algorithm by Jameson[17] for the integration of the time-dependent Euler equations to steady state via a Runge-Kutta scheme. Acceleration to steady state is achieved by the use of a local time step. Additional acceleration is provided by smoothing of the residuals and by enthalpy damping. Finally, the ordered, regular mesh system makes it possible to imbed the numerical scheme in a multigrid sequence of meshes to provide further acceleration. Excellent convergence rates have been exhibited by this method, and flowfields over a wide variety of aircraft configurations have been computed in as little as 5-10 min of computing time on a CRAY-XMP-class machine.

Discretization of the Euler Equations

An inviscid, rotational flow is uniquely defined at every point in space by the values of the density ρ, the three Carte-

Received Nov. 22, 1988; revision received Aug. 22, 1989.

*Senior Staff Scientist. Associate Fellow AIAA.
†Professor, Department of Mechanical and Aerospace Engineering. Associate Fellow AIAA.

sian components of velocity u,v,w, and the total energy E. The Euler equations describe the motion of such a flow and are derived from the physical laws of conservation, which embody the notion that, in the absence of singularities, the time rates of change of mass, momentum, and energy contained within any given volume must be equal to the net flux of the quantity through the boundary of the volume. In integral form, the equations are written as

$$\frac{\partial}{\partial t}\iiint_{\Omega} W^{(m)}\,d\Omega + \iint_{\partial\Omega} F^{(m)}\cdot dS = 0 \tag{1}$$

where t is the time, Ω denotes a given volume, and $\partial\Omega$ is the volume's boundary. The $W^{(m)}$ denotes the five-scalar quantities that are conserved, and $F^{(m)}$ is the corresponding vector flux. In a Cartesian coordinate system, the variables and fluxes are given by

$$W^{(1)} = \rho, \qquad F^{(1)} = [\rho u, \rho v, \rho w]^T \tag{2a}$$

$$W^{(2)} = \rho u, \qquad F^{(2)} = [\rho u^2 + p, \rho uv, \rho uw]^T \tag{2b}$$

$$W^{(3)} = \rho v, \qquad F^{(3)} = [\rho uv, \rho v^2 + p, \rho vw]^T \tag{2c}$$

$$W^{(4)} = \rho w, \qquad F^{(4)} = [\rho uw, \rho vw, \rho w^2 + p]^T \tag{2d}$$

$$W^{(5)} = \rho E, \qquad F^{(5)} = [\rho Hu, \rho Hv, \rho Hw]^T \tag{2e}$$

Assuming a perfect gas with a ratio of specific heats equal to γ, the pressure p and total enthalpy H can be expressed in terms of the density, velocity, and energy to close the system of equations. Thus,

$$E = \frac{p}{(\gamma-1)\rho} + \frac{1}{2}(u^2+v^2+w^2) \tag{3a}$$

$$H = E + p/\rho \tag{3b}$$

The solution to a specific flow about an aircraft-like body is obtained numerically by first discretizing the total external space into a large number of hexahedral cells and applying Eq. (1) to each cell. It is assumed that the unknown variables are to be solved for at the nodes of the mesh and that each node (i,j,k) has a control volume consisting of the eight cells meeting at that node (see Fig. 1). At each node, then

$$\frac{d}{dt}\left(\sum_{n=1}^{8} V_n\right) W_n^{(m)} + \sum_{n=1}^{8} Q_n^{(m)} = 0 \tag{4}$$

Here, V_n is the volume of the nth cell and $Q_n^{(m)}$ is the net flux through that cell. Denoting by S_ℓ the directed area of the ℓth face (of the nth cell), and by $F_\ell^{(m)}$ the mean flux vector across that face, the net flux through each cell is given by

$$Q_n^{(m)} = \sum_{\ell} F_\ell^{(m)} \cdot S_\ell \tag{5}$$

In Eq. (4), the fluxes across internal faces cancel exactly, and, therefore, the net flux through each control volume is the sum of the fluxes through the external faces only. In Eq. (5), S_ℓ is to be computed by performing a cross-product of the vectors joining diagonally opposite corners of the ℓth face, and $F_\ell^{(m)}$ is to be computed using averages of the values stored at the four constituent corners of the face.

Spurious oscillations could be set up by this discretization scheme and can take two forms. One type of oscillation can occur on either side of discontinuities in the solution, such as shocks. An odd-even point oscillation mode, which would give a zero net contribution to the flux balance in each control volume, could also be generated. It is possible that the latter mode might be suppressed in a steady-state solution by the boundary conditions, but it could adversely affect transients. A numerical device to control such oscillations is to add to Eq. (4) a dissipation term that goes to zero in the limit of zero mesh spacing. The dissipation has a low background level everywhere in order to suppress the odd-even point oscillations, and it is increased in the presence of a physical discontinuity in the flowfield. The dissipation is constructed in a way that preserves the conservation form of the equations. It is implemented by subtracting from $Q_n^{(m)}$ in Eq. (4) a dissipative flux, $D_n^{(m)}$, which is the sum of three terms separately constructed for each of the three computational coordinate directions. The term in the i-coordinate direction is

$$D_{n_i}^{(m)} = d_{i+1,j,k} - d_{i,j,k} \tag{6}$$

with

$$d_{i,j,k} = r\left[\epsilon^{(2)} - \epsilon^{(4)}\delta_x^2\right]\left(W_{i,j,k}^{(i)} - W_{i-1,j,k}^{(i)}\right) \tag{7}$$

Similar terms are defined for the j and k directions. In Eq. (7), δ_x^2 is a second-difference operator, and $\epsilon^{(2)}$ and $\epsilon^{(4)}$ are adaptive coefficients. The scaling factor r is based on an estimate of the maximum local wave speed. This is the speed at which the fastest wave could cross a mesh interval and, denoting by Δt^* the local time step used to integrate Eq. (4), it would be proportional to $1/\Delta t^*$. The amount of background dissipation is controlled by $\epsilon^{(4)}$, and the dissipation near discontinuities is controlled by $\epsilon^{(2)}$. The discontinuities are detected by making $\epsilon^{(2)}$ proportional to a normalized second difference of the pressure.

This formulation introduces dissipation terms that are of third order, except in regions of steep pressure gradients. The discretization of the convective terms can be shown to be second-order accurate.

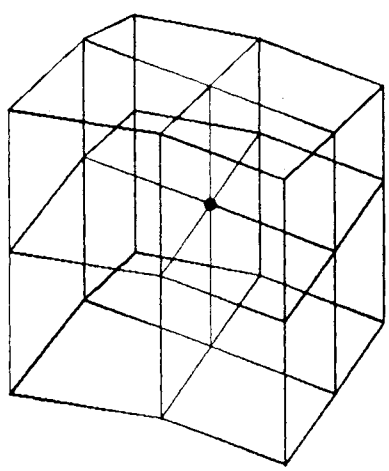

Fig. 1 Control volume around point i,j,k.

Time Integration

With a computational mesh that is independent of time, Eq. (4) can be rewritten as

$$\frac{d}{dt} W_{i,j,k}^{(m)} + R(W^{(m)}) = 0 \tag{8}$$

where $R(W^{(m)})$ denotes the residual and it is given by

$$R(W^{(m)}) = \frac{1}{V_{i,j,k}}(Q_{i,j,k} - D_{i,j,k}) \tag{9}$$

and where $Q_{i,j,n}$ and $D_{i,j,n}$ respectively, are the total convective flux and the total dissipative flux through the control volume $V_{i,j,k}$. The steady-state solution is obtained by integrating Eq. (8) with a multistage Runge-Kutta scheme. If only the steady-

state solution is of interest, a locally varying time step Δt can be used. As discussed by Jameson,[18] this class of schemes can exhibit excellent stability properties and very fast convergence rates.

With an N-stage scheme, the advance from time (t) to time $(t+\Delta t)$ in each variable W is computed by the following recursive formulas:

$$W^{(0)} = W(t) \tag{10a}$$

$$\vdots$$

$$W^{(1)} = W(t) - \alpha_1 \Delta t R(W^{(0)}) \tag{10b}$$

$$\vdots$$

$$W^{(m)} = W(t) - \alpha_n \Delta t R(W^{(n-1)}) \tag{10c}$$

$$\vdots$$

$$W(t+\Delta t) = W^{(N)} \tag{10d}$$

A five-stage scheme, with $\alpha_1 = 1/4$, $\alpha_2 = 1/6$, $\alpha_3 = 3/8$, $\alpha_4 = 1/2$, and $\alpha_5 = 1$, has worked very well in practice. Also, in concert with the findings of Jameson,[19] the artificial dissipation terms can be frozen at the values computed during the second stage. This strategy not only decreases the computing time but also increases the stability margin of the scheme.

Residual Smoothing

As indicated in previous studies, Ref. 3 for example, computational efficiency can be enhanced by a smoothing of the residuals. The maximum permissible time step is set by the local Courant number. This limitation is relaxed if each residual is replaced by an average of its neighbors. This average, denoted by $\bar{R}$, is computed implicitly by solving

$$(1-\epsilon_x \delta_x^2)(1-\epsilon_y \delta_y^2)(1-\epsilon_z \delta_z^2)\bar{R} = R \tag{11}$$

where R designates the unsmoothed residual, δ_x^2, δ_y^2, δ_z^2 are second-difference operations, and ϵ_x, ϵ_y, ϵ_z are smoothing coefficients.

Enthalpy Damping

Another useful technique for accelerating convergence to the steady state in the case of a homoenthalpic flow is enthalpy damping. This technique exploits the difference between the transient value of the locally computed enthalpy and its known, steady-state value, which is also the value at infinity. One can assume that the rate of change of each variable is proportional to this difference. Thus,

$$\frac{\partial W}{\partial t} + \beta W(H - H_\infty) = 0 \tag{12}$$

where β is a user-defined constant. This information can be used at the end of each time step to obtain a new improved estimate of a variable W by correcting the value $\tilde{W}$ computed at the end of the Runge-Kutta step through

$$W = \tilde{W} + \beta \Delta t W(\tilde{H} - H_\infty) = 0 \tag{13}$$

In practice, a slight modification is needed when applying this type of correction to the energy variable. In this case, Eq. (13) is modified to

$$(\rho E) - (\tilde{\rho}\tilde{E}) + \beta \Delta t[\rho E + \tilde{p} - \tilde{\rho} H_\infty] = 0 \tag{14}$$

It is instructive to mention at this point that the spatial accuracy of the scheme is not affected by any of these schemes designed to accelerate convergency. The accuracy of the steady-state solution reflects the accuracy with which the residuals at each node of the basic computational method is computed.

Multigrid Strategy

The evolution in time of the solution is dependent on the mesh spacing through the limitations that the grid sets on the Courant number. Even though these limitations can be relaxed by employing a smoothing of the residuals, the implication remains that the time evolution is faster on coarser grids than finer ones. It would seem advantageous, then, to devise a strategy that would use this faster approach of the steady state on a coarser grid to generate information that can be used to accelerate the approach of the steady state on any given finer grid. Alternatively, one could look at the time marching to steady state as an iterative process of reducing the errors (i.e., residuals) at each node. Errors are reduced by an exchange of information between nodes, and during each iterative cycle, information to any particular node comes only from nodes involved in computing the residual at the node. Again, one can see that an exchange of information between two particular points in space takes place faster on coarser grids. Such was the basic idea of multigrid schemes when first presented by Brandt.[20] Here, the scheme described by Jameson[4] is followed.

A multigrid scheme involves the exchange of information among nodes of different grids covering the same physical (and computational) space. The exchange is greatly simplified if a coarser mesh is generated by eliminating alternate points in each of the coordinate directions of a given mesh. With such a setup, one can inject to each point on a coarse mesh $(K+1)$ the values of the variables W at the coincident point on next higher level mesh (K). Once this is accomplished, Runge-Kutta time steps can be performed on the coarse mesh. The time step on the coarse grid is done with a modified residual, however. The modified residual on the coarser mesh is obtained by adding to the standard, computed residual a term reflecting the difference between the residual computed on the finer mesh and the residual computed on the coarser mesh using the values transferred from the finer grid. Thus, on grid level $K+1$, Eqs. (10a–10c) are replaced by

$$W_{K+1}^{(0)} = T_{K+1,K} W_K \tag{15a}$$

$$W_{K+1}^{(1)} = W_{K+1}^{(0)} - \alpha_1 \Delta t \left(R_{K+1}^{(0)} + P_{K+1} \right) \tag{15b}$$

$$W_{K+1}^{(N)} = W_{K+1}^{(0)} - \alpha_N \Delta t \left(R_{K+1}^{(N)} + P_{K+1} \right) \tag{15c}$$

where R_{K+1} is the residual, as computed on the $K+1$ mesh and

$$P_{K+1} = Q_{K+1,K} R_K(W_K) - R_{K+1}^{(0)} \tag{16}$$

is the additional term. The $Q_{K+1,K}$ is a transfer operator denoting a weighted average of the residuals at the 27 points nearest the coincident point on mesh level K (including the point itself). The P_{K+1} is constant as long as one operates on the $K+1$ mesh without returning to a higher level. A number of time steps can be performed on any grid level before passing on to a coarser level or returning to the next higher level. In returning to level K, the correction to W calculated on level $K+1$ is used to provide an improved value of the variable on the finer mesh. Denoting by W_{K+1}^{+} the improved value of W on grid $K+1$ after various time steps and after correcting for any passage to a still coarser mesh $K+2$, an improved value of W can be computed from

$$W_K^{+} = W_K + I_{K,K+1}(W_{K+1}^{+} - W_{K+1}^{(0)}) \tag{17}$$

Here, the operator $I_{K,K+1}$ is defined in such a way that at coincident points of the two meshes the value of W is simply transferred. At intermediate points, a trilinearly interpolated value of the difference between values at two mesh levels is added to the current value on the finer mesh.

The transfer of information between grids and the computation of time steps on the coarser grids obviously entail some computing times. However, the time steps are performed in

only a fraction of the time needed on the fine mesh, and the projected reduction in the total number of time steps more than offsets the cost of sequencing through the grids. In the current scheme, a typical multigrid sequence calls for a single time step on each grid level before passing on to a coarse grid and a simple transfer of data without any time step in stepping up through the grid levels. Occasionally, better convergence has been observed by performing one time step on the way up through the meshes.

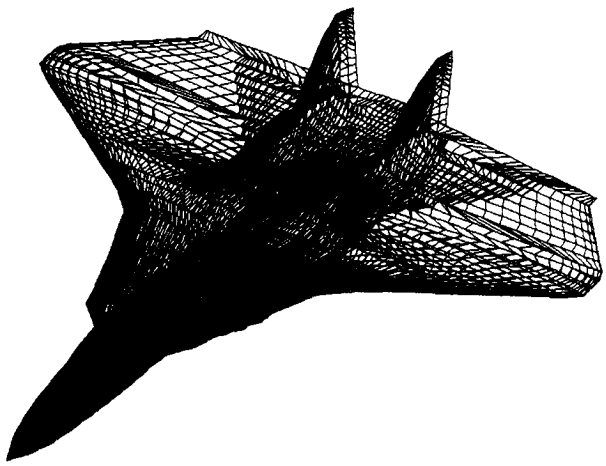

Fig. 2 Surface/wake grid on an aircraft with sweptback wings, horizontal tails, and twin vertical tails.

Computational Mesh

In this work, an H-O mesh is used for space discretization, and it is generated by combining a series of two-dimensional meshes around selected cross sections of the aircraft. Each mesh is generated by conformal mapping technique in planes normal to the longitudinal axis of the aircraft. The technique was originally suggested by Moretti[21] and has been used by Siclari[22] for computing supersonic flows by space marching techniques.

An arbitrary cross section can be mapped into a near circle by removing corners through a sequence of Kármán-Trefftz mappings. Denoting by $Z = X + iY$ the physical ordinate and by ζ the corresponding point in the mapped plane,

$$\left(\frac{Z - Z_0}{Z + \bar{Z}_0}\right) = \left(\frac{\zeta - Z_0}{\zeta + \bar{Z}_0}\right)^{\tau} \tag{18}$$

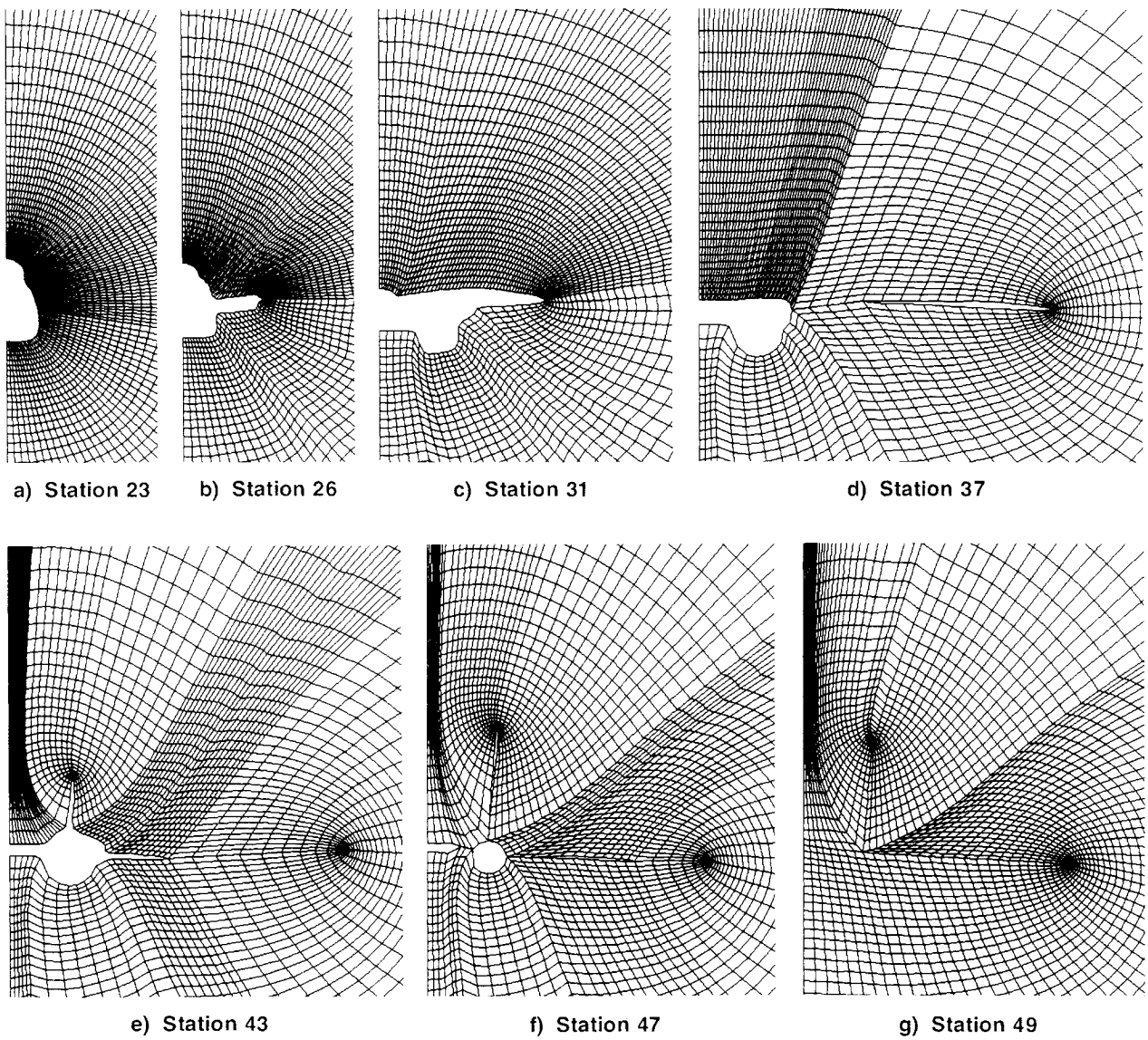

Fig. 3 Representative cross-sectional grid planes on aircraft with twin vertical tails.

where Z_0 is the location of the singularity and $\bar{Z}_0$ its conjugate value. The $\tau\pi$ is the external angle of the corner being removed. The number of mappings is equal to the number of corners to be removed, and on mappings subsequent to the first, Z and Z_0 are identified with the ordinates in the most recently mapped plane. In the near-circle plane, a computational grid is obtained in a straightforward manner. One family of lines is given by a set of radial lines centered on a point equidistant from the vertical and horizontal extremes of the mapped cross section. The mesh is completed by a second set of lines wrapping around the mapped body. This second set is gradually distorted to a perfect circle at a predetermined distance denoting the outer boundary of the mesh.

A typical cross section of the aircraft may cut through one or more components (e.g., fuselage, wing, nacelles, tails), and these may be connected or separated. In the latter case, the separate pieces are connected by slits. If slits are present, care is taken to match up grid points on either side of the slit. By having grid continuity across a slit, application of boundary conditions in the numerical scheme is simplified. The distance to which each two-dimensional mesh extends is kept constant. Upstream of the aircraft's nose, the first mesh plane on the aircraft is extended to a predetermined distance with gradually increasing spacing between the planes. This portion of the grid is purely cylindrical; then, if the first aircraft mesh plane is very close to the nose, as it typically is, a degenerate axis is present in the grid upstream of the aircraft. A similar strategy produces a cylindrical grid downstream of the aircraft. This part of the grid wraps around a slit, since the last cross section includes the wing's trailing edge and/or wake. In a typical grid, one-half of the planes is positioned along the aircraft's length, and the other half of the planes is divided between the upstream and the downstream portions of the grid.

The grid generated about an F-14-like aircraft (wing, nacelle, horizontal and twin vertical control surfaces) is depicted in Figs. 2 and 3. The wing and both control surfaces have sweptback leading and trailing edges. In Fig. 2 the surface grid is shown reflected about the symmetry plane. The 0 grid around several representative cross sections of the aircraft are shown in Fig. 3. On this configuration, the wake slit coming off the wing is gradually warped to intercept the rearward horizontal tail. As the back end of the aircraft is approached, additional wake slits appear between the engine nacelle, each of the horizontal and vertical tails, and the "pancake" between the twin vertical tails. All the slits eventually join to form an "inverted-T" slit extending to infinity.

For supersonic freestreams, a more efficient use of grid points is to vary the distance to which the two-dimensional mesh extends in each cross-sectional plane, from a small value near the aircraft's nose to a larger value downstream. In addition, because of the restricted domains of dependence and influence, a larger percentage of the grid planes can be concentrated along the aircraft's length.

Boundary Conditions

At node points lying on the aircraft, flow tangency is enforced at the end of each time step by setting to zero any normal component of velocity. In addition, fluxes through faces lying on the aircraft surface are explicitly set to reflect this condition. Continuity is enforced across wake slits behind the wing by replacing the values of each variable at matching nodes by their average. Similarly, values at nodes sharing a common grid point on the degenerate axis ahead of the aircraft are replaced by the average. At the outer boundaries of the mesh, appropriate inflow or outflow boundary conditions are imposed using Riemann invariants to attenuate the spurious reflections of outgoing waves into the field. With zero sideslip, in addition, only one-half of the flowfield need be considered. In this case, symmetry is enforced on the plane of symmetry of the aircraft.

Numerical Examples

Flows over a wide variety of configurations and covering a considerable Mach number range have been calculated with the method. The solution algorithm is very efficient. In the cases that will be shown, including those representing extreme flow conditions, convergence to steady state is typically obtained in 200–300 cycles. Fewer are needed for less severe flow conditions. The mesh topology chosen is capable of treating very complex bodies as is shown in Figs. 2 and 3. Pressure distributions computed on that configuration for a Mach number of 0.80 and an angle of attack of 20 deg are shown in Fig. 4. In this computation the surface geometry is treated as a solid body (i.e., both the inlet of the engine and the nozzle are closed). The mesh contained a total of 65 half planes normal to the longitudinal axis of the aircraft. Of these, half were on

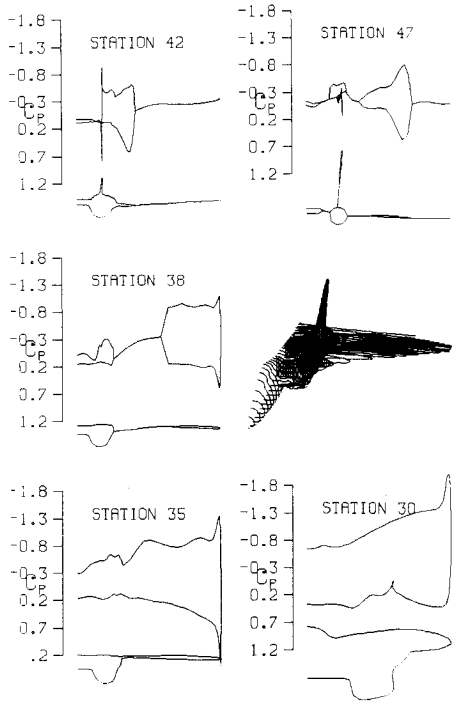

Fig. 4 Computed surface pressure distributions; fighter-type aircraft; $M_\infty = 0.80$, $\alpha = 20$ deg.

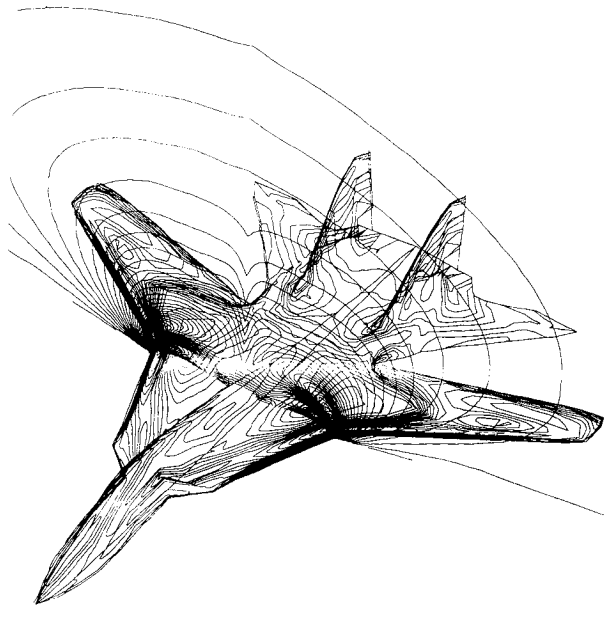

Fig. 5 Isomach contours on upper wing/body surface and in a cross-sectional plane; fighter-type aircraft; $M_\infty = 0.80$, $\alpha = 20$ deg.

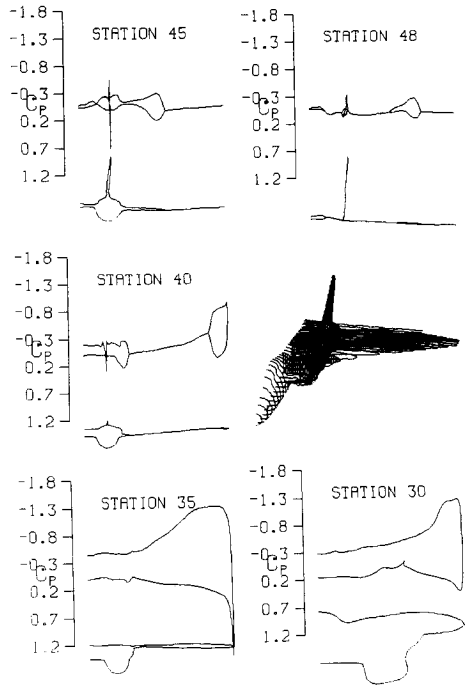

Fig. 6 Computed surface pressure distributions; fighter-type aircraft; $M_\infty = 0.95$, $\alpha = 10$ deg.

the aircraft, and the other half were split between the regions upstream and downstream of the body. Each plane in turn contained a 97×33 polar-like grid. Representative portions of these planes were shown in Fig. 3. The computation over this 200,000-point mesh requires approximately 12 Mwords of storage. The cross sections at which the surface distributions are displayed in Fig. 4 correspond, in ascending order to station 30 behind the glove; station 35 just before the location where the wing leaves the fuselage; station 38 cutting through the rear/outer portion of the wing and the pancake/engine region; station 42 past the trailing edge of the wing where the horizontal and vertical control surfaces are already present; and station 47 close to the tail of the aircraft. Notice that the solution is continuous across all wake slits. It is to be pointed out in particular that at station 47 where the control surfaces do not touch the engine nozzle that four separate wake slits are present. In Fig. 4 there is indication of a shock close to the leading edge of the glove and the wing. In Fig. 5, which depicts the isomach contours on the body surface and in a midfuselage cross-sectional plane. The location of the shock is indicated by the clustering of contour lines near the leading edge. The region of supersonic flow can be seen to extend into the field as a "bubble" leaning toward the symmetry plane. A region of supersonic flow also is present on the in-board portion of the wing, just past the cross-plane that was depicted.

At a Mach number of 0.95 and an angle of attack of 10 deg, the flow is supersonic over most of the wing, and the shock moves back from the leading edge, as can be noticed in Fig. 6. Of interest in this case is the shock on the pancake just in front of the vertical tails, which can be discerned in Fig. 7. Typically, shocks in the present method are smeared over three or four mesh cells. The smearing is largely dependent on the value chosen for the dissipation coefficients, $\epsilon^{(2)}$ and $\epsilon^{(4)}$. These are kept constant throughout the examples presented. Convergence rates and accuracy in shock representation were weighed in selecting the coefficients. The convergence rate can be increased at the cost of additional smearing of shocks. The solution elsewhere shows little sensitivity to the choice of dissipation coefficients.

This example demonstrates the geometric complexity that can be handled by the method. The range of applicability and the accuracy of the method will be examined in the following.

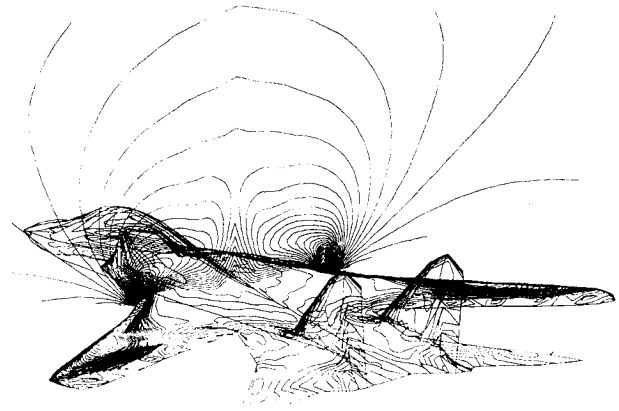

Fig. 7 Isomach contours on upper wing/body surface and in a cross-sectional plane; fighter-type aircraft; $M_\infty = 0.95$, $\alpha = 10$ deg.

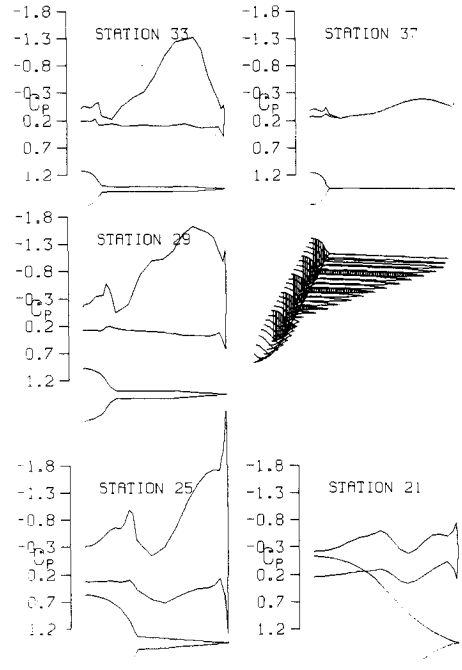

Fig. 8 Computed surface pressure distributions; chine-forebody configuration; $M_\infty = 0.20$, $\alpha = 15$ deg.

The next example illustrates the low-speed capability of the algorithm. Figure 8 depicts surface distributions of pressure coefficients computed on a generic fighter aircraft at a free-stream Mach number of 0.20 and an angle of attack of 15 deg. Streamline "ribbons" are shown in Fig. 9. This configuration features a chine-shaped forebody and a sharp-edged, cropped delta wing. Typical cross sections are visible in Fig. 8. This configuration is similar to one that has been tested by Erickson,[23] and a sketch of his observations is also shown in Fig. 9.

This computation was carried out on a mesh consisting of 49 longitudinal planes, each of which contained 49 points in the azimuthal direction and 24 points in the radial direction. It is remarkable that the flow features are captured with this relatively coarse mesh. This example also has been computed on a mesh containing twice as many points in each coordinate direction on the CRAY-2 machine of the NASA Ames National Aerodynamic Simulator. The flow features were largely unchanged in passing to the finer mesh. The major noticeable difference concerned the vortices, which were slightly tighter on the finer mesh.

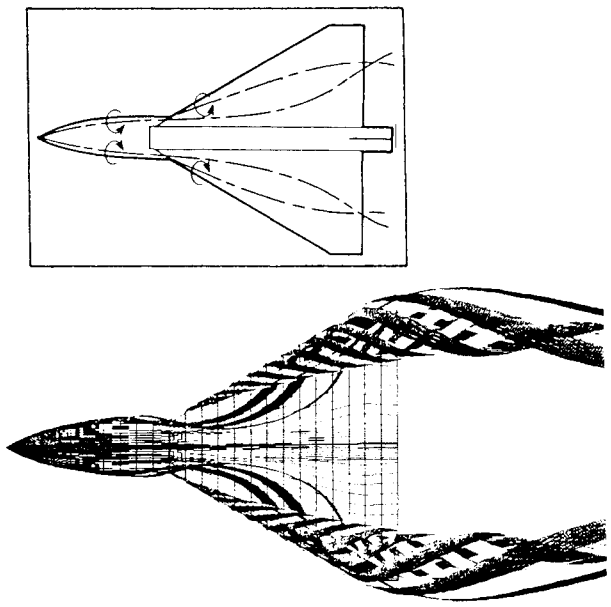

Fig. 9 Top view of streamline "ribbons" on chine-forebody configuration; $M_\infty = 0.20$, $\alpha = 15$ deg; inset: experimental tracks of vortex cores (from Ref. 23).

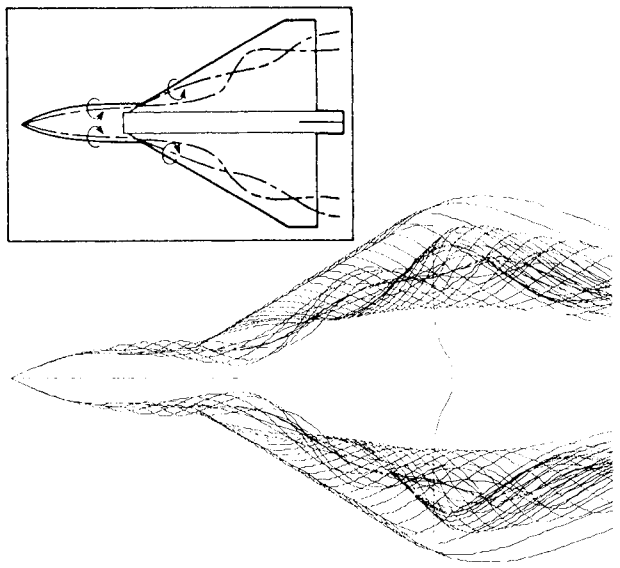

Fig. 11 Top view of streamlines on chine-forebody configuration; $M_\infty = 0.20$, $\alpha = 25$ deg; inset: experimental tracks of vortex cores (from Ref. 23).

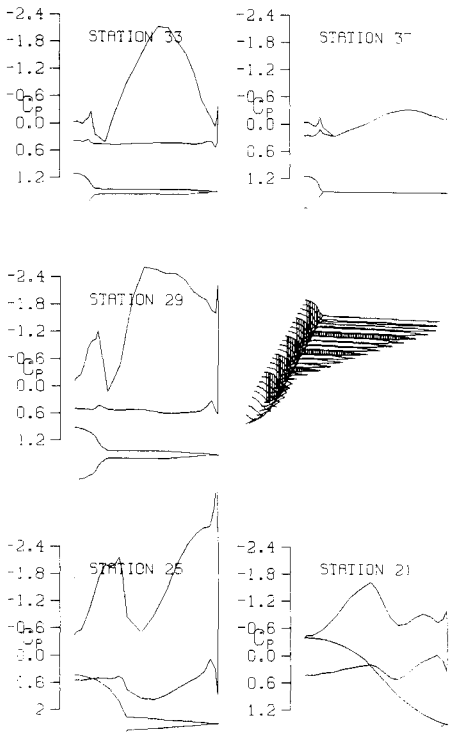

Fig. 10 Computed surface pressure distributions; chine-forebody configuration; $M_\infty = 0.20$, $\alpha = 25$ deg.

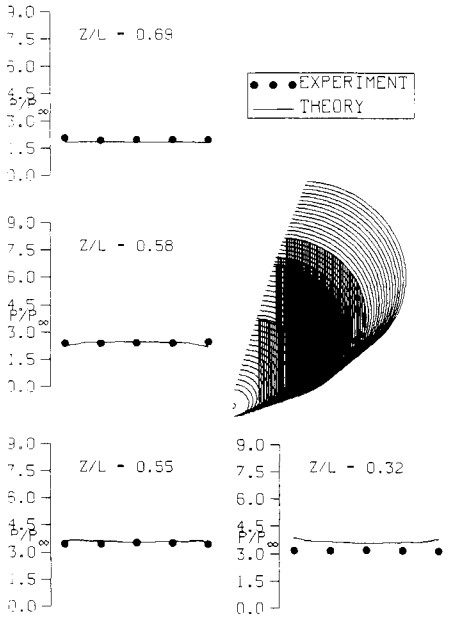

Fig. 12 Computed surface pressure distributions; biconic missile body; $M_\infty = 6.0$, $\alpha = 0$ deg.

At this point, it should be mentioned that the convergence rate of the algorithm for a particular geometry and set of flow conditions has shown little sensitivity to mesh size. Thus, on refining a mesh, the time needed to advance one time step increases in proportion to the number of grid points, but the number of time steps needed to achieve convergence (usually defined as a reduction in the residual by three orders of magnitude) is largely unchanged.

For this generic fighter configuration, increasing the angle of attack to 25 deg increases the interaction between the chine and wing vortices. The chine vortex is pushed outboard sooner, as can be observed in Fig. 10. Vortices can be identified as "humps" in a pressure plot. The chine vortex starts wrapping around the wing vortex earlier along the length of the body. The wing vortex is substantially larger than it was at the lower angle of attack. The computations are again in qualitative agreement with the observation of Erickson, as can be seen from Fig. 11.

The last example addresses the flow past a spherically blunted, on-axis biconic body at Mach 6. This configuration, which consists of a conical shape with a half angle of 12.84 deg in the front part followed by another conical section with a 7-deg half angle, has been tested in the Langley 20-in. Mach-6 tunnel by Miller and Gnoffo.[24] Computations were done with a mesh comprising 65 longitudinal planes, of which 8 were located upstream of the body and 8 were set downstream. Each cross-sectional plane contained 49 points in the circumferential

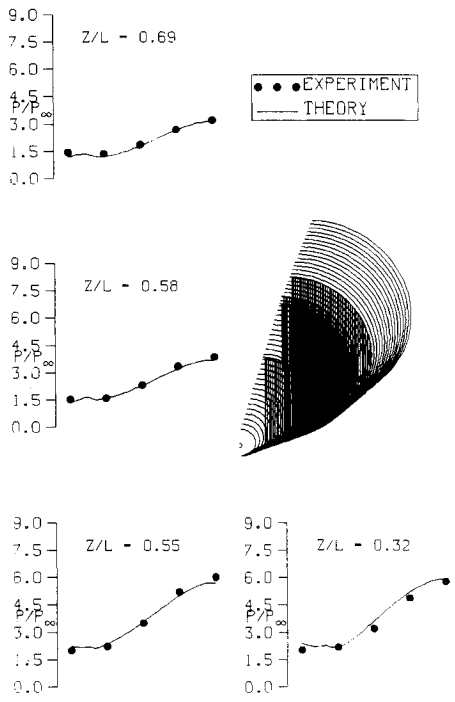

Fig. 13 Computed surface pressure distributions; biconic missle body; $M_\infty = 6.0$, $\alpha = 5$ deg.

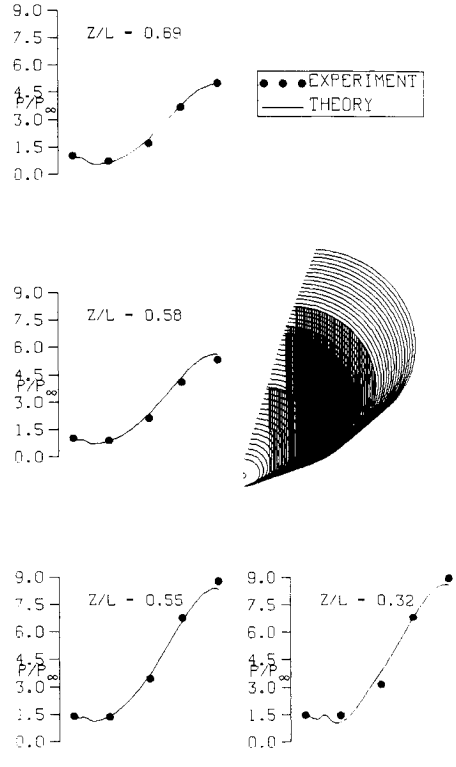

Fig. 14 Computed surface pressure distributions; biconic missile body; $M_\infty = 6.0$, $\alpha = 10$ deg.

direction and 33 in the radial direction. The outer boundary of the mesh was gradually expanded, as described earlier, to create a "supersonic" mesh. Predictions of surface pressure distributions were in very good agreement with experimental data for all three angles of attack computed (0, 5, and 10 deg) as can be seen from Figs. 12-14. Total force coefficients (decomposed along in the normal and axial directions) also were predicted quite well, as can be noticed from Table 1. Pressure contours on the surface and in three cross-sectional planes are depicted in Fig. 15 for the 5-deg incidence case. As can be seen, the bow shock is captured quite well. The clustering of lines on the surface of the body just in front of the second plane which is depicted is a result of the change in body slope angle there.

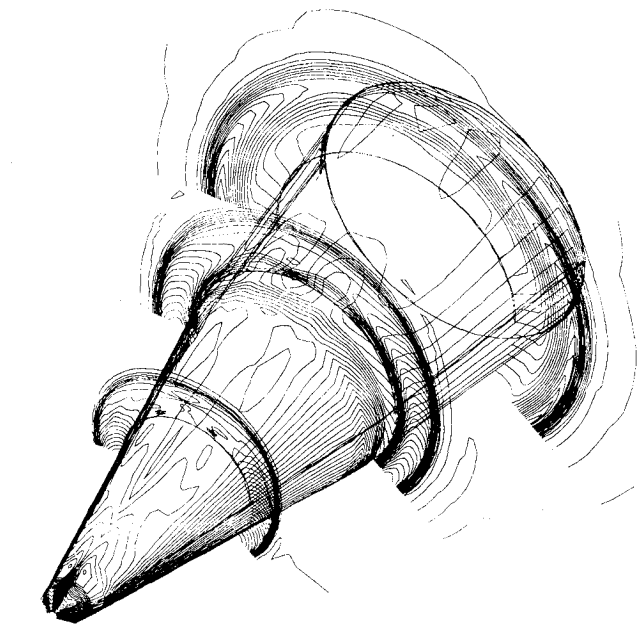

Fig. 15 Pressure contours on surface and three cross-sectional planes; biconic missile body; $M_\infty = 6.0$, $\alpha = 5$ deg.

Table 1 Computed and experimental aerodynamic coefficients for biconic missile body

	Theory		Experiment	
α, deg	C_N	C_A	C_N	C_A
0	−0.005	0.073	−0.003	0.096
5	0.151	0.091	0.153	0.102
10	0.313	0.120	0.313	0.119

Concluding Remarks

The algorithm described in this paper has proved very reliable and efficient over a considerable Mach number range, from subsonic through transonic to high supersonic flow conditions. Even in cases of extreme flow condition, where substantial flow separation is encountered, it has exhibited excellent convergence properties. The mesh topology used simplifies the grid generation process and is capable of adequately describing bodies of considerable geometric complexity. The simplicity and speed of the combined grid generation/solution algorithm procedure make the method an ideal tool in a preliminary design environment. Preliminary qualitative and quantitative comparisons with experimental data at both the low and the high ends of the Mach regime have attested to the accuracy of the predictions. Validation of the method against experimental data is currently underway.

References

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

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

[3]James, A., and Baker, T., "Solution of the Euler Equations for

Complex Configurations," AIAA Paper 83-1929, July 1983.
[4]Jameson, A., "Solution of the Euler Equations by a Multigrid Method," *Applied Math and Computation,* Vol. 13, 1983, pp. 327–356.
[5]Hemker, P. W., and Spekreijse, S. P., "Multigrid Solution of the Steady Euler Equations," *Notes on Numerical Fluid Mechanics,* Vol. 11, Vieweg Verlag, 1985.
[6]Pulliam, T. H., and Steger, J. L., "Recent Improvements in Efficiency, Accuracy and Convergence for Implicit Approximate Factorization Algorithms," AIAA Paper 85-0360, Jan. 1985.
[7]Hall, M. G., "Cell Vertex Multigrid Schemes for Solution of the Euler Equations," *Numerical Methods for Fluid Dynamics, Part II,* edited by K. W. Monton and M. J. Bounes, Clarendon Press, Oxford, U.K., 1986, pp. 305–345.
[8]Jameson, A., and Schmidt, W., "Recent Developments in Numerical Methods for Transonic Flows," *Proceedings of the 3rd International Conference on Finite Elements in Nonlinear Mechanics,* edited by J. St. Doltsinis, North Holland, 1985, pp. 467–493.
[9]Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow over a Complete Aircraft," AIAA Paper 86-0103, Jan. 1986.
[10]Thomas, J. L., Van Leer, B., and Walters, R. W., "Implicit Flux-Split Schemes for the Euler Equations," AIAA Paper 85-1680, 1985.
[11]Anderson, W. K., Thomas, J. L., and Whitfield, D. L., "Multigrid Acceleration of the Flux Split Euler Equations," AIAA Paper 86-0105, Jan. 1986.
[12]Belk, D. M., "Time-Accurate Euler Equations Solutions on Dynamic Blocked Grids," AIAA Paper 87-1127, June 1987.
[13]Rai, M. M., "A Relaxation Approach to Patched-Grid Calculations with the Euler Equations, *Journal of Computational Physics,* Vol. 66, 1986, pp. 99–131.
[14]Benek, J. A., Donegan, T. L., and Suhs, N. E., "Extended Chimera Grid Embedding Scheme with Applications to Viscous Flows," AIAA Paper 87-1126, June 1987.
[15]Baker, T., "Three-Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets," AIAA Paper 87-1124, June 1987.
[16]Lohner, R., "The Efficient Simulation of Strongly Unsteady Flows by the Finite Element Method," AIAA Paper 87-0555, Jan. 1987.
[17]Jameson, A., "A Vertex-Based Multigrid Algorithm for Three-Dimensional Compressible Flow Calculations," ASME Symposium on Numerical Methods for Compressible Flow, American Society of Mechanical Engineers, New York, 1986.
[18]Jameson, A., "Transonic Flow Calculations for Aircraft," *Lecture Notes in Mathematics, Numerical Methods in Fluid Dynamics,* edited by F. Brezzi, Springer-Verlag, 1127, 1985, pp. 156–242.
[19]Jameson, A., "A Nonoscillatory Shock-Capturing Scheme Using Flux Limited Dissipation," *Lectures in Applied Mathematics,* Vol. 22, Pt. 1, *Large Scale Computations in Fluid Mechanics,* edited by B. E. Engquist, S. Osher, and R. C. J. Sommerville, American Mathematical Society, Providence, RI, 1985, pp. 345–370.
[20]Brandt, A., "Multi-Level Adaptive Solution to Boundary Value Problems," *Mathematical Computations,* Vol. 31, 1977, pp. 333–390.
[21]Moretti, G., "Conformal Mappings for Computations of Steady-Three-Dimensional, Supersonic Flows," ASME Winter Meeting, American Society of Mechanical Engineers, New York, 1976.
[22]Siclari, M. J., "The Computational Treatment of Supersonic Wake Flows in NCOREL," AIAA Paper 85-0304, Jan. 1985.
[23]Erickson, G. E., and Brandon, J. M., "On The Nonlinear Aerodynamic and Stability Characteristics of a Generic Chine-Forebody Slender-Wing Fighter Configuration," NASA-TM 89447, 1987.
[24]Miller, C. G., and Gnoffo, P. A., "Pressure Distributions and Shock Shapes for 12.84°/7° On-Axis and Bent-Nose Biconics in Air at Mach 6," NASA-TM 53222, 1981.