AIAA -86-0103

Calculation of Inviscid Transonic Flow over a Complete Aircraft

A. Jameson, T.J. Baker and N.P. Weatherill
Department of Mechanical Engineering
Princeton University
Princeton, New Jersey 08544
U.S.A.

AIAA 24th Aerospace Sciences Meeting
January 6–9, 1986
Reno, Nevada

CALCULATION OF INVISCID TRANSONIC FLOW OVER A COMPLETE AIRCRAFT *

A. Jameson, T.J. Baker and N.P. Weatherill
Department of Mechanical Engineering
Princeton University
Princeton, New Jersey 08544
U.S.A.

We present preliminary results for the calculation of inviscid transonic flow over a complete aircraft. These results have been obtained by a novel approach that combines a new procedure for mesh generation with a new finite element method for solving the Euler equations.

Introduction

A principal goal of computational aerodynamics has been the calculation of transonic flow over a complete aircraft. Great strides have been made in this direction with major improvements in both the flow solution algorithms and mesh generation techniques. During the last five years, in particular, a variety of methods for solving the complete Euler equations of inviscid flow have been presented, and have been extended to progressively more complex configurations [1,2,3,4,5]. While complete aircraft were treated without too much difficulty by the earlier panel methods for subsonic flow, the goal of solving the full nonlinear equations of compressible flow for a complete aircraft has proved less easy to reach, and has been paced by the difficulties of mesh generation. A finite element solution of the potential flow equation for a complete aircraft has been obtained by Bristeau, Glowinski, Periaux, Perrier, Pironneau and Poirier [6]. We present here a new finite element method for solving the complete Euler equations of three dimensional compressible flow on a tetrahedral mesh. The method has no requirement of structure in the mesh, and we show preliminary results for the prediction of transonic flow past a complete aircraft, including flow through the engines.

Most of the methods hitherto proposed for calculating transonic flow, either by solving the potential equation or by solving the Euler equations, have been based on the use of rectangular cells to discretize the flow equations. This is essentially equivalent to the introduction of curvilinear coordinates and leads to a high degree structure in the mesh. For geometrically simple shapes such as airfoil sections and wings this does not pose a problem. Indeed the curvilinear structure in such meshes has usually been exploited in the solution algorithm. It may help to determine the sweep direction, for example, or allow factorization of the iteration operator for ADI schemes, or in the case of multigrid methods facilitate the interpolation between different grid levels. For multiply connected regions such as the domain around a multi-element airfoil, and for complicated three dimensional regions, the structure imposed by rectangular cells becomes very restrictive. Methods based on the use of several different mesh blocks, each composed of a set of rectangular cells, have been proposed for dealing with such complex shapes. The difficulty of defining the mesh blocks, and ensuring the contiguity of mesh lines at the various interfaces is, however, considerable. Another approach that has been suggested is the use of separate overlapping .meshes, with the flow variables being interpolated between these meshes as the iterative solution proceeds. Like the multiblock formulation, this approach is likely to run into difficulties when applied to closely coupled components such as the nacelle/pylon/wing region of a modern commercial aircraft. Finally the use of nonaligned meshes which do not conform to the shape of the configuration has been proposed, and some success has been reported for flow computations involving relatively simple shapes. Nevertheless, the application of the solid wall boundary conditions leads to complex interpolation formulas, and there remains some doubt about the accuracy obtainable from such a procedure.

We believe that for complicated three dimensional shapes the imposition of mesh structure, implied by the use of rectangular cells, leads to restrictions which outweigh its benefits. This has motivated us to develop a method based on an entirely unstructured mesh of triangles in two dimensions or tetrahedra in three dimensions. The difficulties of generating a triangulation for a complete aircraft and achieving a satisfactory flow solution are still formidable. Nevertheless this line of attack is free of many of the problems that have plagued other approaches.

Separate meshes are generated around the individual aircraft components to create a cluster of points surrounding the whole aircraft. We do not require any regularity in the initial point distribution, only that a

*Reprinted from AIAA-86-1003 paper, Reno, NV, 1986

reasonable point density is created corresponding to the anticipated variation in the flowfield. The swarm of mesh points is then connected together to form a mesh of tetrahedral cells that provide the basis for a single finite element approximation for the entire domain. This use of triangulation to unify separately generated meshes bypasses the need to devise interpolation procedures for transferring information between overlapping grids.

The triangulation of a set of points to form disjoint tetrahedra is in general nonunique. Our procedure is to generate the Delaunay triangulation[7,8,9,10]. This is dual to the Voronoi diagram which results from a division of the domain into polyhedral neighborhoods, each consisting of the subdomain of points nearer to a given mesh point than any other mesh point. To avoid difficulties in the triangulation of a nonconvex region, it proves convenient to allow the triangulation of both the interior and the exterior of the configuration. The principal difficulty is then to ensure the capture of the surface of the configuration, and to prevent the formation of tetrahedra which break through from the exterior to the interior.

The finite element approximation is obtained by directly approximating the integral equations for the balance of mass, momentum and energy in polyhedral control volumes. Each of these is formed by the union of the tetrahedra meeting at a common vertex. It turns out that the flux balance can be broken down into contributions of fluxes through faces in a very elegant way. This novel decomposition reduces the evaluation of the Euler equations to a single main loop over the faces. It can be shown that the scheme can be reformulated so that it is essentially equivalent to a Galerkin method, but the present formulation leads to a substantial reduction of the computational complexity. The algorithm can be organized to permit vectorization on a vector computer by separating the faces into groups such that no vertex at which a flux balance is being accumulated is referenced more than once in each group.

Shock waves are captured with the assistance of added artificial dissipation [1,11]. A variety of options for the introduction of dissipation are under investigation, including an upwind option based on maintaining positivity of the coefficients [12].

Steady state solutions are obtained by integrating the time dependent equations with a multistage time stepping scheme. Convergence is accelerated by the use of locally varying time steps, residual averaging, and enthalpy damping [11]. We anticipate the use of multiple grids [11,13] in a later version of the program.

Finite Element Approximation

Let $p, \rho, u, v, w, E,$ and H denote the pressure, density, Cartesian velocity components, total energy, and total enthalpy, For a perfect gas

$$E = \frac{p}{(\gamma-1)\rho} + \frac{1}{2}(U^2+v^2+w^2), \qquad H = E + \frac{p}{\rho}$$

where γ is the ratio of the specific heats. The Euler equations for flow of a compressible inviscid fluid can be written in integral form as

$$\frac{\partial}{\partial t}\int\int\int_{\Omega} w \; d\Omega + \int\int_{d\Omega} \mathbf{F}\cdot d\mathbf{S} = 0 \tag{1}$$

for a domain Ω with boundary $\partial\Omega$ and directed surface element $d\mathbf{S}$. Here w represents the conserved quantity and $\mathbf{F}$ is the corresponding flux. For mass conservation

$$w = \rho, \qquad \mathbf{F} = (\rho u, \rho v, \rho w)$$

For momentum conservation

$$w = \rho u, \qquad \mathbf{F} = (\rho u^2 + p, \rho uv, \rho uw)$$

with y and z momentum quantities similarly defined, and for energy conservation

$$w = pE, \quad \mathbf{F} = (\rho Hu, \rho Hv, \rho Hw)$$

Equation (1) can be approximated on a tetrahedral mesh by first writing the flux balance for each tetrahedron assuming $\mathbf{F}$ to vary linearly over each face. Then at any given mesh point one considers the rate of change of w for a control volume consisting of the union of the tetrahedra meeting at a common vertex. This gives

$$\frac{d}{dt}\left(\sum_k V_k\right) w + \sum_k R_k = 0 \tag{2}$$

where V_k is the volume of the k^{th} tetrahedron meeting at a given mesh point and R_k is the flux balance of that tetrahedron.

When the flux balances of neighboring tetrahedra are summed all contributions across interior faces cancel. Equation (2) can therefore be rewritten as

$$\frac{d}{dt}\left(\sum_k V_k\right) w + \sum_k \mathbf{F_k} \cdot \mathbf{S_k} = 0 \tag{3}$$

where the second sum is over the external faces of the polyhedron formed from the union of the tetrahedra. Referring to Figure 1, which illustrates a two dimensional mesh, it may be seen that with a triangular or tetrahedral mesh, each face is a common external boundary of exactly two control volumes. Therefore each internal face can be associated with a set of 5 mesh points consisting of its three corners 1, 2 and 3, and the vertices 4 and 5 of the two tetrahedra based on the face, as illustrated in Figure (2). Vertices 4 and 5 are the centers of the two control volumes influenced by the face. It is now possible to generate the approximation (3) by presetting the flux balance at each mesh point to zero, and then performing a single loop over the faces. For each face one first calculates the fluxes of mass, momentum and energy across the face, and then one assigns these contributions to the vertices 4 and 5 with positive and negative signs respectively. Since every contribution is transferred from one control volume into another, all quantities are perfectly conserved. Mesh points on the inner and outer boundaries lie on the surface of their own control volumes, and the accumulation of the flux balance in these volumes has to be correspondingly modified. At a solid surface it is also necessary to enforce the boundary condition that there is no convective flux through the faces contained in the surface.

In order to illustrate the correspondence of this procedure with a Galerkin method consider the differential form of equation (1)

$$\frac{dw}{dt} + \nabla \cdot \mathbf{F} = 0$$

Multiplying by a test function 4, and integrating by parts over space leads to

$$\frac{\partial}{\partial t}\int\int\int_\Omega \phi w \; d\Omega = \int\int\int_\Omega \mathbf{F}\nabla\cdot\phi \; d\Omega - \int\int_{\partial\Omega} \phi\mathbf{F}\cdot d\mathbf{S} \tag{4}$$

Suppose now that we take ϕ to be the piecewise linear function with the value unity at one node (denoted by 0 in Figure 3., and zero at all other nodes. Then the last term vanishes except in the case when 0 is adjacent to the boundary. Also $\nabla\phi$ is constant in every tetrahedron, and differs from zero only in the tetrahedra with a common vertex at node 0. Since ϕ_x is constant in a tetrahedron it may be evaluated as

$$\phi_x = \frac{1}{V}\int\int\int \phi_x dxdydz = \frac{1}{V}\sum_k S_{x_k}\overline{\phi_{\mathbf{k}}}$$

where V is the cell volume, S_{x_k} and $\overline{\phi_{\mathbf{k}}}$ are the projected area of the k^{th} face in the x direction and the average value of ϕ on the k^{th} face, and the sum is taken over the faces of the tetrahedron. For the given test function $\overline{\phi} = 1/3$ on the faces 012, 023, and 031 and zero on the face 123. Also the projected area S_x on face 123 is equal and opposite to the sum of the projected face areas of the other three faces. Using the same procedure to evaluate ϕ_y and ϕ_z, it follows that

$$\nabla\phi = \mathbf{S}/3V \tag{5}$$

where $\mathbf{S}$ is the directed area of the face opposite vertex 0. Now treat $\mathbf{F}$ as piecewise linear and use equation (5) to evaluate the volume integral on the right side of equation (4). Then each tetrahedron meeting at node 0 introduces a contribution $\mathbf{F_k}.\mathbf{S_k}$, where F_k is the average value of F in cell k. For the cell illustrated in Figure 3, for example,

$$\mathbf{F} = \frac{1}{4}(F_0 + F_1 + F_2 + F_3)$$

Since the control volume is closed, however,

$$\sum_k \mathbf{S_k} = 0$$

Therefore the contribution of F_0 to F_k can be discarded, leading to the sum over the faces appearing in equation (3), multiplied by a constant. Thus the Galerkin method corresponds to the use of a weighted average of neighbors to evaluate the time derivative in equation (3). If one takes w to be piecewise linear then F, which is a nonlinear function of w, is not piecewise linear, and accordingly the face sum in equation (3) represents an approximate integration formula.

Dissipation

Equation (3) represents a nondissipative approximation to the Euler equations. Dissipative terms may be needed for two reasons; to eliminate the occurrence of undamped or lightly damped nodes, and to prevent oscillations near shock waves.

Consider an equation of the form (1) for one dependent variable. Figure (4a) illustrates an undamped mode permitted by the approximation (3) at interior nodes of a two dimensional mesh of equilateral triangles. The solution w may take three arbitrary values w_a, w_b and w_c, while the flux balance sums to zero at every interior control volume. In order to obtain a flux balance in control volumes along the boundary, however, as illustrated in Figure (4b), it is necessary that

$$\begin{aligned} f(w_a) &= \tfrac{1}{2}(f(w_b)+f(w_c)) \\ f(w_b) &= \tfrac{1}{2}(f(w_c)+f(w_a)) \\ f(w_c) &= \tfrac{1}{2}(f(w_a)+f(w_b)) \end{aligned}$$

whence

$$f(w_a) = f(w_b) = f(w_c)$$

and the mode is suppressed.

The first requirement of a shock capturing scheme is that it should recover the Rankine Hugoniot jump conditions. Using arguments similar to those employed by Lax and Wendroff[14], it may be shown that the conservative form of equation (3) ensures the recovery of the proper jump conditions as long as the approximation converges as the mesh spacing is reduced. In the absence of added dissipation, however, we can anticipate the appearance of oscillations in the neighborhood of shockwaves.

A variety of forms of dissipation are under investigation:

Accumulated Edge Differences

A simple way to introduce dissipation is to add a term generated from the difference between the value at a given node and its nearest neighbors. That is, at node 0, we add a term

$$D_0 = \sum_k \varepsilon_{k0}^{(1)}(w_k - w_0) \tag{6}$$

where the sum is over, the nearest neighbors, as illustrated in Figure 5. The contribution $\varepsilon_{k0}^{(1)}(w_k - w_0)$ is balanced by a corresponding contribution $\varepsilon_{ko}(w_o - w_k)$ at node k, with the result that the scheme remains conservative. The coefficients $\varepsilon_{k0}^{(1)}$ may incorporate metric information depending on local cell volumes and face areas, and can also be adapted to gradients of the solution. Formula (6) is no better than first order accurate unless the coefficients are proportional to the mesh spacing. A more accurate scheme is obtained by recycling the edge differencing procedure. After first setting

$$E_0 = \sum_k (w_k - w_0) \tag{7a}$$

at every mesh point, one then sets

$$D_0 = -\sum_k \varepsilon_{k0}^{(2)}(E_k - E_0) \tag{7b}$$

An effective scheme is produced by blending formulas (6) and (7), and adapting $\varepsilon_{0k}^{(1)}$ to the local pressure gradient. This is accomplished by calculating

$$P_0 = \sum_k \left| \frac{p_k - p_0}{p_k + p_0} \right|$$

at every mesh point, and then taking $\varepsilon_{0k}^{(1)}$ proportional to $\max(P_0, P_k)$. Formulas of this type have been found to have good shock capturing properties, and the required sums can be efficiently assembled by loops over the edges.

Finite Element Approximation to Scaled Laplacian Operator

An alternative way to introduce dissipation is to construct a finite element approximation using piecewise linear elements to terms of the form

$$\frac{\partial}{\partial x}\varepsilon_x \frac{\partial w}{\partial x} + \frac{\partial}{\partial y}\varepsilon_y \frac{\partial w}{\partial y} + \frac{\partial}{\partial z}\varepsilon_z \frac{\partial w}{\partial z}$$

where the coefficients $\varepsilon_x, \varepsilon_y,$ and ε_z are adapted to the local solution. These formulas can be assembled by loops over the cells and are more expensive to evaluate than the schemes derived from edge differences, but may allow more precise control of the dissipation.

Schemes with Positive Coefficients

A third approach stems from an analysis of criteria for the design of nonoscillatory schemes for scalar conservation laws. The idea may be illustrated by considering a general scalar conservation law in two space dimensions of the form

$$\frac{dw}{dt} + \frac{\partial}{\partial x} f(w) + \frac{\partial}{\partial y} g(w) = 0 \tag{8}$$

Suppose that this is approximated by a semi-discrete scheme of the form

$$\frac{d}{dt} w_0 = \sum_k c_{k0}(w_k - w_0) \tag{9}$$

at node 0, where the coefficients c_{k0} are non-negative. Then if there is a maximum at 0, the right side cannot be positive and w_0 cannot increase. Thus the condition of non-negative coefficients ensures that new local maxima and minima cannot be created, yielding a nonoscillatory scheme. It corresponds to the conditions given by Jameson and Lax for multipoint total variation diminishing schemes in one space dimension[15].

The desired positive coefficients can be realized by adding dissipation of the form described by equation (6). Figure 6 illustrates the application of the method described in Section 2 to Equation (8). Equation (3) here takes the form

$$S\frac{dw_0}{dt} + \sum_k (\Delta y f - \Delta x g)_{k+1/2} = 0 \tag{10}$$

where S is the area of the polygon enclosing the vertex 0, and the subscript $k + 1/2$ indicates a mean value on the edge joining k to $k + l$.

Equation (10) can be rearranged as

$$S\frac{dw_0}{dt} = -\sum_k \{\Delta y_k (f_k - f_0) - \Delta x_k (g_k - g_0)\}$$

where

$$\Delta x_k = \frac{1}{2}(\Delta x_{k+1/2} + \Delta x_{k-1/2}) = \frac{1}{2}(x_{k+1} - x_{k-1})$$

and

$$\Delta y_k = \frac{1}{2}(\Delta y_{k+1/2} + \Delta y_{k-1/2}) = \frac{1}{2}(y_{k+1} - y_{k-1})$$

Let

$$d_{k_0} = \begin{cases} \Delta y_k \dfrac{f_k - f_0}{w_k - w_0} - \Delta x_k \dfrac{g_k - g_0}{w_k - w_0} & , \quad w_k = w_0 \\ \Delta y_k \dfrac{\partial f}{\partial w}\bigg|_0 - \Delta x_l \dfrac{\partial g}{\partial w}\bigg|_0 & , \quad w_k \neq w_0 \end{cases}$$

Now let artificial dissipation be added in the form

$$\varepsilon_{k0}(w_k - w_0) \text{ along the edge } k0, \varepsilon_{k0} > |d_{k0}|$$

Then the resulting scheme has the form (9) with nonnegative coefficients.

This method of construction can be applied to a system of equations through the introduction of flux splitting. As in the scalar case, one can break down the flux balance at the vertex 0 to a sum of contributions

$$(f_k - f_0)\Delta y_k - (g_k - g_0)\Delta x_k$$

where $f(w)$ and $g(w)$ are now vectors. Then, following Roe [16], one can introduce a matrix C_{k0} suck that

$$C_{k0}(w_k - w_0) = (f_k - f_0)\Delta y_k - (g_k - g_0)\Delta x_k$$

C_{k0} can be expressed as

$$C_{k0} = T\Lambda T^{-1}$$

where Λ is a diagonal matrix of eigenvalues Λ_1 and the columns of T are the corresponding eigenvectors. Then one introduces a dissipative flux

$$TMT^{-1}(w_k - w_0)$$

along the edge $k0$, where M is a diagonal matrix with elements $\mu_i > |\lambda_i|$.

The addition of dissipative terms constructed along these lines leads to schemes which are no better than first order accurate. To improve the accuracy the dissipative fluxes should either be adaptively switched on only where they are needed in the neighborhood of discontinuities, or else compensated by the addition of antidiffusive terms [17]

Integration to a steady State

The discretization procedure of Sections 2 and 3 lead to a set of coupled ordinary differential equations which can be written in the form

$$\frac{dw}{dt} + R(w) = 0 \tag{11}$$

where w is the vector of the flow variables at the mesh points, and $R(w)$ is the vector of the residuals, consisting of the flux balances defined by equation (3), together with the added dissipative terms. These are to be integrated until they reach a steady state.

For this purpose we use a multistage time stepping scheme of the same type which has proved effective in calculations on rectilinear meshes. Let w^n be the result after n steps. To advance one step Δt with an m stage scheme we set

$$\begin{aligned} w^{(0)} &= w^n \\ w^{(1)} &= w^{(0)} - \alpha_1 \Delta t R^{(0)} \\ &\cdots \\ w^{(m-1)} &= w^{(0)} - \alpha_{m-1} \Delta t R^{(m-2)} \\ w^{n+1} &= w^{(m)} \end{aligned}$$

The residual in the (q+1)st stage is evaluated as

$$R^{(q)} = \frac{1}{V} \sum_{r=0}^{q} \left\{ \beta_{qr} Q(w^{(r)}) - \gamma_{qr} D(w^{(r)}) \right\} \tag{12}$$

where $Q(w)$ is the approximation to the Euler equations and $D(w)$ represents the dissipative terms, and the coefficients β_{qr} and γ_{qr} satisfy the consistency condition that

$$\sum_{r=0}^{q} \beta_{qr} = \sum_{r=0}^{q} \gamma_{qr} = 1 \tag{13}$$

In practice a 3 stage scheme has proved effective. For this scheme

$$\begin{aligned} \alpha_1 &= .6 \quad , \quad \alpha_2 = .6 \\ \beta_{qq} &= 1 \quad , \quad \beta_{qr} = 0 \quad , \quad q > r \\ \gamma_{q0} &= 1 \quad , \quad \gamma_{qr} = 0 \quad , \quad r > 0 \end{aligned}$$

Convergence to a steady state is accelerated by using a variable time step close to the stability limit at each mesh point. The scheme is accelerated further by the introduction of residual averaging [2]. At the mesh point 0 the residual R_0 is replaced $\widetilde{R_0}$, where $\widetilde{R_0}$ is an approximation to the solution R_0 of the equation

$$\bar{R_0} + \sum_k \varepsilon(\bar{R_0} - \bar{R_k}) = R_0 \tag{14}$$

in which the sum is over the nearest neighbors. This is similar to the weighted average appearing in the Galerkin method, but with the opposite sign for the coefficient ε, leading to an increase in the permissible time step instead of a reduction. In practice it has been found effective to use two steps of the Jacobi iteration

$$\widetilde{R}_0^{(m)} + \sum_k \varepsilon(\widetilde{R}_k^{(m-1)} - \widetilde{R}_k^{(m-1)}) = R_0 \tag{15}$$

starting from $\widetilde{R}_0^{(0)} = R_0$

Mesh Generation

The generality of our approach allows us to generate separate meshes for each aircraft component. The mesh points from several overlapping meshes can then be united to form a single cloud of points which can be connected to form tetrahedral cells. For our preliminary calculations we have made use of the sequential mapping technique [18] to generate a mesh around the combination of wing/body/tail/fin. We next use a similar combination of conformal mapping plus shearing to create a mesh around the isolated nacelles and on the pylon surfaces.

In order to describe the mapping for an isolated wing, consider the point $\mathbf{x} = (x, y, z)$ in physical space and denote the wing surface by the set of points $W = \{\mathbf{x_w}\}$. We require a transformation that will generate a C-mesh around the wing. In order to keep the mesh nearly orthogonal we define a parabolic unwrapping of the wing into a shape having everywhere small curvature. Let $\mathbf{X} = (X, Y, Z)$ be the point in mapped space corresponding to $\mathbf{x}$ in physical space. The parabolic unwrapping is given by the transformation

$$\mathbf{X} = P_w \mathbf{x}$$

where P_w is defined by

$$\begin{aligned} x - x_0(z) &= X^2 - Y^2 \\ y - y_0(z) &= 2XY \\ z &= Z \end{aligned}$$

and $\mathbf{x_0}$ is a point just inside the wing leading edge. Let $Y_w(X, Z)$ be the surface of the wing in mapped space and define a shearing S_w taking $\mathbf{X}' = (X', Y', Z')$ to $\mathbf{X}$ by the transformation

$$\begin{aligned} \mathbf{X} &= S_w \mathbf{X}' \\ X &= X' \\ Y &= Y' + Y_w(X, Z) \\ Z &= Z' \end{aligned}$$

This maps the half space $Y' \geq 0$ onto the region in $\mathbf{X}$-space above Y_w.

We may now suppose that a stretching transformation T is sued to map the unit cube onto $\mathbf{X}'$ space. We can summarize the procedure as a mapping

$$\mathbf{X_w} = P_w \mathbf{x_w}$$

to take the wing surface into $\mathbf{X}$ space, followed by the mapping sequence

$$\mathbf{x} = P_w^{-1} S_w T \boldsymbol{\xi}$$

which takes a point $\boldsymbol{\xi}$ in the unit cube to the point $\mathbf{x}$ in physical space.

The mesh around a combination of a wing plus body is generated by introducing a further transformation B which maps an arbitrary shape body into the symmetry place, $z = 0$. This mapping can be constructed as a combination of a Joukowski mapping plus shearing. We can then summarize the procedure as

$$\mathbf{X_w} = P_w B \mathbf{x_w}$$

to determine the wing geometry in mapped space followed by

$$\mathbf{x} = B^{-1} P_w^{-1} S_w T \boldsymbol{\xi}$$

This sequence of operations will generate a mesh that conforms with the body surface but such that the crest line of the body is not necessarily aligned with any mesh line. This deficiency is rectified by deforming the mesh lines in mapped space to ensure that the resulting mesh is completely aligned with the body surface. Further details are given in reference [18].

The extension of these ideas to include a tail and fin follows the same principle of first utilizing a mapping to simplify the configuration, fitting a mesh in mapped space and then mapping back to obtain the mesh in physical space.

In the neighborhood of the tail we observe that the mesh has locally the structure of an H-mesh. In order to fit an H-mesh around an airfoil, say, we can apply a parabolic unwrapping, similar to that described for the wing to both the airfoil geometry and a set of Cartesian coordinates in physical space. Under the parabolic unwrapping these coordinate lines map to a family of hyperbolae. We can now apply a shearing transformation-to each point above the mapped airfoil surface in such a way that the X axis maps onto the mapped airfoil profile. The inverse

parabolic mapping will now return the airfoil shape in physical space and create an H-mesh that is aligned with the airfoil surface.

A simple modification of this procedure can be used to invert a horizontal tail. We assume that the tail is defined by the set of points $\{\mathbf{x_T}\}$ and first apply the body mapping to the wing point set $\{\mathbf{x_w}\}$ and the tail point set $\{\mathbf{x_T}\}$. This, in effect, transforms the combination of wing/body/tail into a distorted wing plus a distorted tail. We now apply the transformation P_T which is a parabolic unwrapping about a singular line just inside the tail leading edge.

We then transform the mapped wing points using a tail shearing S_T, invert the mapping P_T and next apply the wing parabolic unwrapping to obtain the final mapped surface

$$\mathbf{X_w} = P_w P_T^{-1} S_T^{-1} P_T B \mathbf{x_w}$$

We now apply the wing shearing transformation, invert the wing parabolic unwrapping, inflate the tail plane and invert the body mapping. This can be represented by the mapping sequence

$$\mathbf{x} = B^{-1} P_T^{-1} S_T P_T P_w^{-1} S_w \mathbf{X}'$$

The fin is treated in a similar way, by first using a parabolic unwrapping P_F and shearing S_F to map the fin into the symmetry plane.

The complete mapping sequence for the combination of wing/body/tail/fin can be summarized as a mapping of the wing surface

$$\mathbf{X_w} = P_w P_T^{-1} S_T^{-1} P_T B P_F^{-1} S_F^{-1} P_F \mathbf{x_w},$$

the setting up of a stretching transformation T in mapped space followed by the reverse mapping procedure

$$\mathbf{X}' = T\boldsymbol{\xi}$$

$$\mathbf{x} = P_F^{-1} S_F P_F B^{-1} P_T^{-1} S_T P_T P_W^{-1} S_W \mathbf{X}'$$

The nacelle mesh is also generated by a combination of unwrapping plus shearing. In this case we define a mapping P_N by the conformal transformation

$$\xi = z - e^{-z}$$

where $z = x + iy$ and $\xi = X + iY$. Here x is a coordinate aligned with the nacelle axis, y is the radial coordinate corresponding to a cylindrical coordinate system such that $y = 0$ is the nacelle axis. If the nacelle is not axisymmetric we take the axis of the cylindrical coordinate system to be an approximate center line through the nacelle. The z coordinates are scaled so that $y = \pi$ corresponds to a cut inside the nacelle section with the point $(0, i\pi)$ just inside the section leading edge. The above mapping, applied to each nacelle section transforms the space around the nacelle onto the space inside a deformed cylinder with the nacelle surface mapped to the cylinder surface.

A shearing transformation can now be combined with the inverse of the above mapping to generate a mesh that is aligned with the nacelle surface. A straightforward extension of the sequential mapping procedure can be used to accommodate a center body.

Finally we can generate points around the pylons by treating each pylon as an isolated wing and using the mapping sequence that has previously been described.

Delaunay Triangulation

The Delaunay triangulation of a set of points is a unique construct that connects the points to form a covering of space by a collection of disjoint simplexes. The geometric dual of the Delaunay triangulation is the Voronoi diagram. This concept has been rediscovered several times and goes under a variety of different names (Dirichlet tessellation, Theissen regions and Wigner-Seitz cells). We believe, however, that its application as a method of mesh generation for computational aerodynamics is a new departure.

If the set of points is denoted by $\{P_i\}$, the Voronoi neighborhood of the point P_i is defined as the region of space

$$V_i = \{d(x, P_i) < d(x, P_j) \text{ for all } i \neq j\}$$

Here $x \quad \varepsilon \quad E_n$, n dimensional Euclidean space and d is the Euclidean metric. However, the concept generalizes to other metric spaces. Each such region V_i is the intersection of the open half plans bounded by the perpendicular bisectors of the lines joining P_i to each of the other P_j. The regions are thus convex polyhedra and, in general,

$n+1$ such regions meet at each vertex of the Voronoi diagram where n is the number of dimensions. We refer to regions that have common boundary faces as contiguous and likewise-denote the points associated with two such regions as contiguous points. For each vertex of the Voronoi diagram we can join the $n+1$ contiguous points, which have that vertex in common, by $n+1$ hyperplanes to form an n simplex (a triangle in two dimensions, a tetrahedron in three dimensions). The aggregate of simplexes forms the unique triangulation of the convex hull of points P_i which we call the Delaunay triangulation. Each vertex is the circumcenter of the simplex with which it is associated, and the above construction ensures that no other point lies within the sphere that circumscribes the simplex. This property ensures that the aspect ratio of the simplexes is reasonable and in some sense leads to the optimum triangulation for a given distribution of points.

The computation of the Voronoi diagram and its associated triangulation has received considerable attention recently [7,8,9]. The algorithm used here is based on Bowyer's method[8] and has been successfully applied in both two and three dimensions. To clarify the presentation we shall describe the algorithm for the specific case of three dimensional space, so that at each vertex there are four regions meeting and the four forming points associated with these regions form a tetrahedron. As Bowyer notes, it is possible to record the structure of the triangulation by constructing two lists for each vertex in the structure. Each list has four entries; the first contains the forming points of the tetrahedron associated with the vertex and the second list holds the addresses of the neighboring vertices.

The algorithm is described in more detail elsewhere[16]; here we will briefly describe the steps. The process is sequential; each new point is introduced into the existing structure which is broken and then reconnected to form a new Delaunay triangulation. When a new point is introduced into the existing triangulation, it is first necessary to identify a vertex of the Voronoi diagram that will be deleted by the new point. As the vertex at the circumcenter of the tetrahedron in which the point lies must necessarily be deleted, we are assured that at least one deleted vertex can be identified. Next we look at the neighbors of the deleted vertex for other vertices of the Voronoi diagram that may be deleted. We continue the tree search until all deleted vertices have been identified. The deleted region must necessarily be simply connected and star shaped. We then join the new point to the points which lie in Voronoi neighborhoods contiguous with the deleted region. It remains to label the new Voronoi neighborhoods and revise the lists that record the data structure.

The initial step that determines the deleted vertices requires the distance between the new point and a vertex of the Voronoi diagram to be compared with the radius of the circumscribing sphere for the tetrahedron associated with that vertex. The only other floating point operations required in this algorithm concern the computation of the new vertex coordinates (i.e. the center of the circumscribing sphere) and the radius of each new vertex. All other operations are of an entirely logical nature involving searches and manipulations of the data structure that records the Voronoi diagram and its associated Delaunay triangulation.

Although the Delaunay triangulation of a given set of points is unique, the computational effort required clearly depends on the number of vertices that must be tested before the first deleted vertex is found, as well as the size of the deleted region that must be reconfigured after the introduction of each new point. The search time required to find the first deleted vertex can probably be reduced substantially by periodic sorting of the vertex list to ensure that the vertex search starts close to where one expects to find the first deleted vertex. The size of the deleted region that must be reconfigured depends critically on the order in which the new points are introduced. A good strategy is to introduce a coarse sprinkling of widely separated points and then introduce further points by what is essentially a mesh refinement procedure. It should also be possible to arrange the introduction of points so that different regions of the flowfield are completely triangulated and would not be reconfigured by the introduction of points in another region. This would enable one to restrict the search and reconfiguration to a subset of the Voronoi diagram. There are further difficulties in the application of this triangulation procedure to generate a mesh around aircraft components. It is necessary to ensure that the surface remains intact and that no connections are formed which would pass through any component surface. It turns out that it is convenient to allow the triangulation to proceed throughout the whole space, both inside and outside the aircraft. The tetrahedra that lie entirely inside any component can then be removed to leave the external triangulation that is required by the flow solver. To ensure the integrity of the aircraft surface, it is necessary to introduce extra surface points to prevent any connections being made through the surface. At present this is done by placing a dense distribution of points over all surfaces and rejecting external points that would force connections that break through the surface. Our immediate concern is to improve this procedure by adding extra surface points wherever they are required in an adaptive fashion.

Preliminary Results

In order to demonstrate the accuracy that can be achieved by the use of our finite element method on triangular meshes, we first present a two dimensional result. The example is a NACA 0012 airfoil at zero angle of attack

and a Mach number of 0.8. This particular case has been computed by several different methods and there is now a fairly clear consensus of what should be the correct inviscid solution. The mesh for this case was obtained from a direct triangulation based on an 0-mesh around the airfoil. The surface pressure distribution is presented in Figure 7. A smooth surface pressure distribution has clearly been obtained with a sharp shock profile. The computed drag coefficient of 0.0087 is within two or three drag counts of the drag obtained by other robust and accurate Euler methods. The evidence of this result suggests that our new flow solver is at least as accurate as the finite volume method based on rectangular cells.

However, the real motivation for using a flow solver based on triangular or tetrahedral elements lies in the possibility of calculating the inviscid flow over complex configurations. The remaining figures present a preliminary result for the flow over a Boeing 747-200 including engine nacelles and pylons. The surface mesh is shown in Figure 8 and we have endeavored to achieve a fairly high density of points over the entire aircraft surface. The mesh was generated by. the technique described earlier and triangulated by the Delaunay procedure. We have Pot yet examined the triangulation in the flowfield in order to see the variety of connections that are made and the aspect ratio of the tetrahedra that have been formed. We anticipate, however, that future developments will lead to better control of the triangulation and a higher degree of regularity in the mesh. Nevertheless we have carried out what we believe to be the first Euler calculation for a complete aircraft. The mesh used for this calculation contained 12038 nodes and 57914 tetrahedra. The flow solution was obtained using a three stage time stepping scheme with implicit residual averaging. The average residual was reduced by more than six decades in 500 steps (from .378 102 to .147 10-4). The lift and drag coefficients were frozen to three figures after 200 steps, which seem to be enough for the convergence of all significant features of the flow field. With this mesh a run of 200 steps takes about 10 minutes on a Cray XMP computer. Surface pressure contours are presented in Figures 9(a), (b), (c) and (d). Although we are not claiming great accuracy for this preliminary calculation, it is clear that the important flow features are predicted, including the interference effects of the wing and tail on the body, and the mutual interference of the wing nacelle and pylon.

Planned Developments

Now that the feasibility of calculating the flow past an aircraft by the present method has been established, we plan to pursue its development in a variety of directions designed to reduce the cost of the calculations, improve their accuracy, and extend their range of application. Some topics of investigation are outlined below:

1. Vectorization

   To avoid unnecessary computer costs it is important to take proper advantage of the vector and parallel processing capabilities of the computers now available. Vectorization of the main loops has already been achieved by separating the cells, faces and edges into groups such that no vertex at which contributions are being accumulated is referred to more than once in each group. Using this procedure rates of computation ranging from 17-38 megaflops have been realized on a Cray XMP computer, depending on the mesh. These variations stem from variations in the sizes of the groups and the associated vector lengths. The efficiency can be improved by making sure that no group is too small. The analysis of the associated sorting problems leads to some general map coloring problems: for example, what is the minimum number of colors needed to color the tetrahedra in such a way that tetrahedra meeting at the same vertex do not have the same color.

2. Improved Mesh Generating Procedures

   The present triangulation procedure is extremely time consuming. We anticipate that the time required to triangulate the mesh points can be drastically reduced by more sophisticated search procedures, improved protection methods to prevent the triangulation breaking through the surface, and a more favorable ordering of the introduction of the points into the triangulation. Vectorization of the triangulation procedure presents an interesting research problem. Other ways to reduce the triangulation cost include systematic subdivision of a preliminary coarse mesh generated by the Delaunay triangulation procedure, and the use of a hybrid mesh with an unstructured triangulation embedded in a structured triangular or rectangular mesh.

3. Improved Shock Capturing and Shock Fitting

   Careful control of the dissipation should lead to improved accuracy. The criterion of nonnegative coefficients proposed in Section 3 provides a guideline for the construction of flux split schemes using characteristic decomposition to produce just enough dissipation to avoid oscillations. The flexibility provided by the use of an unstructured tetrahedral mesh may also facilitate the development of effective shock fitting schemes.

4. Multigrid Acceleration

Eventually we hope to make a further reduction in the cost of the flow calculation by using multiple grids to accelerate the convergence to a steady state. This will require the development of efficient transfer procedures between the meshes.

5. Adaptive Mesh Refinement

   The unstructured tetrahedral mesh provides a natural setting for the introduction of an adaptive mesh refinement procedure in which additional mesh points are inserted in regions where there are rapid variations in the flow, or an indication of relatively large discretization error. This provides a method of reducing the thickness, for example, of a computed shock layer.

6. Extension to Navier Stokes Equations

   The viscous terms of the Navier Stokes equations can be approximated within the present framework by using piecewise linear elements following the standard finite element formulation.

7. Simulation of Engine Power Effects

   The present model includes flow through the engine nacelles. A more realistic simulation can be achieved by introducing source terms to represent the engine power effects.

Acknowledgements

Most of the development work leading up to our present computer program has been carried out on computers belonging to the Cray Research Corporation, who have provided us with access to their XMP 48 and XMP 216 at Mendota Heights. We are also deeply indebted to Kent Misegades, Jef Dawson and Janet Low for their help and support, to Kent Misegades in particular for converting our results to graphic output. We also wish to thank Satish Samant of the Boeing Company for providing us with the geometric data for the Boeing 747-200. Finally our work has benefited from substantial financial support provided by the IBM Corporation, the Office of Naval Research, and the NASA Langley Research Center.

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]Pulliam, T. H. and Steger, J.L., "Implicit Finite Difference Simulations of Three-Dimensional Compressible Flow", AIAA Journal, Vol. 18, 1980, pp. 159-167.

[3]Ni, R.H., "A Multiple Grid Scheme for Solving the Euler Equations", Proc. AIAA 5th Computational Fluid Dynamics Conference, Palo Alto, 1981, pp. 257-264.

[4]Jameson, A. and Baker, T.J., "Solution of the Euler Equations for Complex Configurations", AIAA 6th Computational Fluid Dynamics Conference, Danvers, MA, AIAA Paper 83-1929, July 1983.

[5]Jameson, A. and Baker, T.J., "Multigrid Solution of the Euler Equations for Aircraft Configurations", AIAA 22nd Aerospace Sciences Meeting, Reno, Nevada, AIAA Paper 84-0093, January 1984.

[6]Bristeau, M.O., Glowinski, R., Periaux, J., Perrier, P., Pironneau, 0. and Poirier, G., "On the Numerical Solution of Nonlinear Problems in Fluid Dynamics by Least Squares and Finite Element Methods (II). Application to Transonic Flow Simulations", Proceedings of the 3rd International Conference on Finite Elements in Nonlinear Mechanics, Stuttgart, West Germany, September 1984.

[7]Watson, D.F., "Computing the n-Dimensional Delaunay Tessellation with Application to Voronoi Polytopes", The Computer Journal, Vol. 24, No. 2, pp. 162-166.

[8]Bowyer, A., "Computing Dirichlet Tessellations", The Computer Journal, Vol. 24, No. 2, pp. 162-166.

[9]Weatherill, N.P., "The Generation of Unstructured Grids Using Dirichlet Tessellations", Princeton University, MAE Report No.1715, July 1985.

[10]Baker, T.J., "Delaunay Triangulation for Three Dimensional Mesh Generation", Princeton University, MAE Report No. 1733, December 1985.

[11]Jameson, A., "Transonic Flow Calculations", Princeton University, MAE Report No. 1651.

[12]Jameson, A., "Algorithms for Compressible Flow Calculations on Triangles and Tetrahedral Meshes", Princeton University MAE Report No. 1732, December 1985.

[13]Jameson, A., "Solution of the Euler Equations for Two-Dimensional Transonic Flow by a Multigrid Method", presented at the International Multigrid Conference, Copper Mountain, April 1983(see also Princeton Univ., MAE Report No. 1613, June 1983.

[14]Lax, P.D. and Wendroff, B., "Systems of Conservation Laws", Comm. Pure Appl. Math., Vol. 13, 1960, pp. 217-237.

[15]Jameson, A. and Lax, P.D., "Conditions for the Construction of Multi-quint Total Variation Diminishing Difference Schemes", Princeton University, MAE Report 1650, 1984.

[16]Roe, P.L., "Some Contributions to the Modelling of Discontinuous Flows", Proc. AMS/SIAM Summer Seminar on Large Scale Computation in Fluid Mechanics, San Diego, 1983.

[17]Harten, A., "High Resolution Schemes for Hyperbolic Conservation Laws", New York University Report DOE/ER 03077-175, 1982

[18]Baker, T.J., "Mesh Generation by a Sequence of Transformations ", to be published.

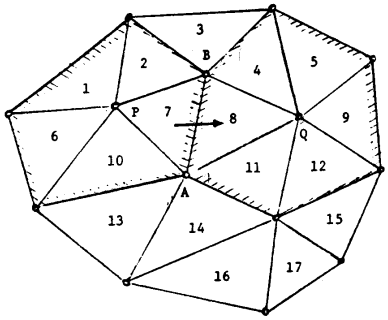

Fig. 1 A triangular mesh in 2 dimensions: The control volume at P is the union of triangles 1, 6, 10, 7 and 2, while that at Q is the union of triangles 4, 8, 11, 12, 9, and 5. The flux across the edge AB is from the control volume at P to the control volume at Q

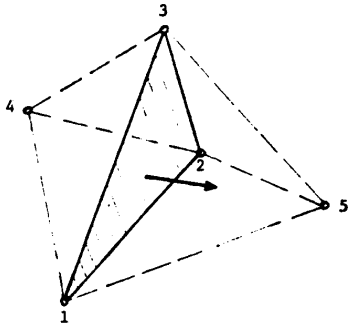

Fig. 2 Flux through face defined by nodes 1, 2 and 3 is out of the control volume centered at node 4, and into the control volume centered at node 5

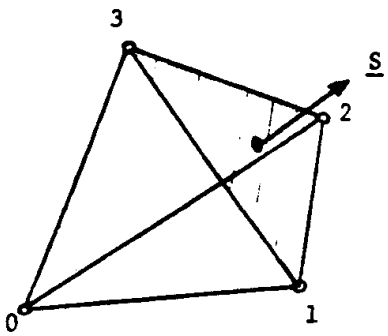

Fig. 3 One tetrahedron of the control volume centered at node 0.

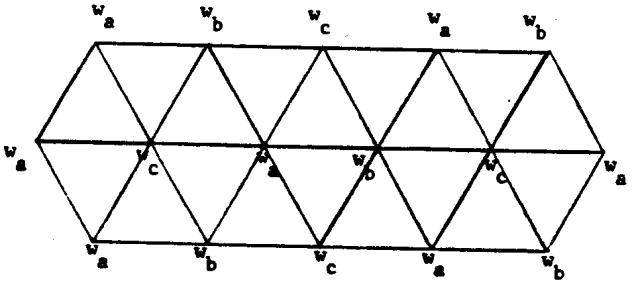

Figure 4(a)

Fluxes across every edge of control volume centered at P have the same mean values

$\bar{f} = 1/2(f(w_b)+f(w_c))$, $\bar{g} = 1/2(g(w_b)+g(w_c))$.

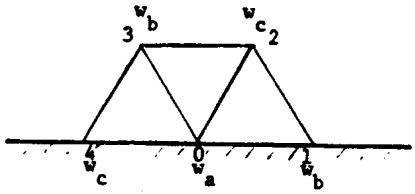

Figure 4(b)

Flux balance at 0 requires

$$\tfrac{1}{2}(f(w_b) + f(w_c))\ (\Delta y_{12} + \Delta y_{23} + \Delta y_{34})$$

$$= \tfrac{1}{2}(f(w_c) + f(w_a))\ \Delta y_{40} + \tfrac{1}{2}\ (f(w_a) + f(w_b))\ \Delta y_{01}$$

whence

$$f(w_a)(\Delta y_{40} + \Delta y_{01}) = f(w_b)\ \Delta y_{40} + f(w_c)\ \Delta y_{01}$$

Fig. 4

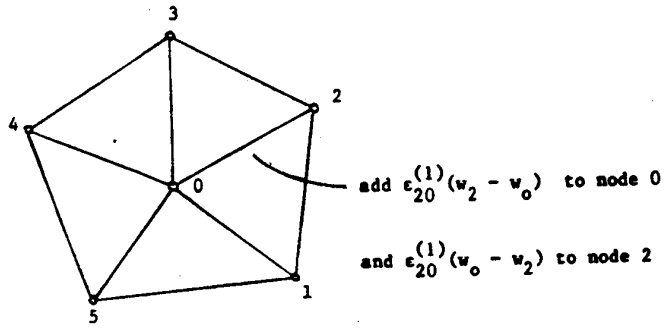

Fig. 5 Construction of dissipation from differences along edges in a two dimensional mesh

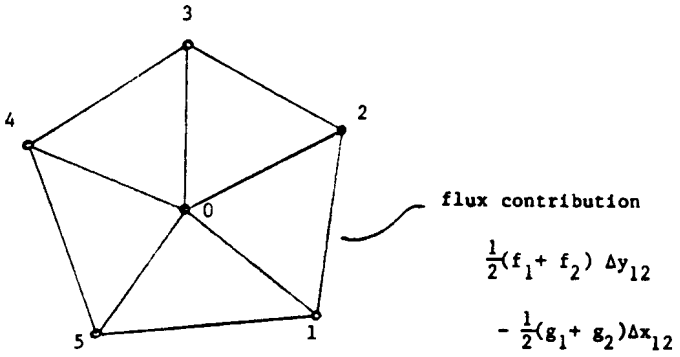

Fig. 6 Flux balance on a two dimensional mesh

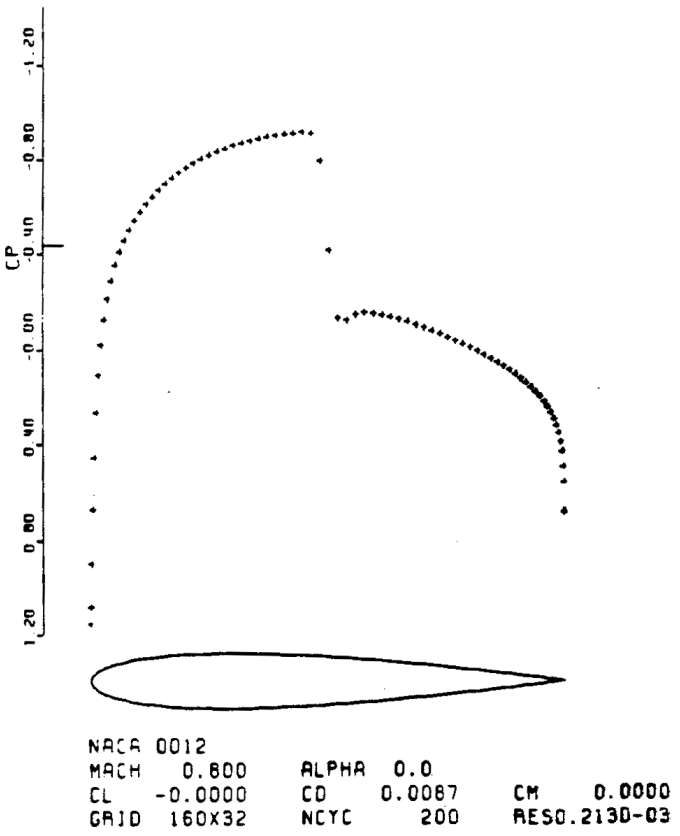

Fig. 7 Pressure Distribution for NACA 0012, M=0.8, $\alpha = 0^\circ$

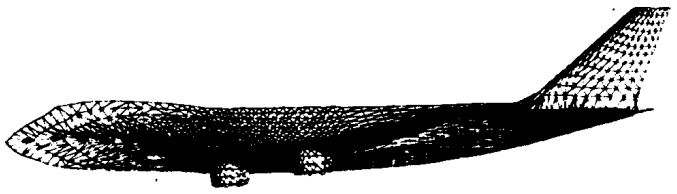

Fig. 8 Surface Mesh Distribution for the Boeing 747-200

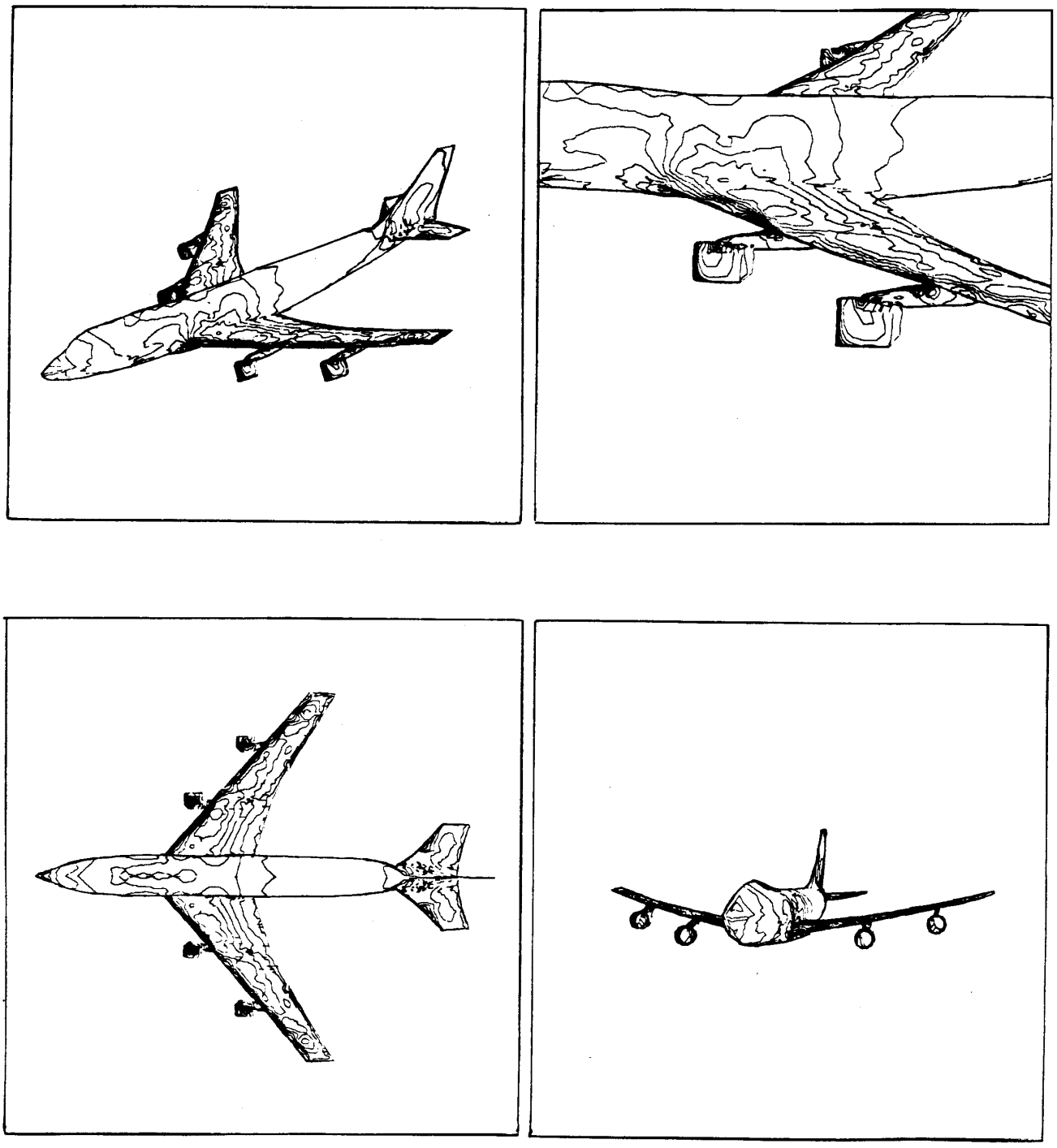

Fig. 9 Pressure Contours on the Aircraft Surface;M=0.84,$\alpha = 2.73^{\circ}$