A Novel Finite Element Method for the Calculation of Inviscid Flow Over a Complete Aircraft

by

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

* * * * *

Abstract

A method for generating tetrahedral meshes around complete aircraft has been developed and linked to an Euler flow solver. Details of this novel finite element method are described and results of a transonic flow calculation for a commercial aircraft are presented.

Presented at the Sixth International Symposium on Finite Element Methods in Flow Problems, Antibes, France, June 16-20, 1986 .

Introduction

The emergence of a method for calculating transonic flow over a complete aircraft has been severely hampered by the difficulty of generating a suitable mesh. Finite difference and finite volume methods for solving flow problems have traditionally used structured meshes based on a set of rectilinear cells[1,2,3,4,5]. For airfoil sections and simple 3-D shapes, it is possible to generate rectilinear meshes without too much difficulty. For more complicated shapes, however, it becomes increasingly difficult to produce a structured mesh that is aligned with all solid surfaces. The generation of a structured mesh that wraps around the wing, pylon and engine nacelle is particularly difficult, and there appears to be no straightforward way to achieve this goal with the numerical and algebraic mesh generation techniques currently available.

Multiblock methods using several different blocks of rectilinear cells have been proposed for dealing with such regions. This approach partitions the mesh into a collection of smaller blocks so that the mesh generation problem in each individual block is simplified. The difficulty of defining the mesh blocks, and ensuring contiguity of mesh lines at the various interfaces is still considerable.

An alternative procedure is the use of tetrahedral cells leading to an unstructured mesh that can negotiate the complicated changes in surface shape. Finite element methods based on triangular cells in two dimensions and tetrahedral cells in three dimensions have been developed by several authors[6,7]. Nevertheless, generation of a tetrahedral mesh for the space around a complete aircraft is still a formidable problem. The work of Bristeau, Glowinski, Periaux, Perrier, Pironneau and Poirier[6] is therefore particularly striking and is, to our knowledge, the first and hitherto only successful demonstration of mesh generation for a complete aircraft.

In this paper we describe a new method[8], for generating a tetrahedral mesh, that connects an arbitrary cluster of points by a systematic procedure based on the Delaunay criterion. This is dual to the Voronoi diagram that results from a division of the domain into polyhedral neighborhoods, each consisting of the subdomain of points nearer to a given mesh point that any other mesh point. The

implementation of this method and the need to maintain the integrity of solid surfaces presents a number of interesting problems. Although the Delaunay triangulation and associated Voronoi diagram has been exploited by others as a natural setting for calculations involving irregularly spaced points[9,10], we believe that the use of the Delaunay criterion as an explicit method of generating meshes for complex shapes is a new departure.

The finite element approximation is obtained by directly approximating the integral equations for conservation of mass, momentum and energy in polyhedral control volumes. The scheme can be regarded as a Galerkin method in which the test function space is the set of piecewise linear tetrahedral elements. This can be shown to be equivalent to a flux balance based on polyhedral control volumes formed by the union of tetrahedra meeting at a common vertex. It turns out that each face is associated with precisely two such control volumes and it is therefore possible to reformulate the calculation in a particularly elegant way. This novel decomposition reduces the flux evaluation to a single main loop over the faces resulting in a substantial reduction in computational complexity. 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.

2. Finite Element Approximation

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

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

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

$$\frac{\partial}{\partial t}\iiint_{\Omega} w\,d\Omega + \iint_{\partial\Omega} \underline{F} \cdot d\,\underline{S} = 0 \qquad (1)$$

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

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

For momentum conservation

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

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

$$w = \rho E, \ \underline{F} = (\rho Hu, \ \rho Hv, \ \rho Hw)$$

Consider the differential form of equation (1)

$$\frac{\partial w}{\partial t} + \nabla \cdot \underline{F} = 0$$

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

$$\frac{\partial}{\partial t} \iiint_{\Omega} \phi w d\Omega = \iiint_{\Omega} \underline{F} \cdot \nabla \phi d\Omega - \iint_{\partial\Omega} \phi \underline{F} \cdot d\underline{S} \qquad (2)$$

Suppose now that we take ϕ to be the piecewise linear function with the value unity at one node (denoted by 0 in Figure 1), 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

$$\phi_x = \frac{1}{V} \iiint \phi_x \, dxdydz = \frac{1}{V} \sum_k S_{x_k} \overline{\phi}_k$$

where V is the cell volume, S_{x_k} and $\overline{\phi}_k$ are projected area of the kth face in the x direction and the average value of ϕ on the kth 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

$$\nabla\phi = - \underline{S}/3V \qquad (3)$$

where $\underline{S}$ is the directed area of the face opposite vertex 0. Now treat $\underline{F}$ as

piecewise linear and use equation (3) to evaluate the volume integral on the right side of equation (2). Then each tetrahedron meeting at node 0 introduces a contribution $(\overline{\underline{F}} \cdot \underline{S})/3$ where $\overline{\underline{F}}$ is the average value of $\underline{F}$ in the cell. For the cell illustrated in Figure 1, for example,

$$\overline{\underline{F}} = \frac{1}{4} (\underline{F}_0 + \underline{F}_1 + \underline{F}_2 + \underline{F}_3)$$

Summing over all cells meeting at node 0 leads to the total contribution

$$\frac{1}{3} \sum_k \overline{\underline{F}}_k \cdot \underline{S}_k$$

Since the control volume is closed, however,

$$\sum_k \underline{S}_k = 0$$

Therefore the contribution of F_0 to $\overline{F}_k$ can be discarded, leading to a sum over the faces multiplied by a constant. Thus if we write

$$\tilde{\underline{F}} = \frac{1}{3} (\underline{F}_1 + \underline{F}_2 + \underline{F}_3)$$

for the average value of $\underline{F}$ on the face opposite vertex 0 we find that the right-hand side of equation (2) can be replaced by

$$- \frac{1}{4} \sum_k \tilde{\underline{F}}_k \cdot \underline{S}_k$$

On the left hand side of equation (2) we take w to be constant inside the control volume. Since ϕ is piecewise linear, the volume average value is $\overline{\phi} = 1/4$. The factor 1/4 cancels on each side and equation (2) can therefore be written

$$\frac{d}{dt}(\sum_k V_k)\, w + \sum_k \tilde{\underline{F}}_k \cdot \underline{S}_k = 0 \qquad (4)$$

Referring to Figure 2, which illustrates a two dimensional mesh, it may be seen that with a triangular or tetrahedral mesh, each face is a common external boundary to 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 3. Vertices 4 and 5 are the centers of the two control volumes influenced by the face. It is now possible to generate the approximation (4) 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.

3. Dissipation

Equation (4) 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.

The simplest form of 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

$$D_0 = \sum_k \varepsilon^{(1)}_{k0}(w_k - w_0) \tag{5}$$

where the sum is over the nearest neighbors, as illustrated in Figure 4. The contribution $\varepsilon(1)_{k0}(w_k-w_0)$ is balanced by a corresponding contribution $\varepsilon(1)_{k0}(w_0-w_k)$ at node k, with the result that the scheme remains conservative. The coefficients $\varepsilon(1)_{k0}$ may incorporate metric information depending on local cell volumes and face areas, and can also be adapted to gradients of the solution. It is shown in reference 8 that the addition of properly controlled differences along edges can be used to assure a positivity condition on the coefficients of the semi-discrete scheme, which will prevent growth in the maximum norm and inhibit oscillations in the solution.

Formula (5) 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

$$E_0 = \sum_k (w_k - w_0) \tag{6}$$

at every mesh point, one then sets

$$D_0 = - \sum_k \varepsilon^{(2)}_{0k} (E_k - E_0) \tag{7}$$

An effective scheme is produced by blending formulas (5) and (7), and adapting $\varepsilon^{(1)}_{0k}$ 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 $\varepsilon^{(1)}_{0k}$ 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.

4. Integration to a Steady State

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

$$\frac{dw}{dt} + R(w) = 0 \tag{8}$$

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 (4), 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

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

$$w^{n+1} = w(m)$$

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

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

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

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

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

$\alpha_1 = .6$, $\alpha_2 = .6$

$\beta_{qq} = 1$, $\beta_{qr} = 0$, $q > r$

$\gamma_{q0} = 1$, $\gamma_{qr} = 0$, $r > 0$

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[4]. At the mesh point 0 the residual R_0 is replaced by $\tilde{R}_0$ where $\tilde{R}_0$ is an approximation to the solution $\bar{R}_0$ of the equation

$$\bar{R}_0 + \sum_k \varepsilon(\bar{R}_0 - \bar{R}_k) = R_0 \qquad (9)$$

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 obtain $\bar{R}$ by using two steps of the Jacobi iteration

$$\tilde{R}_0^{(m)} + \sum_k \varepsilon(\tilde{R}_0^{(m)} - \tilde{R}_k^{(m-1)}) = R_0 \qquad (10)$$

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

5. Mesh Generation

The triangulation procedure will connect a completely arbitrary collection of points to form a tetrahedral mesh. If the aircraft surface is adequately defined, we can introduce the aircraft into some pre-defined cloud of points, remove all points lying inside the aircraft structure, and then connect up the remaining points including a prescribed set of points lying on the aircraft surface. In the present version of our code, however, we have chosen to make use of existing mesh generation techniques[4,11] to create a cloud of points around the wing/body/tail/fin combination and a further cloud of points around each nacelle.

In order to describe the mapping for an isolated wing, consider the point $\underline{x} = (x,y,z)$ in physical space and denote the wing surface by the set of points $W = \{\underline{x}_w\}$. 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 $\underline{X} = (X,Y,Z)$ be the point in mapped space corresponding to $\underline{x}$ in physical space. The parabolic unwrapping is given by the transformation

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

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 $\underline{x}_0$ 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 $\underline{X}' = (X', Y', Z')$ to $\underline{X}$ by the transformation

$$\underline{X} = S_w \underline{X}'$$

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

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

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

$$\underline{X}_W = P_W \, \underline{x}_W$$

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

$$\underline{x} = P_W^{-1} \, S_W \, T \, \underline{\xi}$$

which takes a point $\underline{\xi}$ in the unit cube to the point $\underline{x}$ 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 shaped body into the symmetry plane, $z = 0$. This mapping can be constructed as a combination of a Joukowski mapping plus a shearing. We can then summarize the procedure as

$$\underline{X}_W = P_W \, B \, \underline{x}_W$$

to determine the wing geometry in mapped space followed by

$$\underline{x} = B^{-1} \, P_W^{-1} \, S_W \, T \, \underline{\xi}$$

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[11].

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.

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, and 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 is applied to each nacelle section, transforming 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.

6. Delaunay Triangulation

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

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

Here x is a point in three dimensional Euclidean space and d is the Euclidean metric. Each such region V_i is the intersection of the open half spaces 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, four such regions meet at each vertex of the Voronoi diagram. 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 four contiguous points, which have that vertex in common, by four planes to form a tetrahedron. The aggregate of tetrahedra forms the unique triangulation of the convex hull of points $\{Pi\}$ known as the Delaunay triangulation. Each Voronoi vertex is the circumcenter of the tetrahedron with which it is associated and the above construction ensures that no other point lies within the sphere that circumscribes the tetrahedron. This property ensures that the aspect ratio of the tetrahedra is reasonable and, in some sense, leads to an optimum triangulation for a given distribution of points.

The computation of the Voronoi diagaram and its associated triangulation has received considerable attention recently[12,13,14,15]. The algorithm used here is based on Boywer's method[13] and has been successfully applied in both two and three dimensions. As Boywer 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 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, creating a list of deleted vertices until all deleted vertices have been identified. From the list of deleted Voronoi vertices, we can determine the neighboring contiguous vertices in the undeleted set. Each point lying on the interface with the deleted region is joined to the new point. The deleted region is necessarily simply connected and star shaped. The new tetrahedra thus formed will exactly fill the deleted region and, moreover, will also satisfy the Delaunay criterion. It remains to label the new Voronoi neighborhoods and revise the lists that record the data structure.

The initial steps that determine the deleted vertices require a comparison of the distance between the new point and a vertex of the Voronoi diagram 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 the circumscribing sphere for 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.

In general, the list of vertices (i.e. tetrahedra) does not have any systematic ordering. However, new vertices are labelled using the addresses that corresponded to dead vertices from the deleted region. When a new point is introduced, it is therefore sensible to start searching through the most recently generated vertices since one of these will quite likely be deleted by the introduction of the new point.

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. This approach leads to a regular distribution of points at each stage. Consequently the Delaunay spheres do not become excessively large and neither does the deleted region.

It is expedient to triangulate the entire space including the interior of the aircraft as well as the exterior. It is important to identify interior tetrahedra correctly, as these tetrahedra must be removed before carrying out the flow calculation. Furthermore, it is necessary to prevent connections from exterior points breaking through the aircraft surface. Thus the interior tetrahedra must be identified at an early stage in the triangulation process.

The triangulation starts by first introducing the farfield points and then the aircraft surface points, component by component. After each component has been introduced, the tetrahedra are scanned to discover which of them belong to the most recent component. The list of tetrahedra which form the component is further examined to determine which tetrahedra have Delaunay spheres exceeding a specified threshold. When a tetrahedron has been flagged as exceeding the threshold, an extra point is introduced, the new triangulation computed and the set of component tetrahedra re-determined. This procedure can be iterated until

all Delaunay spheres have been reduced to an acceptable size. After all points on the surface of every aircraft component have been triangulated, the flowfield points are introduced.

The deleted region, associated with a new point, is now examined and if the deleted region contains tetrahedra belonging to an aircraft component, the point is rejected. Provided the threshold on the allowable size of the component Delaunay spheres is kept sufficiently small, then almost all points which genuinely lie outside the aircraft will be accepted. Of course, any points that fall inside the aircraft structure will be detected and rejected by this process.

Results

In Figure 5 we show computed pressure contours on the surface of a Boeing 747-200 including engine nacelles and pylons. The mesh contains 24685 points and is composed of 132793 tetrahedra. The complete calculation took 3924 seconds on a CRAY X-MP. One third of the time was consumed by the triangulation with the remaining time required for the flow computation of 400 cycles. The three stage time stepping scheme was used and an implicit smoothing factor $\varepsilon = 1$ was introduced to obtain a nominal Courant number of 5. The number of supersonic points was frozen after 200 cycles and the average residual was reduced, after 400 cycles, by five orders of magnitude. Although the mesh is fairly coarse, all significant flow features are evident including interference effects of the wing and tail on the body and the mutual interference of the wing, nacelle and pylon.

The present calculation requires about 8 megawords of memory. For a realistic calculation that will exhibit detailed information about the flowfield over all parts of the aircraft, we anticipate the need to increase the number of mesh points by a factor of between five and ten with a corresponding increase in the memory requirement. The availability of CRAY X-MP's with large solid state storage devices and likewise of the CRAY 2 enables one to carry out computations requiring this amount of memory. Realistic, detailed solutions of the Euler equations for complete aircraft are thus completely feasible using currently available supercomputers.

Acknowledgments

Most of our work on the airplane computer program has been carried out on computers belonging to the Cray Research Corporation. We should like to thank Cray Research for providing us with access to their computers and we are particularly grateful to Kent Misegades for his help and support. Substantial financial support for our work has been 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, O. 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. Loehner, R., Morgan, K., Peraire, J. and Zienkiewicz, O.C., "Finite Element Methods for High Speed Flows", AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, Ohio, July 1985, AIAA Paper 85-1531.

8. Jameson, A., Baker, T.J. and Weatherill, N.P., "Calculation of Inviscid Transonic Flow Over a Complete Aircraft", AIAA 24th Aerospace Sciences Meeting, Reno, Nevada, January, 1986, AIAA Paper 86-0103.

9. Augenbaum, J.M., "A Langrangian Method for the Shallow Water Equations Based on a Voronoi Mesh-One Dimensional Results", J. Comp. Physics, Vol. 53, No. 2, February 1984.

10. McCartin, B., "Discretization of the Semiconductor Device Equations", in New Problems and New Solutions for Device and Process Modeling, pp. 72-82, pub. Boole Press, 1985.

11. Baker, T.J., "Mesh Generation by a Sequence of Transformations", to appear in Applied Numerical Mathematics, December, 1986.

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

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

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

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

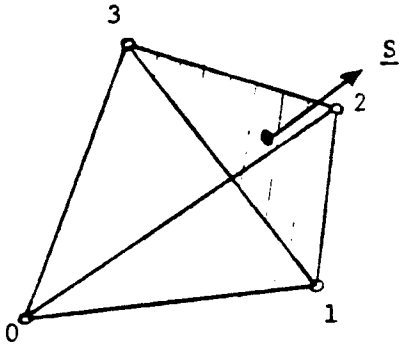

Figure 1

One tetrahedron of the control volume centered at node 0.

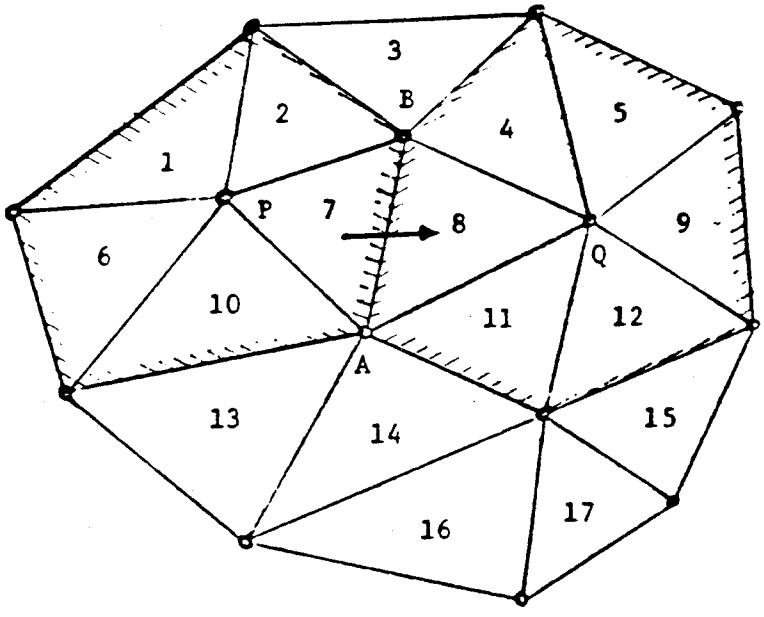

Figure 2

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.

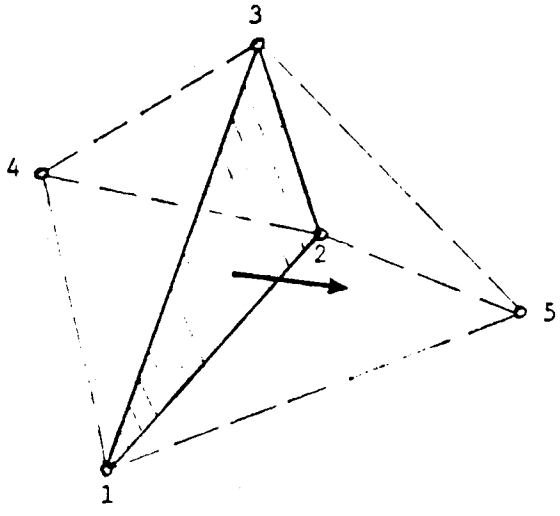

Figure 3

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.

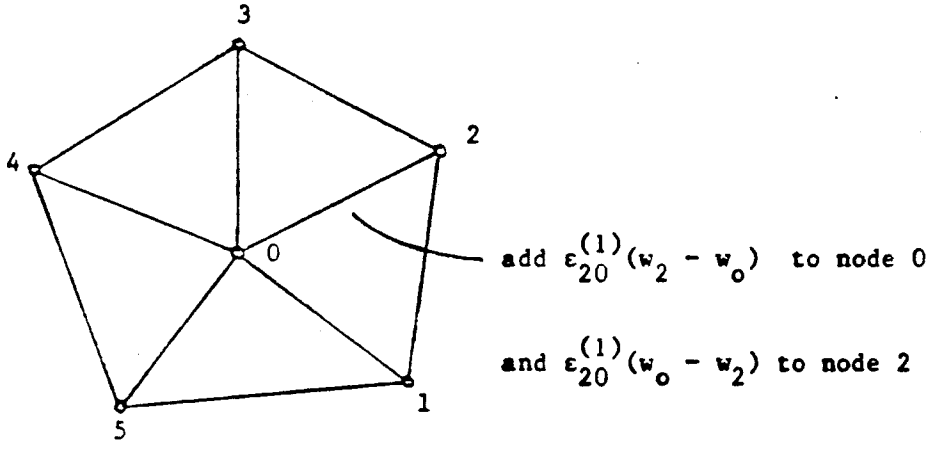

Figure 4

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

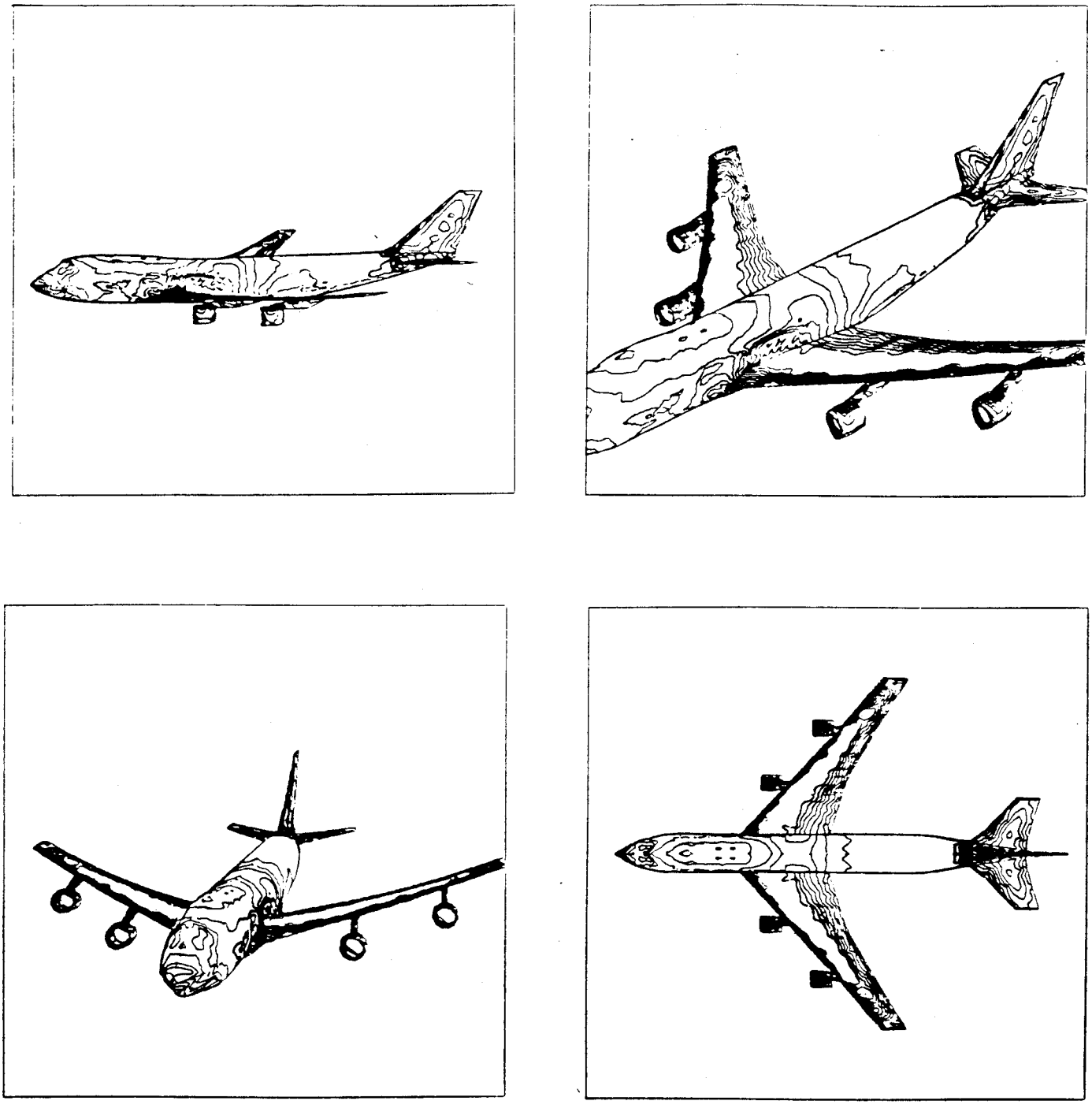

Figure 5

Surface Pressure Contours