AIAA JOURNAL
Vol. 31, No. 5, May 1993

Implicit Upwind Solution Algorithms for Three-Dimensional Unstructured Meshes

John T. Batina*
NASA Langley Research Center, Hampton, Virginia 23681

The development of implicit upwind algorithms for the solution of the three-dimensional, time-dependent Euler equations on unstructured tetrahedral meshes is described. The implicit temporal discretization involves either a two-sweep Gauss-Seidel relaxation procedure, a two-sweep point-Jacobi relaxation procedure, or a single-sweep point-implicit procedure; the upwind spatial discretization is based on the flux-difference splitting of Roe. Detailed descriptions of the three implicit solution algorithms are given, and calculations for the Boeing 747 transport configuration are presented to demonstrate the algorithms. Advantages and disadvantages of the implicit algorithms are discussed. A steady-state solution for the 747 configuration, obtained at transonic flow conditions using a mesh of over 100,000 cells, required less than 1 h of CPU time on a Cray-2 computer, thus demonstrating the speed and robustness of the general capability.

Introduction

In recent years, significant progress on developing numerical algorithms for the solution of the governing fluid flow equations on unstructured meshes has been made.[1-7] This progress includes improvements in solution accuracy as well as computational efficiency. For example, upwind methods have been developed for unstructured meshes which are based on the local wave propagation characteristics of the flow and, consequently, produce highly accurate solutions.[2,3] Most of these upwind methods, however, use explicit time-marching schemes to integrate the governing equations in time to steady state. The explicit approach is computationally efficient when applied to meshes that are coarse, but the rate of convergence deteriorates significantly when finer meshes are used. For cases where finer meshes are used, either a multigrid strategy for convergence acceleration or an implicit temporal discretization which allows large time steps is required to obtain steady-state solutions in a computationally efficient manner. Implicit upwind solution algorithms for unstructured meshes in two dimensions have been reported by the author in Ref. 8. These algorithms are similar to the point-implicit scheme of Thareja et al.,[9] although the methods of Ref. 8 are fully implicit and not point implicit.

The purpose of the paper is to describe the development of three implicit upwind algorithms for the solution of the three-dimensional time-dependent Euler equations. The implicit discretization involves either a two-sweep Gauss-Seidel relaxation procedure, a two-sweep point-Jacobi relaxation procedure, or a single-sweep point-implicit procedure. The spatial discretization of the scheme is based on the upwind approach of Roe[10] referred to as flux-difference splitting (FDS). The FDS approach is naturally dissipative and, consequently, captures shock waves and contact discontinuities sharply. Detailed descriptions of the three implicit solution algorithms are given, and calculations for the Boeing 747 transport configuration are presented to investigate the efficiency of the algorithms.

Received Nov. 18, 1991; presented as Paper 92-0447 at the AIAA 30th Aerospace Sciences Meeting, Reno, NV, Jan. 6-9, 1992; revision received Sept. 8, 1992; accepted for publication Sept. 12, 1992.

*Senior Research Scientist, Unsteady Aerodynamics Branch, Structural Dynamics Division. Associate Fellow AIAA.

Euler Equations

In the present study the flow is assumed to be governed by the three-dimensional time-dependent Euler equations which may be written in integral form as

$$\frac{\partial}{\partial t}\int_{\Omega} Q\,\mathrm{d}V + \int_{\partial\Omega} (En_x + Fn_y + Gn_z)\,\mathrm{d}S = 0 \tag{1}$$

where the vector of conserved variables Q and the convective fluxes E, F, and G are given by

$$Q = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{Bmatrix} \tag{2a}$$

$$E = \begin{Bmatrix} \rho U \\ \rho U u + p \\ \rho U v \\ \rho U w \\ (e+p)U + x_t p \end{Bmatrix} \tag{2b}$$

$$F = \begin{Bmatrix} \rho V \\ \rho V u \\ \rho V v + p \\ \rho V w \\ (e+p)V + y_t p \end{Bmatrix} \tag{2c}$$

$$G = \begin{Bmatrix} \rho W \\ \rho W u \\ \rho W v \\ \rho W w + p \\ (e+p)W + z_t p \end{Bmatrix} \tag{2d}$$

The velocities U, V, and W are defined by

$$U = u - x_t, \qquad V = v - y_t, \qquad W = w - z_t \tag{3}$$

where x_t, y_t, and z_t are the grid speeds in the x, y, and z directions, respectively, and the pressure p is given by the equation of state for a perfect gas

$$p = (\gamma - 1)\,[e - \tfrac{1}{2}\rho(u^2 + v^2 + w^2)] \tag{4}$$

The preceding equations have been nondimensionalized by the freestream density ρ_∞, the freestream speed of sound a_∞, and a reference length. Also, the second integral in Eq. (1) is a boundary integral resulting from application of the divergence theorem, and n_x, n_y, and n_z are Cartesian components of the unit normal to the boundary surface.

Spatial Discretization

The spatial discretization is based on Roe's flux-difference splitting which is herein implemented as a cell-centered scheme whereby the flow variables are stored at the centroid of each tetrahedron, and the control volume is simply the tetrahedron itself. Consequently, the spatial discretization involves a flux balance where the fluxes across the four faces of a given tetrahedron are summed as

$$\sum_{m=1}^{4} H\Delta S = \sum_{m=1}^{4} (En_x + Fn_y + Gn_z)\Delta S \tag{5}$$

where ΔS is the area of the face. The flux vector H is approximated by

$$H = \tfrac{1}{2}[H(Q^+) + H(Q^-) - |\tilde{A}|(Q^+ - Q^-)] \tag{6}$$

where Q^- and Q^+ are the state variables to the left and right of the cell face and A is the flux Jacobian matrix given by $\partial H/\partial Q$. Also, the tilde and the absolute value sign indicate that the flux Jacobian is evaluated using the so-called Roe-averaged flow variables and the absolute value of the characteristic speeds.

The left and right states Q^- and Q^+ are determined by upwind-biased interpolations of the primitive variables q. In three dimensions, for a given tetrahedron j, for example, the upwind-biased interpolation for q^- across the common face between tetrahedra j and k is defined by

$$q^- = q_j + \tfrac{1}{6}[(1 - \kappa)\Delta_- + (1 + \kappa)\Delta_+] \tag{7}$$

where

$$\Delta_+ = \tfrac{3}{2}(q_k - q_j) \tag{8a}$$

$$\Delta_- = q_j - q_i \tag{8b}$$

In Eqs. (7) and (8), q_j and q_k are the vectors of primitive variables at centroids j and k, respectively; and q_i, the vector of primitive variables at node i (the node of tetrahedron j opposite to the face being considered), is determined by an inverse-distance-weighted average of the flow variables in the tetrahedra surrounding node i. The upwind-biased interpolation for q^+ is determined similarly. Also, the parameter κ in Eq. (7) controls a family of difference schemes by appropriately weighting Δ_- and Δ_+. On structured meshes it is easy to show that $\kappa = -1$ yields a fully upwind scheme, $\kappa = 0$ yields Fromm's scheme, and $\kappa = 1$ yields central differencing.

On highly stretched meshes, the formula for Δ_+ is modified to be

$$\Delta_+ = \frac{3a}{a+b}(q_k - q_j) \tag{9}$$

where a and b are the distances from the midpoint of the face to the centroids of tetrahedra j and k, respectively. This formula weights the flow variables in the interpolation formula [Eq. (7)] differently to account for the stretching of the mesh. For example, by substituting Eq. (9) into Eq. (7) and letting $\kappa = 1$

$$q^- = \frac{b}{a+b} q_j + \frac{a}{a+b} q_k \tag{10}$$

Furthermore, in calculations involving upwind-biased schemes, oscillations in the solution near shock waves are expected to occur. To eliminate these oscillations flux limiting is usually required. The flux limiter modifies the upwind-biased interpolations for q^- and q^+ such that, for example,

$$q^- = q_j + \tfrac{1}{6}[(1 - \kappa s)\Delta_- + (1 + \kappa s)\Delta_+] \tag{11}$$

where s is the flux limiter. In the present study, a continuously differentiable flux limiter was employed which is defined by

$$s = \frac{2\Delta_-\Delta_+ + \epsilon}{\Delta_-^2 + \Delta_+^2 + \epsilon} \tag{12}$$

where ϵ is a very small number used to prevent division by zero in smooth regions of the flow.

Temporal Discretizations

The temporal discretizations are of the implicit type and are generally derived by first linearizing the flux vector H according to

$$H^{n+1} = H^n + \frac{\partial H}{\partial Q}\Delta Q \tag{13}$$

where $\partial H/\partial Q$ is the flux Jacobian A, as discussed before, and $\Delta Q = Q^{n+1} - Q^n$. Linearizing both flux terms on the right-hand side of Eq. (6) and using Eq. (13), and ignoring the tilde on the flux Jacobian, results in

$$\left[\frac{\text{vol}}{\Delta t} I + \sum_{m=1}^{4} A^+(Q_j)\Delta S\right]\Delta Q_j + \sum_{m=1}^{4} A^-(Q_m)\Delta S\Delta Q_m = -\frac{1}{2}\sum_{m=1}^{4}[H(Q^+) + H(Q^-) - |\tilde{A}|(Q^+ - Q^-)]^n\,\Delta S \tag{14}$$

where I is the identity matrix, vol is the volume of the tetrahedron j, and ΔQ_m is the change in flow variables in each of the four tetrahedra adjacent to tetrahedron j. Also, in Eq. (14) A^+ and A^- are forward and backward flux Jacobians, respectively. For flux-difference splitting, the exact Jacobian A [derivative of the right-hand side of Eq. (6) with respect to Q] is too expensive to compute, and thus an approximate Jacobian is normally used. This is accomplished by constructing the Jacobians making use of the fact that the forward and backward Jacobians should have nonnegative and nonpositive eigenvalues (characteristic speeds), respectively. This is accomplished by expressing alternatively the Jacobians using similarity transformations such that

$$A^+ = R\Lambda^+R^{-1} \qquad A^- = R\Lambda^-R^{-1} \tag{15}$$

where Λ^+ and Λ^- are diagonal matrices whose diagonal elements are the eigenvalues

$$\lambda^+ = \tfrac{1}{2}(\lambda + |\lambda|) \qquad \lambda^- = \tfrac{1}{2}(\lambda - |\lambda|) \tag{16}$$

and R is the matrix whose columns are the corresponding eigenvectors.

A similar implicit temporal discretization that is more efficient than the discretization of Eq. (14) is derived by linearizing the flux vector of the quasilinear form of the Euler equations with respect to the primitive flow variables. This

approach results in an equation similar to that of Eq. (14) given by

$$\left[\frac{\text{vol}}{\Delta t} I + \sum_{m=1}^{4} a^+(q_j)\Delta S\right] \Delta q_j + \sum_{m=1}^{4} a^-(q_m)\Delta S \Delta q_m = -\frac{B^{-1}}{2} \sum_{m=1}^{4} [H(Q^+) + H(Q^-) - |\tilde{A}|(Q^+ - Q^-)]^n \Delta S \tag{17}$$

where the matrix B relates the time derivative of Q to the time derivative of q as simply

$$\frac{\partial Q}{\partial t} = B\frac{\partial q}{\partial t} \tag{18}$$

The discretization represented by Eq. (17) is more efficient than the discretization represented by Eq. (14) because the flux Jacobians a^+ and a^- are simpler mathematically and, therefore, faster to compute than the flux Jacobians A^+ and A^-.

Gauss-Seidel Relaxation Procedure

Direct solution of the system of simultaneous equations, which results from application of Eq. (17) for all tetrahedra in the mesh, requires the inversion of a large matrix with large bandwidth which is computationally expensive. Instead, a Gauss-Seidel (GS) relaxation approach is used to solve the equations whereby the summation involving Δq_m is moved to the right-hand side of Eq. (17). The terms in this summation are then evaluated for a given time step using the most recently computed values for Δq_m. The solution procedure then involves only the inversion of a 5×5 matrix [represented by the terms in brackets on the left hand side of Eq. (17)] for each tetrahedron in the mesh. Also, although the procedure is implemented for application on (randomly ordered) unstructured meshes, it is not a point Gauss-Seidel procedure. The method is in fact more like line Gauss-Seidel since the list of tetrahedra that make up the unstructured mesh is reordered from upstream to downstream, and the solution is obtained by sweeping two times through the mesh as dictated by stability considerations. The first sweep is performed in the direction from upstream to downstream, and the second sweep is from downstream to upstream. For purely supersonic flows, the second sweep is unnecessary.

Point-Jacobi Relaxation Procedure

The innermost do-loop of the Gauss-Seidel relaxation procedure does not vectorize due to vector recurrences resulting from the evaluation of Δq_m using the most recently computed values. Hence, a two-sweep point-Jacobi (PJ) type of relaxation procedure has been developed that fully vectorizes. It is consequently faster per iteration than the GS procedure but is expected to have diminished stability properties since the first sweep ignores the Δq_m term altogether, and the second sweep uses the values of Δq from the first sweep to evaluate the Δq_m term in the second sweep. Thus the PJ procedure is represented by the following.

First sweep:

$$\left[\frac{\text{vol}}{\Delta t} I + \sum_{m=1}^{4} a^+(q_j)\Delta S\right] \Delta q_j^{(1)} = -\frac{B^{-1}}{2} \sum_{m=1}^{4} [H(Q^+) + H(Q^-) - |\tilde{A}|(Q^+ - Q^-)]^n \Delta S \tag{19a}$$

Second sweep:

$$\left[\frac{\text{vol}}{\Delta t} I + \sum_{m=1}^{4} a^+(q_j)\Delta S\right] \Delta q_j = -\sum_{m=1}^{4} a^-(q_m)\Delta S \Delta q_m^{(1)} - \frac{B^{-1}}{2} \sum_{m=1}^{4} [H(Q^+) + H(Q^-) - |\tilde{A}|(Q^+ - Q^-)]^n \Delta S \tag{19b}$$

Point-Implicit Procedure

Advantages of the Gauss-Seidel and point-Jacobi relaxation procedures are that they are numerically stable for reasonably large CFL numbers even on very fine meshes, and consequently they enable rapid convergence to steady state. For unsteady applications, they allow the selection of the step size based on the physical problem rather than on numerical stability considerations. This is in contrast with an explicit time integration which has a restrictive step size for unsteady applications which is more severe on finer meshes. A disadvantage of the GS and PJ relaxation procedures, though, is that they require about twice the memory of an explicit time integration, primarily because of having to store the backward flux Jacobian a^-. Hence, a single-sweep point-implicit (PI) procedure was developed [represented by Eq. (19a)] that does not require the calculation of the backward flux Jacobian. It is consequently faster than the GS or PJ relaxation procedures but is expected to have diminished stability properties since the Δq_m term is totally ignored.

Boundary Conditions

To impose the flow tangency boundary conditions along the surface of the vehicle, the flow variables are set within dummy cells that are effectively inside the geometry being considered. The velocity components within a dummy cell, $(u, v, w)_d$, are determined from the values in the cell j adjacent to the surface, $(u, v, w)_j$. This is accomplished by first rotating the components into a coordinate system that has a coordinate direction normal to the boundary face. The sign of the velocity component in this direction is changed (hence, imposing no flow through the face) and the three velocity components are then rotated back into the original x, y, z coordinate system. After considerable algebra this yields

$$\begin{Bmatrix} u \\ v \\ w \end{Bmatrix}_d = \begin{bmatrix} 1-2n_x^2 & -2n_xn_y & -2n_xn_z \\ -2n_xn_y & 1-2n_y^2 & -2n_yn_z \\ -2n_xn_z & -2n_yn_z & 1-2n_z^2 \end{bmatrix} \begin{Bmatrix} u \\ v \\ w \end{Bmatrix}_j \tag{20}$$

where n_x, n_y, and n_z are the x, y, and z components of the unit vector that is normal to the boundary face. Also, pressure and density within the dummy cell are set equal to the values in the cell adjacent to the surface.

After application of the upwind-biased interpolation formula to determine q^- and q^+ at each face, the velocity components are corrected to give a stronger implementation of the surface boundary condition according to

$$\begin{aligned} u_{\text{corrected}} &= u - n_x(un_x + vn_y + wn_z) \\ v_{\text{corrected}} &= v - n_y(un_x + vn_y + wn_z) \\ w_{\text{corrected}} &= w - n_z(un_x + vn_y + wn_z) \end{aligned} \tag{21}$$

This correction effectively allows no flow through the boundary faces.

In the far field a characteristic analysis based on Riemann invariants is used to determine the values of the flow variables on the outer boundary of the grid.[2] This analysis correctly accounts for wave propagation in the far field which is important for rapid convergence to steady state and serves as a "nonreflecting" boundary condition for unsteady applications.

Results and Discussion

To assess the efficiency of the implicit upwind solution algorithms, calculations were performed for the Boeing 747 aircraft configuration. The results were obtained using the unstructured mesh shown in Fig. 1. The 747 geometry includes the fuselage, the wing, horizontal and vertical tails, underwing pylons, and flow-through engine nacelles. The unstructured

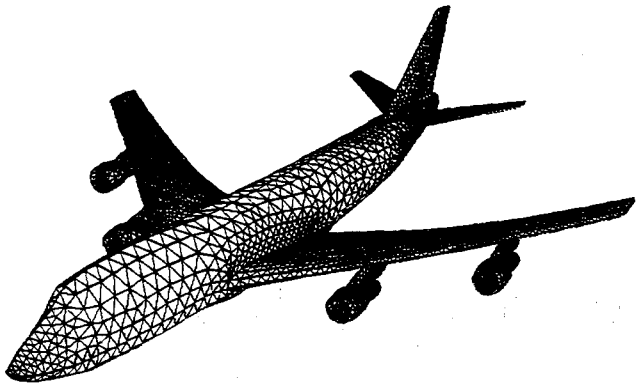


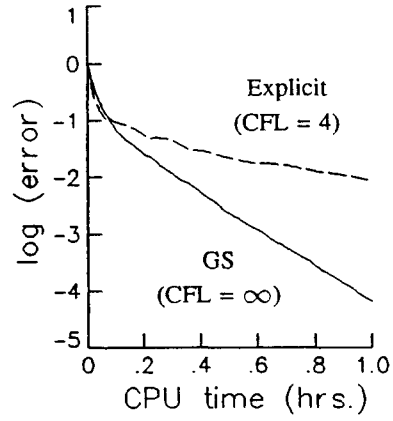
Fig. 1 Surface mesh of triangles for the Boeing 747 aircraft.

mesh for the 747 contains 101,475 tetrahedra and 19,055 nodes for the half-span airplane. Also, there are 4159 nodes and 8330 triangles on the boundaries of the mesh that include the airplane, the symmetry plane, and the far field. Steady-state calculations were performed for the aircraft at a freestream Mach number M_∞ of 0.84 and an angle of attack α of 2.73 deg. All of the results were obtained on the Cray-2 computer (Navier) at the Numerical Aerodynamic Simulation Facility located at NASA Ames Research Center.

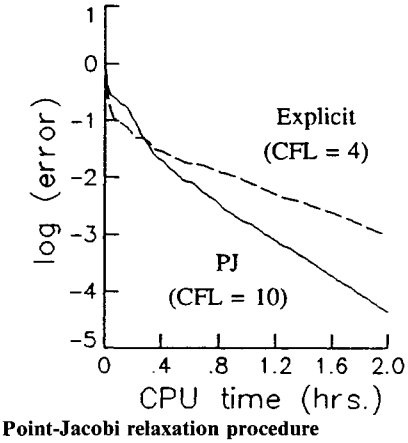
Steady-state calculations were performed first using the implicit Gauss-Seidel relaxation procedure. These implicit results were obtained using a CFL number of infinity and the flux Jacobians were updated only every 20 iterations. Such a large value of the CFL number was used for the GS results since the relaxation scheme has maximum damping and hence fastest convergence for very large time steps. This is in contrast with implicit approximate factorization schemes which have maximum damping for CFL numbers on the order of 10. (Computationally, when a CFL number of infinity is used, the term involving $\text{vol}/\Delta t$ is simply ignored.) Results also were obtained for comparison purposes using an explicit, three-stage, Runge-Kutta time-marching scheme.[2] These explicit results were obtained using a CFL number of 4.0, with residual smoothing and local time stepping to accelerate convergence to steady state.

A comparison of the convergence histories between explicit and implicit GS solutions is shown in Fig. 2a. The "error" in the solution was taken to be the L_2-norm of the density residual. As shown in Fig. 2a, the GS solution converges faster than the explicit solution. The GS solution, for example, required 622 iterations or approximately 3420 CPU s (less than 1h) to converge to engineering accuracy, which is taken to be a four order-of-magnitude reduction in the solution error. In contrast, the explicit solution required 1552 iterations or approximately 11,560 CPU s to achieve the same convergence. The GS relaxation procedure is not only faster on a per iteration basis but provides faster convergence to steady state in terms of CPU time.

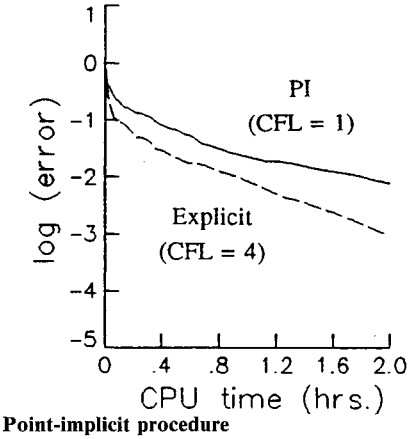
Calculations were performed next with the implicit point-Jacobi relaxation procedure for the 747 aircraft. These calculations were unstable for CFL numbers of infinity and 100 because of updating the Δq_m term in a point-Jacobi fashion rather than in a Gauss-Seidel fashion. Stable results were obtained using a CFL number of 10 and local time stepping was used to accelerate convergence to steady state. As shown in Fig. 2b, the PJ solution converges faster than the explicit solution, but not as fast as the GS solution convergence of Fig. 2a. The PJ result required 1942 iterations or approximately 5080 CPU s to converge the solution four orders of magnitude. Hence, the PJ relaxation procedure, which is faster than either the explicit or GS procedures on a per iteration basis, provides faster convergence to steady state in terms of CPU time in comparison with the explicit Runge-Kutta procedure.



a) Gauss-Seidel relaxation procedure



b) Point-Jacobi relaxation procedure



c) Point-implicit procedure

Fig. 2 Comparison of implicit and explicit convergence histories for the Boeing 747 aircraft at $M_\infty = 0.84$ and $\alpha = 2.73$ deg.

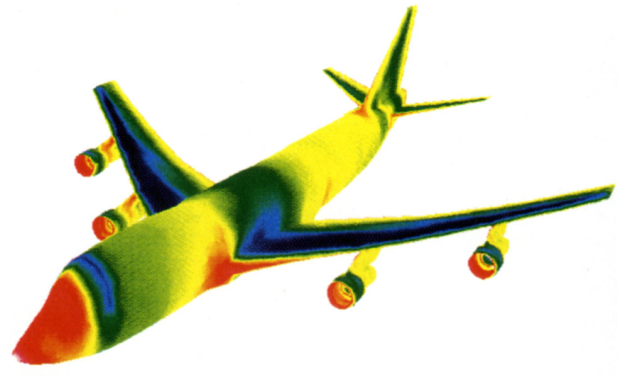


Fig. 3 Steady pressure coefficient contours on the Boeing 747 aircraft at $M_\infty = 0.84$ and $\alpha = 2.73$ deg.

Calculations were performed also with the point-implicit procedure for the 747 aircraft. These calculations were unstable for CFL numbers of infinity, 100, and 10 due to the approximations that are made to solve the implicit equations. Stable results were obtained using a CFL number of unity and local time stepping was used to accelerate convergence to steady state. As shown in Fig. 2c, the PI solution converges slower than the explicit solution and is also slower than the other two implicit procedures. Although the PI algorithm converges the slowest of all of the methods used in the present study, it is the fastest algorithm on a per iteration basis. The PI algorithm is about twice as fast as the GS relaxation procedure and is nearly three times as fast as the explicit Runge-Kutta method. Efforts to improve the rate of convergence of the PI procedure by slowly increasing the CFL number over the course of the calculation were unsuccessful.

Finally, steady pressure coefficient contours on the surface of the 747 aircraft are shown in Fig. 3. These results were obtained using the GS relaxation procedure with a CFL number of infinity. The contours indicate that there is a significant amount of flow compression on the nose of the aircraft, along the inboard leading edge of the wing, and inside the cowl of the engine nacelles. There is flow expansion on the forward fuselage, on the horizontal and vertical tail surfaces, and on the upper surface of the wing terminated by a shock wave.

Concluding Remarks

The development of implicit upwind algorithms for the solution of the three-dimensional time-dependent Euler equations on unstructured tetrahedral meshes was described. The implicit temporal discretization involves either a two-sweep Gauss-Seidel relaxation procedure, a two-sweep point-Jacobi relaxation procedure, or a single-sweep point-implicit procedure; whereas the upwind spatial discretization is based on the flux-difference splitting (FDS) of Roe. The FDS discretization is naturally dissipative and, consequently, captures shock waves and contact discontinuities sharply. Detailed descriptions of the three implicit solution algorithms were given, and calculations for the Boeing 747 transport configuration were presented to demonstrate the algorithms. The 747 geometry included the fuselage, wing, horizontal and vertical tails, underwing pylons, and flow-through engine nacelles. Advantages and disadvantages of the implicit algorithms were discussed. A steady-state solution for the 747 configuration, obtained at transonic flow conditions using a mesh of over 100,000 cells, required less than 1h of CPU time on a Cray-2 computer, thus demonstrating the speed and robustness of the general capability.

Acknowledgment

The author acknowledges Neal Frink of the Transonic Aerodynamics Branch, NASA Langley Research Center, Hampton, Virginia, for providing the unstructured tetrahedral mesh for the Boeing 747 aircraft.

References

[1]Salas, M. D. (ed.), "Accuracy of Unstructured Grid Techniques Workshop," NASA Langley Research Center, Hampton, VA, Jan. 1990.

[2]Batina, J. T., "Three-Dimensional Flux-Split Euler Schemes Involving Unstructured Dynamic Meshes," AIAA Paper 90-1649, June 1990.

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

[4]Lohner, R., and Baum, J. D., "Numerical Simulation of Shock Interaction with Complex Geometry Three-Dimensional Structures Using a New Adaptive H-Refinement Scheme on Unstructured Grids," AIAA Paper 90-0700, Jan. 1990.

[5]Thareja, R. R., Morgan, K., Peraire, J., and Peiro, J., "A Three-Dimensional Upwind Finite Element Point Implicit Unstructured Grid Euler Solver," AIAA Paper 89-0658, Jan. 1989.

[6]Mavriplis, D. J., "Algebraic Turbulence Modeling for Unstructured and Adaptive Meshes," AIAA Paper 90-1653, June 1990.

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

[8]Batina, J. T., "Implicit Flux Split Euler Schemes for Unsteady Aerodynamic Analysis Involving Unstructured Dynamic Meshes," AIAA Paper 90-0936, April 1990.

[9]Thareja, R. R., Stewart, J. R., Hassan, O., Morgan, K., and Peraire, J., "A Point Implicit Unstructured Grid Solver for the Euler and Navier-Stokes Equations," AIAA Paper 88-0036, Jan. 1988.

[10]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372.