NASA Technical Memorandum 85957

NASA-TM-85957 19840021487

An Efficient Approximate Factorization Implicit Scheme for the Equations of Gasdynamics

Timothy J. Barth and Joseph L. Steger

June 1984

National Aeronautics and
Space Administration

NASA Technical Memorandum 85957

An Efficient Approximate Factorization Implicit Scheme for the Equations of Gasdynamics

Timothy J. Barth
Joseph L. Steger, Ames Research Center, Moffett Field, California

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

N84-29556 #

An Efficient Approximate Factorization Implicit Scheme for the Equations of Gasdynamics

TIMOTHY J. BARTH* AND JOSEPH L. STEGER†

NASA Ames Research Center, Moffett Field, CA 94035

Summary

An efficient implicit finite-difference algorithm for the gasdynamic equations utilizing matrix reduction techniques is presented. A significant reduction in arithmetic operations is achieved while maintaining the same favorable stability characteristics and generality found in the Beam and Warming approximate factorization algorithm. Steady-state solutions to the conservative Euler equations in generalized coordinates are obtained for transonic flows about a NACA 0012 airfoil. The theoretical extension of the matrix reduction technique to the full Navier-Stokes equations in Cartesian coordinates is presented in detail. Linear stability, using a Fourier stability analysis, is demonstrated and discussed for the one-dimensional Euler equations. It is shown that the method offers advantages over the conventional Beam and Warming scheme and can retrofit existing Beam and Warming codes with minimal effort.

I. Introduction

Numerical algorithms for solving the compressible Euler and Navier-Stokes equations have received considerable attention over the last decade. Both implicit and explicit time advance and iterative techniques have been utilized. Explicit methods typically have lower arithmetic operation counts but more restrictive stability bounds than implicit methods. Use of implicit methods is generally advantageous for flow situations in which an explicit time step limitation would be much less than the time step necessary for the desired accuracy. This situation frequently arises in the solution of unsteady flows

*Informatics General Corporation.
†Senior Staff Scientist

that are characterized by low frequency motion, boundary condition forcing, and high Reynolds number viscous effects. Implicit algorithms can often be readily adapted to be efficient relaxation schemes for steady state applications as well.

The purpose of this paper is to present a technique that substantially reduces the arithmetic operation count, but otherwise retains the stability characteristics of a Beam-Warming approximate factorization implicit algorithm for the Euler equations. This new method extends a matrix splitting of Steger (ref. 1), previously restricted to Cartesian coordinates, such that equation decoupling is achieved for the conservative form of the Euler equations. The equation decoupling (or matrix reducibility) results in a significant improvement in the efficiency of the algorithm. By utilizing a local transformation, the equation decoupling techniques are extended to the Euler equations in generalized coordinates. Computation of transonic flow about a NACA 0012 airfoil is used to verify that no numerical stability is lost with the new reducible implicit algorithm. Extension of the technique to the Navier-Stokes equations is also indicated in Appendix A.

This work was partially funded by the Air Force Office of Scientific Research under AFOSR-82-0254. The discussions and support of Drs. Joseph Shang and Wilbur Hankey of the Air Force Flight Dynamics Laboratory are also gratefully acknowleged.

II. Background

In this section we review a sound speed and velocity splitting concept for an approximate factorization implicit algorithm in Cartesian coordinates and describe how it can be used to improve computational efficiency. The extension for general coordinate systems is described in the following section.

A. Beam and Warming Algorithm

The conservative form of the Euler equations for a perfect gas can be written in Cartesian coordinates as

$$\partial_t Q + \partial_x E + \partial_y F = 0 \tag{2.1}$$

where

$$Q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \quad E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho v u \\ u(e+p) \end{bmatrix}, \quad F = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ v(e+p) \end{bmatrix} \tag{2.2}$$

and

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho(u^2 + v^2)\right) \tag{2.3}$$

Here ρ is the fluid density, u and v are the Cartesian velocity components, e is the total energy per volume, and p is the pressure.

A general-purpose implicit finite difference scheme (ref. 2 and 3) for equation (2.1) that can be readily adapted to either steady or unsteady flow applications is given by

$$\begin{aligned}\left[I + h\delta_x A^n + D_x^{(2)}\right]\left[I + h\delta_y B^n + D_y^{(2)}\right]\Delta Q^n = \\ -\Delta t\left[\delta_x E^n + \delta_y F^n + D^{(4)} Q^n\right]\end{aligned} \tag{2.4}$$

where

$$D^{(4)} = \epsilon_e \Delta t[(\nabla\Delta)_x^2 + (\nabla\Delta)_y^2] \tag{2.5a}$$

and

$$D_x^{(2)} = -\epsilon_i \Delta t(\nabla\Delta)_x, \quad D_y^{(2)} = -\epsilon_i \Delta t(\nabla\Delta)_y \tag{2.5b}$$

Here δ_x and δ_y are three-point, second-order accurate, central space difference operators, while Δ and ∇ denote forward and backward space differences. Either first- or second-order time accurate differencing may be used with h=Δt or $\Delta t/2$; alternately, the time step may be used as a relaxation parameter and may vary in space as well. With the use of central space differencing numerical dissipation is needed and implemented with the second and fourth order differences given above. The parameters ϵ_e and ϵ_i control the amount of numerical dissipation. These parameters may be constants or may vary with the solution gradients (in which case they are moved inside the operators so as to maintain conservative form). Steady state convergence can be greatly enhanced by more implicit treatment of the dissipation terms and by use of space-varying scaling parameters.

In equation (2.4), local time linearizations have been utilized to avoid solving nonlinear equations between time or iterative levels n to n+1 by expanding E and F in terms of ΔQ as

$$E^{n+1} = E^n + A^n(Q^{n+1} - Q^n), \quad A = \partial E/\partial Q \tag{2.6a}$$

$$F^{n+1} = F^n + B^n(Q^{n+1} - Q^n), \quad B = \partial F/\partial Q \tag{2.6b}$$

The left hand side operators of equation (2.4) form a linear system of equations that must be solved at each time or iteration level. Although the matrix band structure of this system has been altered by approximately factoring into one-dimensional-like operators, the inversion process has been simplified. The approximate factorization (AF) reduces the inversion work but sometimes at the cost of limiting the time step Δt because of either time accuracy considerations or because of reduced inefficiency in relaxation applications.

In practice the difference equations (2.4) are solved in the ADI algorithm form

$$\mathcal{L}_x \Delta Q^* = RHS(2.4) \tag{2.7a}$$

$$\mathcal{L}_y \Delta Q^n = \Delta Q^* \tag{2.7b}$$

Variants of this algorithm include higher-order difference approximations (including pseudo-spectral right-hand-side operators (ref. 4)), space varying Δt for steady state applications, addition of viscous terms, etc. [ref. 5-9]

B. Diagonalization

Efforts have been made to reduce the inversion work over and above what is obtained with approximate factorization. In particular, Pulliam and Chaussee (ref. 7) have used eigenvector-similarity transforms to reduce the block tridiagonal matrices to scalar tridiagonals (see also ref. 8-10). Specifically, the eigenvectors of the A and B Jacobian matrices, denoted as X and Y, are further approximately factored into the left-hand-side of equation (2.4) as

$$\begin{aligned} X\Big[I + h\delta_x \Lambda_A{}^n + D_x^{(2)}\Big]X^{-1}Y\Big[I + h\delta_y \Lambda_B{}^n + D_y^{(2)}\Big]Y^{-1}\Delta Q^n = \\ -\Delta t\Big[\delta_x E^n + \delta_y F^n + D^{(4)}Q^n\Big] \end{aligned} \tag{2.8}$$

where

$$\Lambda_A = X^{-1}AX \quad \text{and} \quad \Lambda_B = Y^{-1}BY$$

As fig. 1 illustrates, the number of operations in solving a block tridiagonal (that uses only L-U decomposition) increases with the cube of the block size, a formula given in ref. 11 indicates the work increases as $\frac{14}{3}m^3 + \frac{9}{2}m^2 - \frac{1}{6}m$, where m is the dimension of the block. Specifically our own coded 4×4 block solver requires 370 operations per entry (i.e., grid point) while the 1×1 tridiagonal needed in equation (2.8) requires only 9 operations per entry (or 36 operations for 4 scalar tridiagonals). Although the eigenvector matrices must be formed and multiplied through (each 4×4 multiply requires 28 operations per entry), the arithmetic operation count of equation (2.8) is considerably reduced from that of equation (2.4). For block tridiagonal matrices subject to periodicity conditions and for block pentadiagonal matrices, the saving is even more significant.

C. Sound Speed and Velocity Splitting

An alternate method for reducing the inversion (i.e., solution) work was proposed and demonstrated in ref. 1. This method was also motivated by similarity transforms, but it does not explicitly use them in the algorithm. Instead, it involves splitting the Jacobian matrices A and B into velocity and sound speed (or pressure parts) as

$$A = A_u + A_c \tag{2.9a}$$

$$B = B_v + B_c \tag{2.9b}$$

such that the eigenvalues $\sigma(A)$ and $\sigma(B)$ are

$$\sigma(A) = \sigma(A_u) + \sigma(A_c) \tag{2.10a}$$

$$\sigma(B) = \sigma(B_u) + \sigma(B_c) \tag{2.10b}$$

with

$$\sigma(A_u) = (u, u, u, u), \quad \sigma(A_c) = (0, c, 0, -c) \tag{2.11a}$$

$$\sigma(B_v) = (v, v, v, v), \quad \sigma(B_c) = (0, 0, c, -c) \tag{2.11b}$$

The matrices A_u, A_c, B_v, and B_c were found from a natural reduced form of the equations in nonconservative form. Specifically A_c and B_c were split, as

$$A_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 & 0 \\ (u^2+v^2)/2 & -u & -v & 1 \\ 0 & 0 & 0 & 0 \\ a_{41}^c & a_{42}^c & -uv & u \end{bmatrix}, \quad B_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ (u^2+v^2)/2 & -u & -v & 1 \\ b_{41}^c & -uv & b_{43}^c & v \end{bmatrix} \tag{2.12}$$

where

$$a_{41}^c = [u(u^2+v^2)/2] - \gamma up/[\rho(\gamma-1)^2]$$

$$a_{42}^c = \gamma p/[\rho(\gamma-1)^2] - u^2$$

$$b^c_{41} = [v(u^2 + v^2)/2] - \gamma vp/[\rho(\gamma - 1)^2]$$

$$b^c_{43} = \gamma p/[\rho(\gamma - 1)^2] - v^2$$

while A_u and B_v are

$$A_u = A - A_c \tag{2.13a}$$

$$B_v = B - B_c \tag{2.13b}$$

This splitting produces matrices A_u and B_v that are more complex than A and B. However, Steger (ref. 1) noted that Q is an eigenvector of both matrices, i.e.,

$$A_u Q = uQ \tag{2.14a}$$

$$B_v Q = vQ \tag{2.14b}$$

which prompted the ad hoc substitution

$$AQ = (uI + A_c)Q \tag{2.15a}$$

$$BQ = (vI + B_c)Q \tag{2.15b}$$

The matrices $uI + A_c$ and $vI + B_c$ have a reduced form that simplifies inversion compared to A and B.

Insertion of equation (2.15) into the equations for local linearization of the Jacobians, equation (2.6), produces

$$E^{n+1} = E^n + (uI + A_c)^n(Q^{n+1} - Q^n) \tag{2.16a}$$

$$F^{n+1} = F^n + (vI + B_c)^n(Q^{n+1} - Q^n) \tag{2.16b}$$

Utilizing these linearizations in the basic algorithm equation (2.4)

$$L_x L_y \Delta Q = RHS(2.4) \tag{2.17}$$

gives new left-hand-side operators L_x and L_y that are easier to solve

$$L_x = \left[I + h\delta_x(uI + A_c)^n + D_x^{(2)}\right] \tag{2.18a}$$

$$L_y = \left[I + h\delta_y(vI + B_c)^n + D_y^{(2)}\right] \tag{2.18b}$$

The end result of this splitting is that the new operators L_x and L_y form matrices that no longer require 4×4 block tridiagonal inversions. Leaving out the dissipation terms to illustrate the structure of these operators, we obtain

$$L_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \Delta t \delta_x \begin{bmatrix} u & 0 & 0 & 0 \\ a_{21}^c & u + a_{22}^c & a_{23}^c & a_{24}^c \\ 0 & 0 & u & 0 \\ a_{41}^c & a_{42}^c & a_{43}^c & u + a_{44}^c \end{bmatrix} \tag{2.19a}$$

$$L_y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \Delta t \delta_y \begin{bmatrix} v & 0 & 0 & 0 \\ 0 & v & 0 & 0 \\ b_{31}^c & b_{32}^c & v + b_{33}^c & b_{34}^c \\ b_{41}^c & b_{42}^c & b_{43}^c & v + b_{44}^c \end{bmatrix} \tag{2.19b}$$

where a^c and b^c are the respective elements of A_c and B_c given by equation (2.12).

For the L_x operator, for example, the first and third rows decouple from the system and can be solved as scalar tridiagonal matrices with their respective right-hand-sides. Once these rows are solved, the elements of the first and third columns can be moved to the right-hand-side. The second and fourth equations remain coupled and are solved as a 2×2 block tridiagonal matrix. The dissipation terms also form diagonal tridiagonals and so do not alter the structure. The block decoupling of the L_y operator is even more conspicuous and is inverted (i.e., solved for) in a similar manner.

Substitution of the left-hand-side operators given by equation (2.18) in place of those of equation (2.7) results in a substantial reduction in arithmetic operations. Our typical 2×2 block tridiagonal requires 55 operations per point, so the overall inversion, including the two scalar tridiagonals, requires 73 operations per entry. Because the two scalar tridiagonals have identical coefficients, this work can be even further cut by solving them together.

The matrix splitting produces the flux vectors

$$E = AQ = uIQ + A_cQ = E_u + E_c \tag{2.20a}$$

$$F = BQ = vIQ + B_cQ = F_v + F_c \tag{2.20b}$$

where

$$E_u = \begin{bmatrix} \rho u \\ \rho u^2 \\ \rho v u \\ u e \end{bmatrix}, \quad E_c = \begin{bmatrix} 0 \\ p \\ 0 \\ up \end{bmatrix}, \quad F_v = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 \\ v e \end{bmatrix}, \quad F_c = \begin{bmatrix} 0 \\ 0 \\ p \\ vp \end{bmatrix} \tag{2.21}$$

Note that we do not reproduce the matrix splitting in equation (2.9) (with A_c and B_c defined from equation (2.12)) by taking the Jacobians of equations (2.21). Surprisingly, linear stability analysis (Appendix B) as well as numerical experiment have shown that the use of the Jacobian matrices for A_c and B_c is unsatisfactory. However, the same stability analysis and numerical experimentation have confirmed that no numerical stability degradation occurs by using the substitute linearization matrices, equation (2.15). But the linearization in equation (2.16) is only first-order accurate because of the substitution of equation (2.14).

The matrix splitting concept can be extended to three dimensions. In this case the difference equations can be represented by

$$L_x L_y L_z (Q^{n+1} - Q^n) = -\Delta t[\delta_x E^n + \delta_y F^n + \delta_z G^n + D^{(4)} Q^n] \tag{2.21}$$

Associated with each operator are 5×5 block tridiagonal matrices, with each matrix inversion requiring about 700 operations per grid point. Typically the inversion represents 70 to 80 per cent of the overall work per point. Replacing each Jacobian A by $uI + A_c$, etc., reduces the inversion cost from 700 to 82 operations per entry of each block tridiagonal. By combining the scalar tridiagonals, this can be cut to 74 operations.

III. Sound Speed and Velocity Splitting in Steady Generalized Coordinates

The two-dimensional Euler equations can be transformed from Cartesian coordinates to steady curvilinear coordinates using new independent variables

$$\begin{aligned}\tau &= t\\ \xi &= \xi(x,y)\\ \eta &= \eta(x,y).\end{aligned} \tag{3.1}$$

The Euler equations written in steady generalized curvilinear coordinates are

$$\partial_\tau \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} = 0 \tag{3.2}$$

$$\hat{Q} = J^{-1}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ e\end{bmatrix}, \quad \hat{E} = J^{-1}\begin{bmatrix}\rho U\\ \rho u U + \xi_x p\\ \rho v U + \xi_y p\\ U(e+p)\end{bmatrix}, \quad \hat{F} = J^{-1}\begin{bmatrix}\rho V\\ \rho u V + \eta_x p\\ \rho v V + \eta_y p\\ V(e+p)\end{bmatrix}$$

where J is the Jacobian $\xi_x\eta_y - \xi_y\eta_x$ and

$$U = \xi_x u + \xi_y v, \quad V = \eta_x u + \eta_y v$$

are unscaled contravariant velocity components.

Application of the AF implicit scheme, equation (2.4), to equation (3.2) is given by

$$\begin{aligned}\left[I + h\delta_\xi \hat{A}^n + D_\xi^{(2)}\right]\left[I + h\delta_\eta \hat{B}^n + D_\eta^{(2)}\right]\Delta\hat{Q}^n =\\ -\Delta t\left[\delta_\xi \hat{E}^n + \delta_\eta \hat{F}^n + D^{(4)}\hat{Q}^n\right]\end{aligned} \tag{3.3a}$$

where

$$D^{(4)} = \epsilon_e \Delta t J^{-1}[(\nabla_\xi\Delta_\xi)^2 + (\nabla_\eta\Delta_\eta)^2]J \tag{3.3b}$$

and

$$D_\xi^{(2)} = -\epsilon_i \Delta t J^{-1}\nabla_\xi\Delta_\xi J, \quad D_\eta^{(2)} = -\epsilon_i \Delta t J^{-1}\nabla_\eta\Delta_\eta J \tag{3.3c}$$

The Jacobian matrices of $\hat{E}$ and $\hat{F}$ can be given in terms of the Jacobian matrices A and B as

$$\hat{A} = \xi_x A + \xi_y B \tag{3.4a}$$

$$\hat{B} = \eta_x A + \eta_y B \tag{3.4b}$$

Making the substitutions of equation (2.15) into $\hat{A}\hat{Q}$ and $\hat{B}\hat{Q}$ results in

$$\hat{A}\hat{Q} = (UI + \xi_x A_c + \xi_y B_c)\hat{Q} \tag{3.5a}$$

$$\hat{B}\hat{Q} = (VI + \eta_x A_c + \eta_y B_c)\hat{Q} \tag{3.5b}$$

Because of the linear combination of both A_c and B_c being present in equation (3.5), only 3×3 reducibility is achieved. When applied in the implicit AF algorithm, this results in a savings as indicated in ref.1, but not as great a saving as was achieved in Cartesian coordinates. Moreover, in three dimensions one can only reduce to a 4×4 block tridiagonal.

The transformed equations given above have only been transformed in their independent variables. That is, Cartesian velocity or momentum components have been kept, and, as a result, the so-called strong conservation law form is maintained. Suppose, for example, we had used orthogonal body coordinates s and n, and that we had transformed the momentum components as well. We would then obtain equations with coordinate source terms, but otherwise the equations would look like their Cartesian counterparts and the Jacobian matrices could be reduced as before. We can achieve this same effect with equation (3.2) by multiplying through by the matrix

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \xi_x & \xi_y & 0 \\ 0 & \eta_x & \eta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

This matrix transforms u,v momentum components to U,V components. With multiplication of equation (3.2) by C we obtain

$$\partial_\tau J^{-1}\begin{bmatrix} \rho \\ \rho U \\ \rho V \\ e \end{bmatrix} + \partial_\xi J^{-1}\begin{bmatrix} \rho U \\ \rho UU + p(\nabla\xi\cdot\nabla\xi) \\ \rho VU + p(\nabla\xi\cdot\nabla\eta) \\ U(e+p) \end{bmatrix} + \partial_\eta J^{-1}\begin{bmatrix} \rho V \\ \rho UV + p(\nabla\xi\cdot\nabla\eta) \\ \rho VV + p(\nabla\eta\cdot\nabla\eta) \\ V(e+p) \end{bmatrix} = H \tag{3.7}$$

with

$$H = -C[(C_\xi)^{-1}\hat{E} + (C_\eta)^{-1}\hat{F}]$$

These equations look much like their Cartesian counterparts except for the coordinate source term and the appearance of two extra pressure terms.

However, for orthogonal coordinates, $\nabla\xi \cdot \nabla\eta = 0$, and the extra pressure terms then drop out. Although we will not give the details here, one can anticipate from the form of the equations that the flux Jacobians in the implicit algorithm are 2×2 reducible (here the Jacobian matrices are formed with respect to ρU and ρV components, not ρu and ρv components). So if the coordinate generated source term is either treated explicitly or properly distributed in the L_ξ or L_η operators, the inversion efficiency of the previous section can be obtained.

Generally the transformed coordinates will not be orthogonal and multiplication through by C will not lead to the 2×2 reduced matrix operators. However, if we use a transform matrix that brings out a mixture of contravariant and covariant velocity components, the extra pressure terms will also disappear. Although it may not be immediately obvious, we can again obtain the reduced matrix operator structure without requiring the coordinates to be orthogonal.

Consider the transform matrix $\tilde{C}$ given by

$$\tilde{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \eta_y \ell_2^{-1} & -\eta_x \ell_2^{-1} & 0 \\ 0 & -\xi_y \ell_1^{-1} & \xi_x \ell_1^{-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

where

$$\ell_1 = \sqrt{\nabla\xi \cdot \nabla\xi}$$

$$\ell_2 = \sqrt{\nabla\eta \cdot \nabla\eta}$$

and the inner 2×2 of $\tilde{\mathrm{C}}$ transforms u and v into the covariant velocity components

$$\tilde{U} = \ell_2^{-1}(\eta_y u - \eta_x v), \quad \tilde{V} = \ell_1^{-1}(-\xi_y u + \xi_x v)$$

Since the determinant of $\tilde{C}$ is $J l_2^{-1} l_1^{-1}$, its inverse exists for all mappings of interest and is

$$\tilde{C}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \ell_2 y_\eta & -\ell_1 y_\xi & 0 \\ 0 & -\ell_2 x_\eta & \ell_1 x_\xi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Multiplying equation (3.2) by $\tilde{C}$ we obtain

$$\partial_\tau \tilde{Q} + \partial_\xi \tilde{E} + \partial_\eta \tilde{F} = \tilde{H} \tag{3.10}$$

where

$$\tilde{H} = -\tilde{C}(\tilde{C}_\xi{}^{-1}\hat{E} + \tilde{C}_\eta{}^{-1}\hat{F})$$

and $\tilde{Q} = \tilde{C}\hat{Q}, \tilde{E} = \tilde{C}\hat{E}, \tilde{F} = \tilde{C}\hat{F}$. The flux vectors written out are

$$\tilde{Q} = J^{-1}\begin{bmatrix} \rho \\ \rho\tilde{U} \\ \rho\tilde{V} \\ e \end{bmatrix}, \quad \tilde{E} = J^{-1}\begin{bmatrix} \rho U \\ \rho\tilde{U}U + Jp/\ell_2 \\ \rho\tilde{V}U \\ U(e+p) \end{bmatrix}, \quad \tilde{F} = J^{-1}\begin{bmatrix} \rho V \\ \rho\tilde{U}V \\ \rho\tilde{V}V + Jp/\ell_1 \\ V(e+p) \end{bmatrix} \tag{3.11}$$

Unlike equation (3.7), extra pressure terms are not picked up in the momentum equations.

The previous numerical algorithm extended to equation (3.10) is given by

$$\begin{aligned} \left[I + h\delta_\xi \tilde{A}^n + D_\xi^{(2)}\right]\left[I + h\delta_\eta \tilde{B}^n + D_\eta^{(2)}\right]\Delta\tilde{Q}^n = \\ -\Delta t\left[\delta_\xi \tilde{E}^n + \delta_\eta \tilde{F}^n + \tilde{H}^n + D^{(4)}\tilde{Q}^n\right] \end{aligned} \tag{3.12}$$

where we have lagged the source term and where $\tilde{A} = \partial\tilde{E}/\partial\tilde{Q}$ and $\tilde{B} = \partial\tilde{F}/\partial\tilde{Q}$.

It can be verified that $\tilde{A} = \tilde{C}\hat{A}\tilde{C}^{-1}$ and $\tilde{B} = \tilde{C}\hat{B}\tilde{C}^{-1}$. Using these relations as well as equation (3.5) we find

$$\tilde{A}\tilde{Q} = (UI + \tilde{A}_c)\tilde{Q} \quad \tilde{B}\tilde{Q} = (VI + \tilde{B}_c)\tilde{Q} \tag{3.13}$$

with

$$\tilde{A}_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 & 0 \\ J(u^2+v^2)/(2\ell_2) & -U & -V\ell_1/\ell_2 & J/\ell_2 \\ 0 & 0 & 0 & 0 \\ a_{41}^c & a_{42}^c & a_{43}^c & U \end{bmatrix} \tag{3.14a}$$

$$\tilde{B}_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ J(u^2+v^2)/(2\ell_1) & -U\ell_2/\ell_1 & -V & J/\ell_1 \\ b_{41}^c & b_{42}^c & b_{43}^c & V \end{bmatrix} \tag{3.14b}$$

where

$$a^c_{41} = U[(u^2 + v^2)/2 - \gamma p/(\rho(\gamma - 1)^2)]$$
$$a^c_{42} = \ell_2[\gamma p \ell_1^2/(\rho(\gamma - 1)^2) - U^2]/J$$
$$a^c_{43} = \ell_1[\gamma p \ell_{12}^2/(\rho(\gamma - 1)^2) - UV]/J$$
$$b^c_{41} = V[(u^2 + v^2)/2 - \gamma p/(\rho(\gamma - 1)^2)]$$
$$b^c_{42} = \ell_2[\gamma p \ell_{12}^2/(\rho(\gamma - 1)^2) - UV]/J$$
$$b^c_{43} = \ell_1[\gamma p \ell_2^2/(\rho(\gamma - 1)^2) - V^2]/J$$

and

$$\ell_{12} = \sqrt{\nabla\xi \cdot \nabla\eta}$$

Making these substitutions into equation (3.12) we obtain the reduced form of the algorithm

$$\left[I + h\delta_\xi(UI + \tilde{A}_c)^n + D^{(2)}_\xi\right]\left[I + h\delta_\eta(VI + \tilde{B}_c)^n + D^{(2)}_\eta\right]\Delta\tilde{Q}^n = -\Delta t\left[\delta_\xi\tilde{E}^n + \delta_\eta\tilde{F}^n + \tilde{H}^n + D^{(4)}\tilde{Q}\right] \tag{3.15}$$

Finally, to avoid the source term as well as to take advantage of existing codes, one can rewrite these equations in the form

$$\tilde{C}^{-1}\left[I + h\delta_\xi(UI + \tilde{A}_c)^n + D^{(2)}_\xi\right]\left[I + h\delta_\eta(VI + \tilde{B}_c)^n + D^{(2)}_\eta\right]\tilde{C}\Delta\hat{Q}^n = -\Delta t\left[\delta_\xi\hat{E}^n + \delta_\eta\hat{F}^n + D^{(4)}\hat{Q}\right] \tag{3.16}$$

The right-hand-side of this equation is the same as equation (3.3a), while the left-hand-side maintains the update for $\hat{Q}$ quantities rather than $\tilde{Q}$. The advantage of this last form is that one can readily take an existing code for equation (3.2) and modify only the left-hand-side operators so as to take advantage of the reduced inversion work. The steady state solution of equation (3.16) is clearly identical to that of equation (3.3a).

IV. Results

The algorithm given by equation (3.16) was tested on a NACA 0012 airfoil under transonic flow conditions in order to verify that no adverse stability effects are incurred by using the matrix reductions. Versions of the basic NASA Ames Research Center AIR2D/ARC2D, which solve equation (3.3a), were modified to the form of equation (3.16). As noted earlier, only that portion of the code which deals with the left-hand-side operator has to be altered in order to implement the new algorithm. The basic code is described in refs. 5 and 6.

A "C-type" topology was used for the airfoil calculations. The grids were generated with either a hyperbolic grid solver (ref. 12 and 13) that imposes orthogonality up to numerical errors of truncation and smoothing, or an algebraic construction that uses a method similar to the control function approach of Eiseman (ref. 14). In all cases, the far-field boundaries were placed approximately 20 chords from the body. Boundary conditions on the body, wake, and freestream are further described in refs. 5 and 6.

As a first calculation, a relatively coarse grid was used which was generated with the hyperbolic grid solver. This grid has 157 points in the ξ-direction and 33 points in the η-direction (fig. 2a and 2b). The pressure distribution on the NACA 0012 for $M_\infty = 0.8$ and $\alpha = 0.$ is shown in Figure 2c. This distribution was computed using both the standard algorithm, equation (3.3a), and the reduced matrix form, equation (3.16). Both algorithms were run using Euler implicit differencing, which is first-order accurate in time. As shown in fig. 2d, the residual histories are virtually identical. However, the standard algorithm required 120 CPU seconds of CRAY X-MP time per 1000 iterations and the new algorithm required 54 CPU seconds per 1000 iterations.

Numerical experimentation, in which Δt was adjusted over a wide range, has confirmed that the new algorithm is as stable as the standard algorithm. As an example of the robustness of the formulation using equation (3.16), a much finer grid was used to compute the previous case. Figures 3a and 3b show a view of the grid with 249 points in the ξ-direction and 50 points in the η-direction. The solution on this grid (Figures 3c-3d) was computed using a spacially varying time step scaled by $\frac{1}{1+\sqrt{J}}$ where J is the metric Jacobian. For this calculation, the drag coefficient was converged to 5 digits in 600 iterations or 68 seconds of CRAY X-MP time. Figure 3e shows the residual history for this calculation.

Recently, Beam and Bailey have reported in private communication (see also ref. 6) that significantly faster convergence can be obtained by incorporating the fourth-order smoothing terms implicitly into the Beam and Warming algorithm. For the standard algorithm, imposing fourth-order dissipation implicitly necessitates 4×4 block pentadiagonal inversions which are relatively expensive. The reduced matrix algorithm, however, requires only scalar and 2×2 block pentadiagonal inversions which are much less costly. In fact, a special 2×2 block pentadiagonal inversion algorithm, coded for this algorithm, requires only 107 arithmetic operations per entry.

In testing the sound speed-velocity splitting algorithm with implicit fourth order dissipation, a more advanced form of numerical filter was implemented. This filter uses both second- and fourth-order differences such that the second-order difference is dominant near shock waves. The spectral radius of the flux Jacobian matrices is used as a scaling to the smoothing coefficients in an attempt to mimic the dissipative nature of second-order flux split upwind schemes (ref. 15). Full details are given in ref. 6.

As a test case, an algebraic grid was used to compute the flow field around a NACA 0012 airfoil at $M_\infty = 0.8$ and $\alpha = 1.25$ degrees. The grid has 193 points in the ξ-direction and 33 points in the η-direction. The grid and flow field solution are shown in fig. 4a through 4d. The residual history (fig. 4e) shows the rapid convergence. For this particular case, the CPU time on the CRAY X-MP was 93 seconds per 1000 iterations. This compares with 65 seconds per 1000 iterations using the tridiagonal version of the new algorithm with constant smoothing coefficients which had much slower convergence. It should be noted that the smoothing filter used for the pentadiagonal version required 16 seconds more CPU time than the same smoothing with a constant coefficient filter.

Our computational results verified that the reduced algorithm was every bit as numerically stable as the standard algorithm. This experience is similar to what was observed previously by Steger on an algorithm version that reduced to a 3×3 block and a scalar. Here, with reduction to the 2×2 block, just over a factor of two was saved in overall computer time by using the reduced algorithm (3.16) in place of equation (3.3a). The precise savings that can be obtained is, of course, machine and coding dependent and a larger improvement should be obtained in three dimensions.

V. Conclusions

By substituting similar reducible matrices for the Jacobian matrices of the flux vectors, a substantial reduction has been achieved in the arithmetic operations needed in solving approximate factorization implicit algorithms such as Beam-Warming. Numerical calculations indicate that the new similarity split matrices can apparently be used without any loss of numerical stability, although time accuracy can be degraded unless additional steps are taken. The new method can be readily retrofit within existing codes, and can likely be extended to viscous flow calculations as well.

Appendix A
Extension of Sound Speed and Velocity Splitting to the Navier-Stokes Equations

The sound speed and velocity splitting concept can be applied to the Navier-Stokes equations because of the special form of the viscous Jacobians. The Navier-Stokes equations in strong conservation law form can be represented as

$$\partial_t Q + \partial_x E + \partial_y F = \partial_x E_{vx} + \partial_x E_{vy} + \partial_y F_{vx} + \partial_y F_{vy} \tag{A.1}$$

where Q, E, and F have their previous definitions. The viscous flux terms are given by

$$E_{vx} = \begin{bmatrix} 0 \\ (4/3)\mu u_x \\ \mu v_x \\ E_{vx4} \end{bmatrix}, \quad E_{vy} = \begin{bmatrix} 0 \\ -(2/3)\mu v_y \\ \mu u_y \\ E_{vy4} \end{bmatrix} \tag{A.2a}$$

$$F_{vx} = \begin{bmatrix} 0 \\ \mu v_x \\ -(2/3)\mu u_x \\ F_{vx4} \end{bmatrix}, \quad F_{vy} = \begin{bmatrix} 0 \\ \mu u_y \\ (4/3)\mu v_y \\ F_{vy4} \end{bmatrix} \tag{A.2a}$$

with

$$\begin{aligned} E_{vx4} &= (\mu/2)\partial_x[(4/3)u^2 + v^2] + \mu\beta\partial_x c^2 \\ E_{vy4} &= \mu v u_y - (2/3)\mu u v_y \\ F_{vx4} &= \mu u v_x - (2/3)\mu v u_x \\ F_{vy4} &= (\mu/2)\partial_y[u^2 + (4/3)v^2] + \mu\beta\partial_y c^2 \end{aligned}$$

and

$$\beta = Pr^{-1}(\gamma - 1)^{-1}$$

Beam and Warming have developed a general class of integration techniques for equation (A.1) which is illustrated here for Euler implicit time differencing as

$$\begin{aligned} \Delta Q^n - \Delta t\big[\delta_x \Delta E^n + \delta_y \Delta F^n - \bar{\delta}_x \Delta E^n_{vx} - \bar{\delta}_y \Delta F^n_{vy}\big] \\ = -\Delta t\big[\delta_x E^n + \delta_y F^n - \bar{\delta}_x(E^n_{vx} + E^n_{vy}) - \bar{\delta}_y(F^n_{vx} + F^n_{vy})\big] \end{aligned} \tag{A.3}$$

where $\Delta Q^n = Q^{n+1} - Q^n$ and $\bar{\delta}$ is a midpoint operator. In this algorithm the cross derivative flux terms E_{vy} and F_{vx} are treated explicitly. As discussed by Beam and Warming, this circumvents the need to include an implicit viscous Jacobain for these cross terms, and it also maintains unconditional stablility for the scalar model equation.

The spacial flux terms have been locally linearized in time. In locally linearizing these terms Beam and Warming expand the viscous terms in a Taylor series using both the function and its derivative, as for example

$$\Delta E_{vx}^n = E_{vx}^{n+1} - E_{vx}^n = \left[\frac{\partial E_{vx}}{\partial Q}\right]^n (Q^{n+1} - Q^n) + \left[\frac{\partial E_{vx}}{\partial Q_x}\right]^n \partial_x (Q^{n+1} - Q^n) \tag{A.4}$$

Alternately, these terms can be expanded in terms of Q alone, for example

$$\Delta E_{vx}^n = \left[\frac{\partial E_{vx}}{\partial Q}\right]^n (Q^{n+1} - Q^n) \tag{A.5}$$

where $\frac{\partial E_{vx}}{\partial Q}$ is now a differential operator. This latter form will be used here. Typically μ is not linearized in time and the Prandtl number is treated as if it were a constant value as far as time linearization is concerned. For a term such as $\mu \partial_y u$, μ is frozen in time and $\mu_0 \partial_y u$ expanded such as

$$\mu_0 \partial_y (\rho u/\rho) = \mu_0 \partial_y [(\rho_0 u_0/\rho_0) - (\rho_0 u_0/\rho_0{}^2)(\rho - \rho_0) + \rho_0{}^{-1}(\rho u - \rho_0 u_0)]$$

or in general

$$\partial_x E_{vx} = \partial_x [E_{vx}|_0 + \mu_0 \partial_x (A_{vx}|_0 (Q - Q_0))]_0 \tag{A.6a}$$

$$\partial_y F_{vy} = \partial_y [F_{vy}|_0 + \mu_0 \partial_y (B_{vy}|_0 (Q - Q_0))]_0 \tag{A.6b}$$

with

$$A_{vx} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -(4/3)(u/\rho) & (4/3)\rho^{-1} & 0 & 0 \\ -(v/\rho) & 0 & \rho^{-1} & 0 \\ a_{x1} & a_{x2} & a_{x3} & \beta/\rho \end{bmatrix}, \quad B_{vy} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -(u/\rho) & \rho^{-1} & 0 & 0 \\ -(4/3)(v/\rho) & 0 & (4/3)\rho^{-1} & 0 \\ b_{x1} & b_{x2} & b_{x3} & \beta/\rho \end{bmatrix} \tag{A.7}$$

and

$$a_{x1} = -(4/3)[u^2/\rho] - [v^2/\rho] + \beta[-(e/\rho) + (u^2 + v^2)/\rho]$$

$$a_{x2} = [(4/3) - \beta](u/\rho)$$
$$a_{x3} = (1 - \beta)(v/\rho)$$
$$b_{y1} = -[u^2/\rho] - (4/3)[v^2/\rho] + \beta[-(e/\rho) + (u^2 + v^2)/\rho]$$
$$b_{y2} = (1 - \beta)(u/\rho)$$
$$b_{y3} = [(4/3) - \beta](v/\rho)$$

With the introduction of the sound speed and velocity matrix substitutions for the convection Jacobian matrices, we can obtain operators L_x and L_y that are as reducible as before because of the form of A_{vx} and B_{vy}. To illustrate this we will drop the numerical dissipation terms as before. Then equation (A.3) becomes

$$L_x L_y \Delta Q^n = -\Delta t[\delta_x E^n + \delta_y F^n - \bar{\delta}_x(E^n_{vx} + E^n_{vy}) - \bar{\delta}_y(F^n_{vx} + F^n_{vy})] \tag{A.8}$$

where

$$L_x = I + \Delta t \delta_x \left(\begin{bmatrix} u & 0 & 0 & 0 \\ a^c_{21} & u + a^c_{22} & a^c_{23} & a^c_{24} \\ 0 & 0 & u & 0 \\ a^c_{41} & a^c_{42} & a^c_{43} & u + a^c_{44} \end{bmatrix} + \delta_x \begin{bmatrix} 0 & 0 & 0 & 0 \\ a^v_{21} & a^v_{22} & 0 & 0 \\ a^v_{31} & 0 & a^v_{33} & 0 \\ a^v_{41} & a^v_{42} & a^v_{43} & a^v_{44} \end{bmatrix} \right)$$

$$L_y = I + \Delta t \delta_y \left(\begin{bmatrix} v & 0 & 0 & 0 \\ 0 & v & 0 & 0 \\ b^c_{31} & b^c_{32} & v + b^c_{33} & b^c_{34} \\ b^c_{41} & b^c_{42} & b^c_{43} & v + b^c_{44} \end{bmatrix} + \delta_y \begin{bmatrix} 0 & 0 & 0 & 0 \\ b^v_{21} & b^v_{22} & 0 & 0 \\ b^v_{31} & 0 & b^v_{33} & 0 \\ b^v_{41} & b^v_{42} & b^v_{43} & b^v_{44} \end{bmatrix} \right)$$

Here I is the identity matrix and a^c, etc., are elements of A^c etc. Keep in mind that the viscous terms have been second-order accurate differenced to maintain tridiagonal structure using midpoint operators $\bar{\delta}$.

We can now examine the structure of L_x for example, and show that the equations uncouple to two scalar and one 2×2 block tridiagonal. The first row of L_x is clearly uncoupled and can be solved for as a scalar tridiagonal. Its solution is then used to update the right-hand-side (RHS) of rows 2, 3, and 4. With the first column of the matrices brought to the RHS, the third row is seen to be uncoupled and can thus be solved as another scalar tridiagonal. The RHS terms of the second and fourth rows can now be updated and what remains is a 2×2 block tridiagonal to be solved. The L_y operator inverts in a similar fashion.

Thus, the addition of the viscous terms in Cartesian coordinates does not prevent the decoupling technique that was successfully used for the inviscid gasdynamic equations. Preliminary work shows that, using the local transformation of Section III, the same decoupling may be possible for the Navier-Stokes equations in generalized coordinates. This matter is being further investigated.

Appendix B
Stability of Sound Speed and Velocity Splitting

In order to study the stability of the algorithm proposed in Section II, the one-dimensional Euler equations will be tested for linear stability using a Fourier (or matrix) analysis.

The one-dimensional Euler equations in Cartesian coordinates are given by

$$\partial_t Q + \partial_x E = 0 \tag{B.1}$$

where

$$Q = \begin{bmatrix} \rho \\ \rho u \\ e \end{bmatrix}, \quad E = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(e+p) \end{bmatrix}, \tag{B.2}$$

A frozen coefficient form of equation (B.1) is given by

$$\partial_t Q + A^* \partial_x E = 0 \tag{B.3}$$

where A is the true Jacobian, $[\partial E/\partial Q]$, and * denotes that A is evaluated using a frozen value of Q, Q^*. Implicit first-order-accurate differencing of equation (B.3) is given in delta form as

$$\left[I + hA^* \delta_x\right](Q^{n+1} - Q^n) = -hA^* \delta_x Q^n \tag{B.4}$$

In our sound speed-velocity splitting scheme, we replace the A^* matrix of the left-hand-side operator with the similarity matrices $u^* I + A_c^*$ to obtain

$$\left[I + h(u^* I + A_c^*)\delta_x\right](Q^{n+1} - Q^n) = -hA^* \delta_x Q^n \tag{B.5}$$

The right-hand-side matrix A^* is not altered in our procedure, and must not be changed in equation (B.5) as it represents the correct linearization of E about Q^*. The question to be addressed is, is equation (B.5) as stable as equation (B.4) now that the implicit operator has been altered?

If we were to perform a stability analysis of equation (B.4), we would diagonalize A^* using its eigenvectors. We can then readily perform the stability analysis for difference equations corresponding to scalar partial differential equations. For equation (B.5) we can use the eigenvectors of A_c^* to diagonalize $u^*I + A_c^*$, but the right-hand-side matrix will not be brought into diagonal form. Nevertheless, we find that the resulting equations can still be analyzed.

The matrix A_c for one-dimensional flow is given by

$$A_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 \\ u^2/2 & -u & 1 \\ a_{41}^c & a_{42}^c & u \end{bmatrix} \tag{B.6}$$

where

$$a_{41}^c = u[u^2/2 - c^2/(\gamma - 1)^2]$$
$$a_{42}^c = -u^2 + c^2/(\gamma - 1)^2$$

Eigenvectors matrices of A_c are given by

$$X^{-1} = \begin{bmatrix} -[(\gamma-1)u^2 + 2cu]/(4c) & [(\gamma-1)u + c]/(2c) & -(\gamma-1)/(2c) \\ [(\gamma-1)u^2 - 2cu]/(4c) & -[(\gamma-1)u - c]/(2c) & (\gamma-1)/(2c) \\ 1 & 0 & 0 \end{bmatrix} \tag{B.7}$$

$$X = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & u \\ u - (c/(\gamma-1)) & u + (c/(\gamma-1)) & u^2/2 \end{bmatrix} \tag{B.8}$$

and

$$\Lambda_{A_c} = X^{-1}A_cX = \begin{bmatrix} -c & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{B.9}$$

Multiplying equation (B.5) by the frozen-coefficient inverse of the eigenvectors $(X^*)^{-1}$ gives

$$\left[I + h(u^*I + \Lambda_{A_c}^*)\delta_x\right](X^*)^{-1}(Q^{n+1} - Q^n) = -h(X^*)^{-1}A^*X^*\delta_x(X^*)^{-1}Q^n \tag{B.10}$$

or in nondelta form with $W = (X^*)^{-1}Q$

$$\left[I + h(u^*I + \Lambda_{A_c}^*)\delta_x\right]W^{n+1} = I - h[(X^*)^{-1}A^*X^* - u^*I - \Lambda_{A_c}^*]\delta_x W^n \tag{B.11}$$

However, $(X^*)^{-1}A^*X^* - u^*I - \Lambda^*_{A_c}$ is a very simple matrix

$$(X^*)^{-1}A^*X^* - u^*I - \Lambda^*_{A_c} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \tag{B.12}$$

Assuming periodic boundary conditions so as to use a Fourier stability analysis (or the matrix stability method using circulant matrices), the finite difference operator δ_x can be transformed to

$$\frac{2isin(\theta_j)}{2\Delta x} = \kappa$$

and $\tilde{W}$ denotes Fourier transformation of W (i.e. transformation via the eigenvectors of a circulant matrix).

Applying the Fourier method to equation (B.11) and taking the inverse of the left hand side operator we obtain

$$\tilde{W}^{n+1} = \begin{bmatrix} \frac{1}{1+ih\kappa(u+c)} & 0 & 0 \\ 0 & \frac{1}{1+ih\kappa(u-c)} & 0 \\ 0 & 0 & \frac{1}{1+ih\kappa u} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -ih\kappa & -ih\kappa & 1 \end{bmatrix} \tilde{W}^n \tag{B.13}$$

or

$$\tilde{W}^{n+1} = \begin{bmatrix} \frac{1}{1+ih\kappa(u+c)} & 0 & 0 \\ 0 & \frac{1}{1+ih\kappa(u-c)} & 0 \\ \frac{-ih\kappa}{1+ih\kappa u} & \frac{-ih\kappa}{1+ih\kappa u} & \frac{1}{1+ih\kappa u} \end{bmatrix} \tilde{W}^n \tag{B.14}$$

The amplification matrix associated with equation (B.14) is lower triangular. Therefore, its eigenvalues are the diagonal elements, and these clearly have modulus less than one for any positive value of $h = \Delta t$. In fact, these eigenvalues are identical to those obtained with the standard scheme given by equation (B.4). Thus the necessary condition for stability is met for any values of h. A sufficient condition for stability requires bounding the norm of the 3×3 amplification matrix. If we observe the $3,1$ and $3,2$ elements, it is clear that the ℓ_∞ norm can be unbounded for $u \to 0$ and $h \to \infty$; however, even in this norm, powers of the amplification matrix will be quickly bounded unless u is identically zero.

It is interesting to note that if one were to form A_c using the Jacobian of E_c, the resulting algorithm is unconditionally unstable. In this case A_c is given as

$$A_c = (\gamma - 1)\begin{bmatrix} 0 & 0 & 0 \\ u^2/2 & -u & 1 \\ a^c_{41} & a^c_{42} & u \end{bmatrix} \tag{B.15}$$

where

$$a^c_{41} = u[u^2/2 - c^2/\gamma(\gamma - 1)]$$

$$a^c_{42} = -u^2 + c^2/\gamma(\gamma - 1)$$

If we procede in a fashion similar to the above, the resultant representation in Fourier space for equation (B.5) is

$$\tilde{W}^{n+1} = \begin{bmatrix} \frac{2\beta + ih\kappa c}{2\beta(1+ih\kappa(u+\alpha c))} & \frac{ih\kappa c}{2\beta(1+ih\kappa(u+\alpha c))} & 0 \\ \frac{-ih\kappa c}{2\beta(1+ih\kappa(u-\alpha c))} & \frac{2\beta - ih\kappa c}{2\beta(1+ih\kappa(u-\alpha c))} & 0 \\ \frac{-ih\kappa}{1+ih\kappa u} & \frac{-ih\kappa}{1+ih\kappa u} & \frac{1}{1+ih\kappa u} \end{bmatrix} \tilde{W}^{n} \tag{B.16}$$

where

$$\alpha = \sqrt{\gamma(\gamma - 1)} \quad \beta = \sqrt{\frac{\gamma - 1}{\gamma}}$$

The maximum eigenvalue of the amplification matrix in equation (B.16) can be shown to be greater than one for all Δt's greater than zero, and the necessary condition for stability cannot be met. In actual numerical experimentation where numerical dissipation is added, small values of Δt could be used to obtain stable results.

References

1. Steger, J. L.: Coefficient Matrices for the Implicit Finite Difference Solution of the Inviscid Fluid Conservation Law Equations. Computer Meth. in Appl. Mech. Eng., vol. 13, 1978 pp.175-188.

2. Beam, R. M.; and Warming, R. F.: An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law-Form. J. Comp. Phys., vol. 22, Sept. 1976, pp. 87-110.

3. Warming, R. F.; and Beam, R. M.: On the Construction and Application of Implicit Factored Schemes for Conservation Laws. Symposium of CFD, New York, April 16-17, 1977, SIAM-AMS Proceedings, vol. 11, 1977.

4. Reddy, K. C.: Pseudospectral Approximation in a Three-Dimensional Navier-Stokes Code. AIAA J., vol. 21, no. 8, Aug. 1983.

5. Steger, J. L.: Implicit Finite Difference Simulation of Flow About Arbitrary Two Dimensional Geometries. AIAA J., vol. 16, No. 7, July 1978.

6. Pulliam, T. H.: Euler and Thin Layer Navier Stokes Codes : ARC2D, ARC3D, Notes for Computational Fluid Dynamics User's Workshop, The University of Tennessee Space Institute, Tullahoma, Tenn., March 12-16, 1984.

7. Pulliam, T. H.; and Chaussee, D. S.: A Diagonal Form of an Implicit Approximate-Factorization Algorithm. J. Comp. Phys., vol. 39, no. 2, Feb. 1981, p.347.

8. Chaussee, D. S.; and Pulliam, T. H.: A Diagonal Form of an Implicit Approximate Factorization Algorithm with Application to a Two Dimensional Inlet. AIAA J., Vol. 19, no. 2, Feb. 1981, p.153.

9. Steger, J. L.; Pulliam, T. H.; and Chima, R. V.: An Implicit Finite Difference Code for Inviscid and Viscous Cascade Flow. AIAA paper 80-1427, 1980.

10. Coakley, T.: Numerical Method for Gas Dynamics Combining Characteristics and Conservation Concepts. AIAA paper 81-1257, 1981.

11. Merriam, M. L.: On the Inversion of Block-Tridiagonals without Storage Constraints. NASA TM-84228, March 1982.

12. Steger, J. L.; and Chaussee, D.: Generation of Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations. SIAM J. Sci. Stat. Comput., vol. 1, Dec. 1980, pp 431-437.

13. Kinsey, D.; and Barth, T. J.: Description of a Hyperbolic Grid Generating Procedure for Arbitrary Two-Dimensional Bodies, AFWAL-TM, in press.

14. Eiseman, P.: Geometric Methods in Computational Fluid Dynamics, ICASE Report 80-11, Apr. 1980.

15. Steger, J. L.: A Preliminary Study of Relaxation Methods for the Inviscid Conservative Gasdynamics Equations Using Flux Vector Splitting, NASA CR-3415, March 1981.

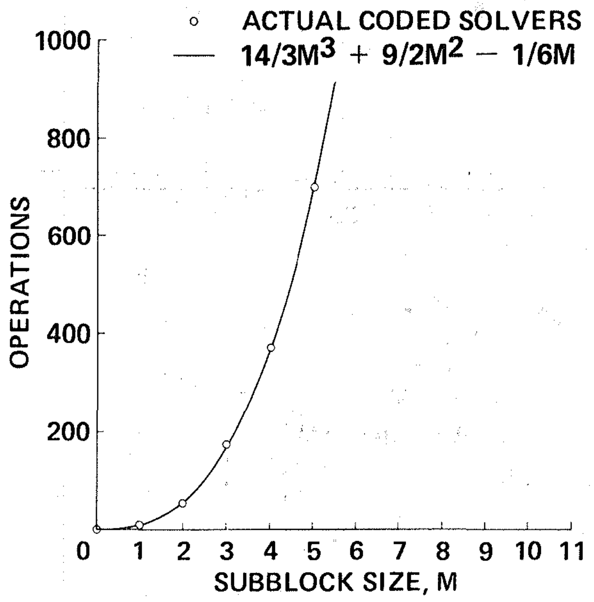
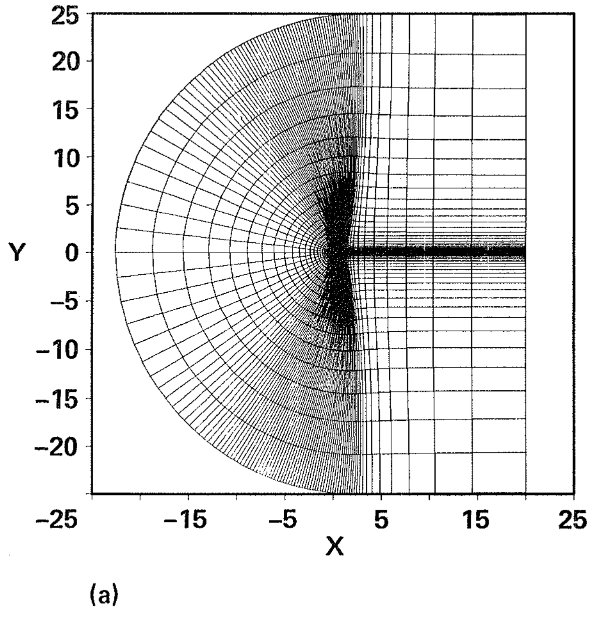
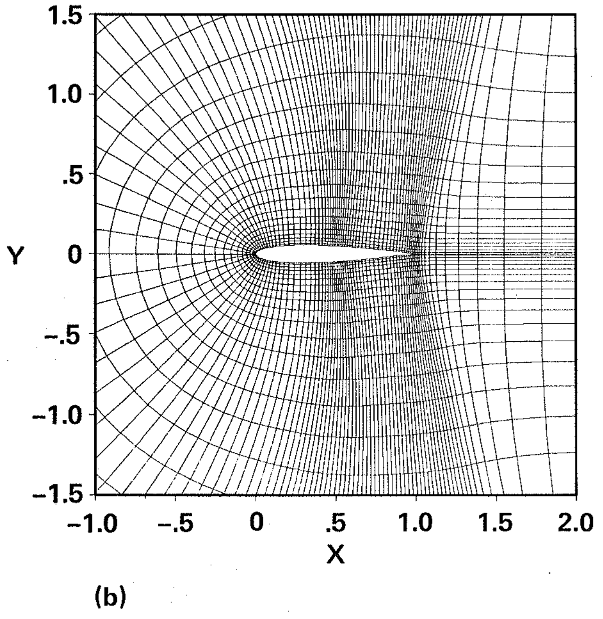
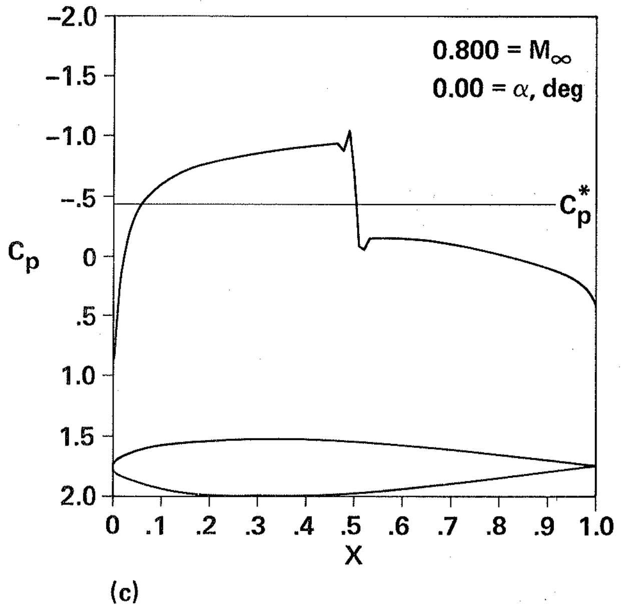


Figure 1. - Arithmetic operation counts of block-tridiagonals using L-U decomposition.







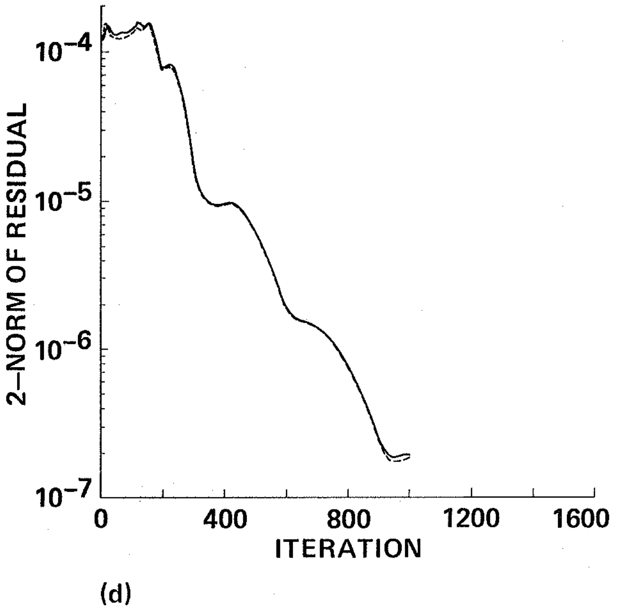


Figure 2. - Coarse grid solution about NACA 0012 airfoil using reduced matrix algorithm. (a) Overview of 157 $\times$ 33 grid generated using hyperbolic grid generator. (b) Detail near body. (c) Pressure distribution ($M_\infty = 0.8$ and $\alpha = 0.0$). (d) Residual history comparison between reduced matrix and standard algorithms.

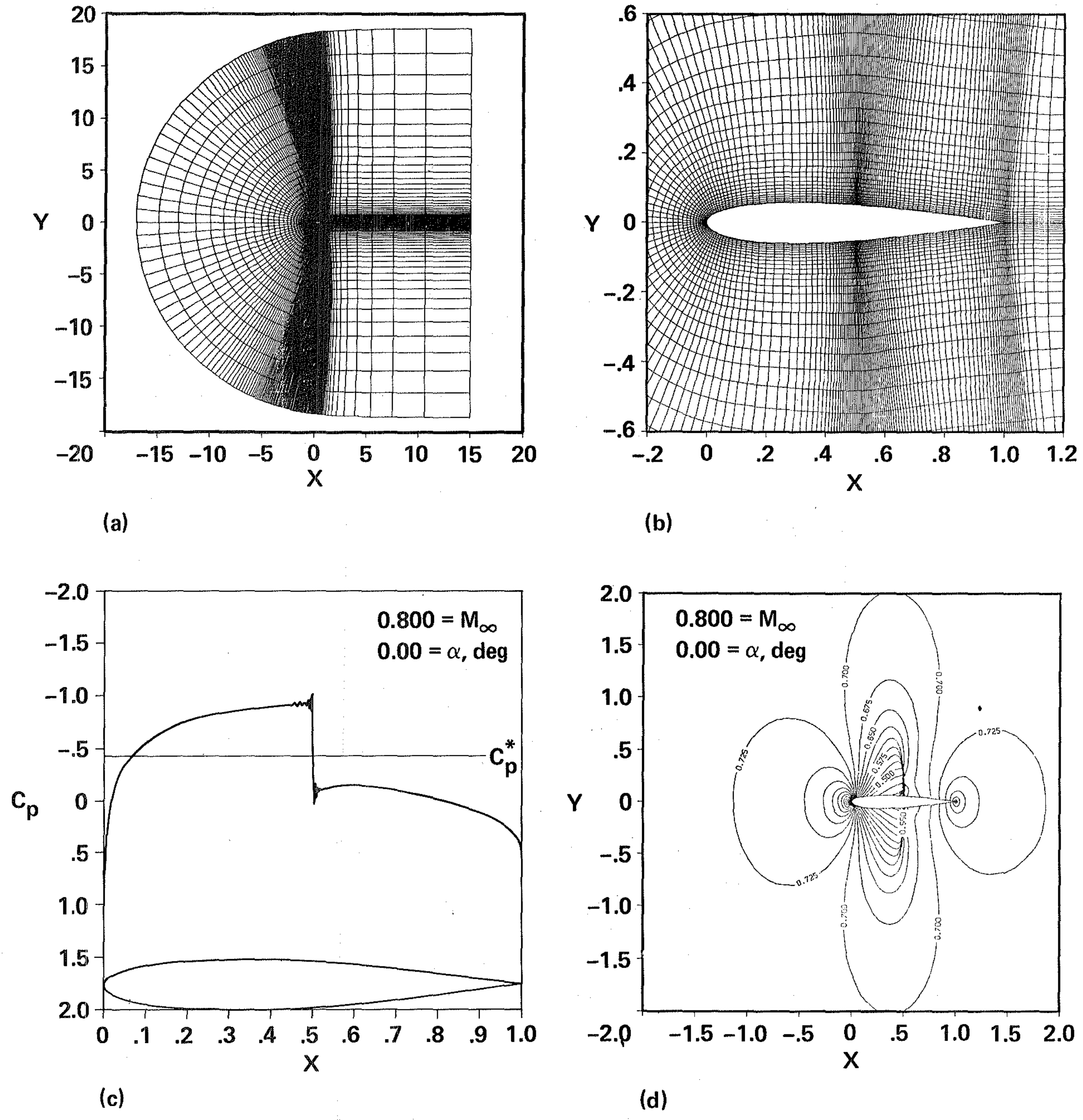
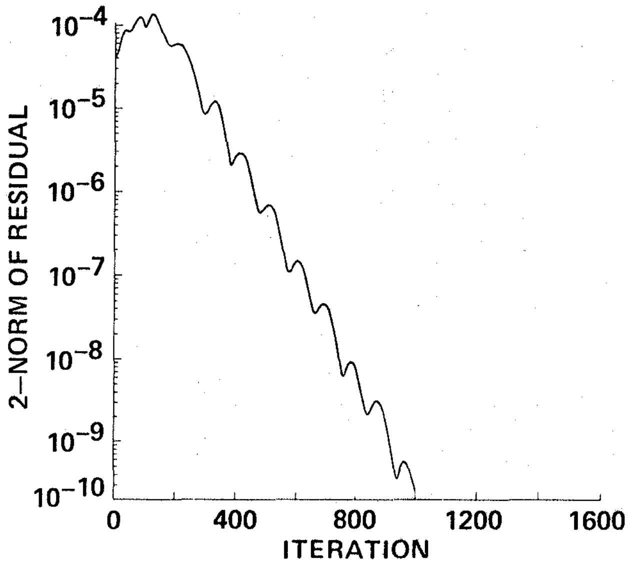


Figure 3. - Fine grid solution about NACA 0012 airfoil using reduced matrix algorithm with scaled Δt. (a) Overview of 249 $\times$ 50 grid generated using hyperbolic grid generator. (b) Detail near body. (c) Pressure distribution on airfoil ($M_\infty = 0.8$ and $\alpha = 0.0$). (d) Pressure contours near airfoil.



(e) Residual history for reduced matrix algorithm with scaled Δt.

Figure 3. - Concluded.

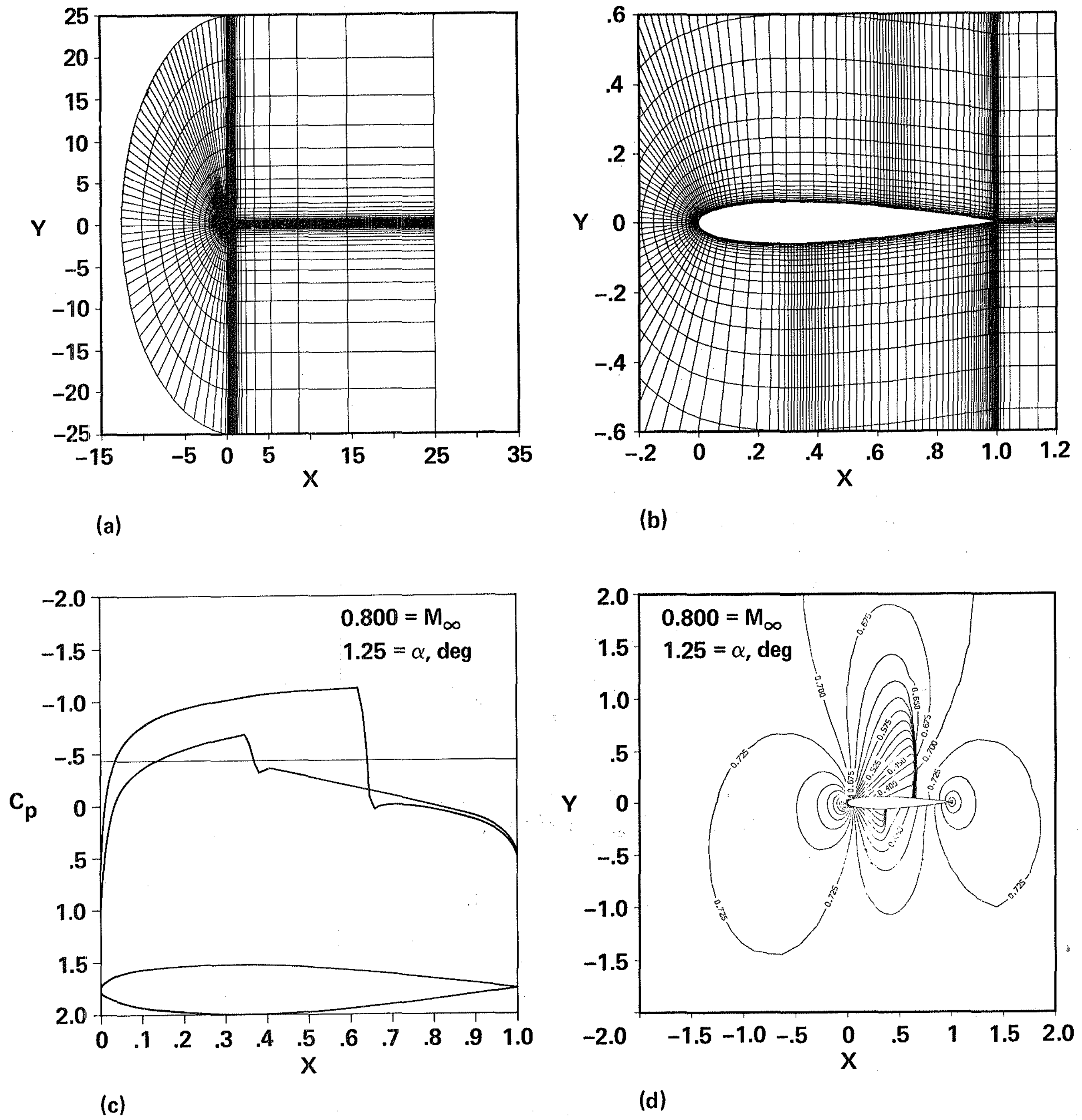
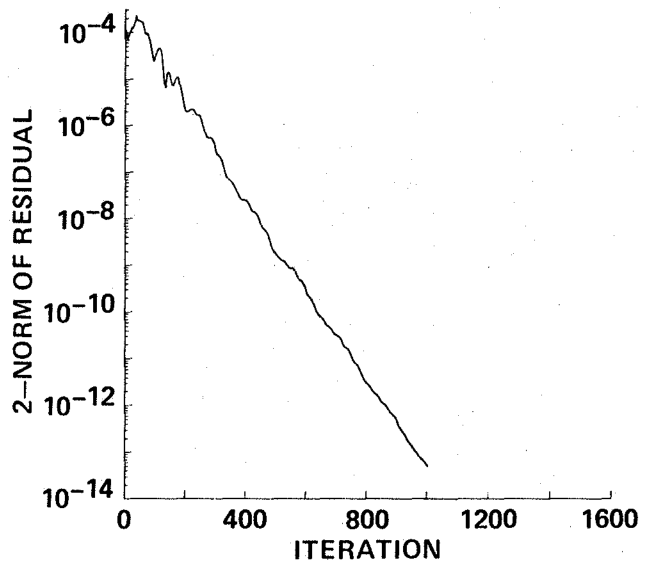


Figure 4. - Solution about NACA 0012 airfoil using advanced numerical filter, scaled Δt, and pentadiagonal inversions. (a) Overview of 193×33 grid generated using algebraic grid generator. (b) Detail near body.(c) Pressure distribution on airfoil ($M_\infty = 0.8$ and $\alpha = 1.25$). (d) Residual history for reduced matrix algorithm.



(e) Residual history for reduced matrix algorithm with scaled Δt and pentadiagonal inversions.

Figure 4. - Concluded.

1. Report No. NASA TM-85957	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle AN EFFICIENT APPROXIMATE FACTORIZATION IMPLICIT SCHEME FOR THE EQUATIONS OF GASDYNAMICS		5. Report Date June 1984
		6. Performing Organization Code
7. Author(s) Timothy J. Barth (Informatics General Corp., Sunnyvale, Calif.) and Joseph L. Steger		8. Performing Organization Report No. A-9744
		10. Work Unit No. T6465
9. Performing Organization Name and Address Ames Research Center Moffett Field, CA 94035		11. Contract or Grant No.
		13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		14. Sponsoring Agency Code 505-3101

15. Supplementary Notes

Point of Contact: Joseph L. Steger, Ames Research Center, M/S 202A-1, Moffett Field, CA 94035 (415) 965-6459 or FTS 448-6459

16. Abstract

An efficient implicit finite-difference algorithm for the gasdynamic equations utilizing matrix reduction techniques is presented. A significant reduction in arithmetic operations is achieved while maintaining the same favorable stability characteristics and generality found in the Beam and Warming approximate factorization algorithm. Steady-state solutions to the conservative Euler equations in generalized coordinates are obtained for transonic flows about a NACA 0012 airfoil. The theoretical extension of the matrix reduction technique to the full Navier-Stokes equations in Cartesian coordinates is presented in detail. Linear stability, using a Fourier stability analysis, is demonstrated and discussed for the one-dimensional Euler equations. It is shown that the method offers advantages over the conventional Beam and Warming scheme and can retrofit existing Beam and Warming codes with minimal effort.

17. Key Words (Suggested by Author(s)) Matrix reduction technique Euler equations Finite-difference Compressible flow		18. Distribution Statement Unlimited Subject Category - 64	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 35	22. Price* A02

*For sale by the National Technical Information Service, Springfield, Virginia 22161

End of Document