NASA
Technical
Paper
2523

January 1987

Efficient Solutions to the Euler Equations for Supersonic Flow With Embedded Subsonic Regions

Robert W. Walters
and Douglas L. Dwoyer

NASA

NASA Technical Paper 2523

1987

Efficient Solutions to the Euler Equations for Supersonic Flow With Embedded Subsonic Regions

Robert W. Walters
and Douglas L. Dwoyer

Langley Research Center
Hampton, Virginia

National Aeronautics
and Space Administration

Scientific and Technical Information Branch

Summary

A line Gauss-Seidel (LGS) relaxation algorithm in conjunction with a one-parameter family of upwind discretizations of the Euler equations in two dimensions is described. Convergence of the basic algorithm to the steady state is quadratic for fully supersonic flows and is linear for other flows. This is in contrast to the block alternating direction implicit methods (either central or upwind differenced) and the upwind biased relaxation schemes, all of which converge linearly, independent of the flow regime. Moreover, the algorithm presented herein is easily coupled with methods to detect regions of subsonic flow embedded in supersonic flow. This allows marching by lines in the supersonic regions, converging each line quadratically, and iterating in the subsonic regions, and yields a very efficient iteration strategy. Numerical results are presented for two-dimensional supersonic and transonic flows containing oblique and normal shock waves which confirm the efficiency of the iteration strategy.

Introduction

A large class of problems of fundamental importance to the field of computational aerodynamics is that of simulating the flow of an inviscid, compressible gas in the transonic and supersonic regimes. The most popular approaches used in obtaining steady-state solutions to these problems have been the implicit spatially split methods of the approximate factorization (AF) type (refs. 1 and 2) and the highly vectorizable explicit schemes such as the multistage Runge-Kutta method used by Jameson and Bakes (ref. 3) and the predictor-corrector scheme of MacCormack (ref. 4). A motivation for algorithm research stems from the fact that the highly vectorizable explicit schemes suffer from stringent stability restrictions, and, though the implicit AF methods are unconditionally stable in two dimensions, they require an optimal set of iteration parameters for rapid convergence which are difficult and time-consuming to obtain. (See ref. 5.) Moreover, the implicit central-differenced AF schemes are unstable for three-dimensional Euler equations. (See ref. 6.)

The recent emergence of upwind-differencing technology for the Euler equations has opened the door to a new class of solution strategies aimed at improving computational efficiency. Since upwind differencing is more costly per iteration to implement than central differencing, overall efficiency gains must be derived from improved convergence rates. This is made possible by the improved conditioning of the system of difference equations afforded by the more physical upwind discretization.

Some preliminary work by Chakravarthy (ref. 7) and Van Leer and Mulder (ref. 8) using relaxation methods and upwind schemes has been promising. In the present study, the fact that upwind differencing closely models the propagation of information along characteristics is exploited by choosing a combination of line relaxation and upwind discretization such that, for fully supersonic flow in the streamwise (marching) direction, the algorithm becomes a direct solver of the linearized problem. That is, the algorithms of the family presented here all revert to an efficient implementation of Newton's method in the supersonic regime and result in quadratic convergence to the steady state.

Further efficiency gains are realized in supersonic regions by iterating on each line to remove the linearization error before proceeding to the next line, thus solving the problem in a marching fashion. This can be accomplished since, in the absence of waves propagating adverse to the marching direction, the combinations of line relaxation and upwind discretization presented in this study result in the uncoupling of the discrete algebraic problem by lines. Thus, the computational efficiency of space marching methods can be effectively recovered. For problems with mixed supersonic-subsonic regions, the solution strategy can be extended to detect the flow-regime interfaces, march in the supersonic regions, and iterate with line Gauss-Seidel (LGS) in the subsonic regions. A comparison of CPU times with and without the new iteration strategy is provided in the section entitled "Results."

The authors would like to acknowledge several fruitful discussions on upwind differencing with J. L. Thomas of Langley Research Center and Bram van Leer of the Delft University of Technology, in the Netherlands.

Symbols

A, B	Jacobian matrices
c	speed of sound
e	total energy per unit volume
f, g	mass, momentum, and energy fluxes
I	identity matrix
j, k	x, y grid-line indices
M	local Mach number
p	pressure
q	conserved variables
R	residual, Riemann invariant
s	entropy

t	time
u, v	x, y Cartesian velocity components
$U1, U2$	upper diagonals
x, y	Cartesian coordinates
γ	ratio of specific heats
Δ, ∇	finite-difference operators
$\Delta x, \Delta y$	mesh increments in x- and y-directions
δq	change in q over a time increment
δt	time increment
κ	spatial differencing parameter
ν	Mach number switch
ρ	density
ϕ	spatial differencing switch

Subscripts:

A, B, L, R	state quantities from above, below, left, and right, respectively
j, k	spatial indices (1 to J and 1 to K)
x, y	x, y coordinate directions
∞	free stream

Superscripts:

$\pm$	positive and negative flux contributions
n	iteration index

Spatial Discretization

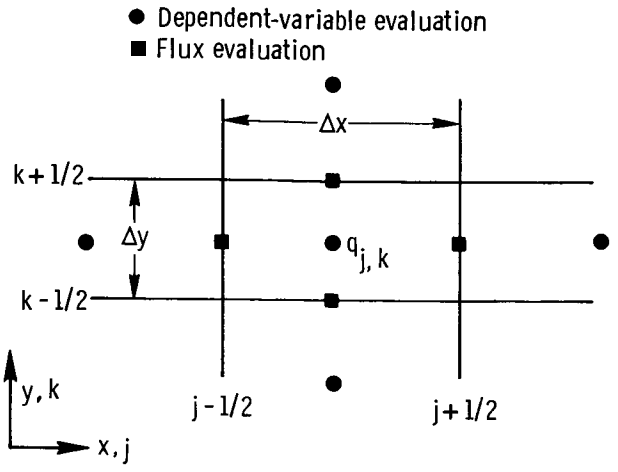
The integral form of the conservation equations governing the flow of an inviscid fluid can be written as

$$\frac{\partial}{\partial t}\int_V q\,dV + \int_S \vec{F}\cdot\hat{n}\,ds = 0 \qquad (1)$$

where V is the volume bounded by the surface S, $\hat{n}$ is the outward-pointing unit normal to the surface, q is the vector of conserved-state variables, and $\vec{F}$ is the flux vector. For simplicity in describing the upwind method, consider the semidiscrete formulation of equation (1) for a uniform Cartesian grid.

Sketch A shows the notation for the semidiscrete formulation. Dividing by the cell volume $(\Delta x, \Delta y)$ and rearranging yields

$$\left(\frac{\partial q}{\partial t}\right)_{j,k} = -\left[\frac{1}{\Delta x}\left(f_{j+1/2,k} - f_{j-1/2,k}\right) + \frac{1}{\Delta y}\left(g_{j,k+1/2} - g_{j,k-1/2}\right)\right] = -R_{j,k} \qquad (2)$$



Sketch A

In equation (2),

$$q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e \end{pmatrix} \quad f = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (e+p)u \end{pmatrix} \quad g = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (e+p)v \end{pmatrix} \qquad (3)$$

where ρ is the density, u and v are the velocity components in the x- and y-directions, respectively, p is the pressure, and e is the total energy per unit volume. The equation set is closed by the ideal gas law.

In this study, upwind differencing is implemented by splitting the convective fluxes, f and g, into positive and negative contributions. The Jacobian matrices associated with the linearization of the positive split-flux contributions have eigenvalues greater than or equal to 0; likewise, the Jacobian matrices of the negative split-flux contributions have eigenvalues less than or equal to 0. The particular flux vector splitting (FVS) technique used here is that of Van Leer. (See ref. 9.) Although other FVS techniques could be implemented in the general framework which follows, Van Leer's was chosen because the individual split-flux contributions vary smoothly across sonic and stagnation points. Also, an important reason for choosing Van Leer's scheme is its simplicity. The splitting of f is given by

$$f(q) = f^+(q) + f^-(q) \qquad (4)$$

where, for $|M_x| < 1$,

$$f^{\pm}(q) = \begin{bmatrix} f^{\pm}_{\text{mass}} \\ f^{\pm}_{\text{mass}}[(\gamma-1)u \pm 2c]/\gamma \\ f^{\pm}_{\text{mass}} \cdot v \\ f^{\pm}_{\text{mass}}\{[(\gamma-1)u \pm 2c]^2/[2(\gamma^2-1)] + v^2/2\} \end{bmatrix}$$

and

$$f^{\bullet}_{\text{mass}} = \pm\rho c\left[\frac{1}{2}\left(M_x \pm 1\right)\right]^2$$

For $M_x \geq 1, f^+ = f$ and $f^- = 0$; for $M_x \leq -1, f^+ = 0$ and $f^- = f$.

In the preceding series of equations, M_x is the local Mach number based on the velocity component in the x-direction u and on the local speed of sound c. The splitting of g in terms of $M_y = v/c$ follows similarly.

Utilizing the flux vector splitting technique, a pair of state vectors are assigned to a common cell interface, and a single numerical flux is derived from this pair. For example, $f_{j+1/2,k}$ can be expressed as

$$\begin{aligned} f(q)_{j+1/2,k} &= f\left(q_L, q_R\right)_{j+1/2,k} \\ &= f^+\left(q_L\right)_{j+1/2,k} + f^-\left(q_R\right)_{j+1/2,k} \end{aligned} \quad (5)$$

where the subscripts L and R imply that the dependent variables at the cell boundary are evaluated from the left and right, respectively. The estimates for the state vector pair at the $(j+1/2, k)$ cell boundary are given by

$$\left(q_L\right)_{j+1/2,k} = q_{j,k} + \frac{\phi}{4}\left[\left(1-\kappa_x\right)\nabla + \left(1+\kappa_x\right)\Delta\right] q_{j,k} \quad (6)$$

$$\left(q_R\right)_{j+1/2,k} = q_{j+1,k} - \frac{\phi}{4}\left[\left(1-\kappa_x\right)\Delta + \left(1+\kappa_x\right)\nabla\right] q_{j+1,k} \quad (7)$$

where

$$\Delta q_{j,k} = q_{j+1,k} - q_{j,k} \quad (8)$$

$$\nabla q_{j,k} = q_{j,k} - q_{j-1,k} \quad (9)$$

Similar formulas hold for the interpolation in the y-direction with parameter κ_y. For $\phi = 0$, the method is first-order accurate in space, and, for $\phi = 1$, the parameters κ_x and κ_y control the spatial accuracy. This accounts for the truncation error of the method. Some common choices are $\kappa = 1/3$, which corresponds to the only third-order accurate member of the family (strictly for a one-dimensional problem) and $\kappa = -1$, the fully upwind second-order scheme. The remaining second-order members are all upwind biased or centered approximations.

To reduce oscillations near discontinuities, a limiter may be incorporated into the scheme by modifying the interpolations given in equations (6) and (7). (See ref. 10 for two examples.) In this study, a monotonicity-preserving limiter was not used, but the effect of limiting was emulated by reducing the second- and third-order interpolations of q on a cell face to first order if the higher order interpolation resulted in a negative value for the square of the sound speed. In the next section, a marching method for supersonic regions is developed based on a particular choice of κ_x.

Relaxation Algorithm

Application of the Euler implicit time-integration scheme in delta form and time linearization to the system of semidiscrete conservation laws given by equation (2) yields

$$\left[\frac{I}{\delta t} + \frac{dR}{dq}\right]\delta q_{j,k} = -R^n_{j,k} \quad (10)$$

where

$$\delta q = q^{n+1} - q^n; \delta t^n = t^{n+1} - t^n; n = \text{Iteration index}$$

In matrix notation, equation (10) can be expressed as

$$N\delta q = -R \quad (11)$$

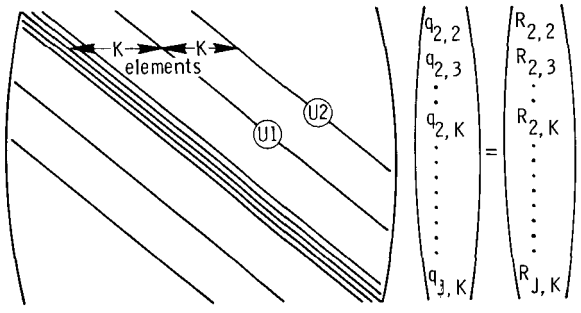
where N is a large, banded, block coefficient matrix with a block size of four. For δt tending to infinity, the solution of equation (11) by direct inversion is Newton's method. In general, ultimately because of the bandwidth of N, the direct solution of equation (11) is not efficient. For a fixed grid, the bandwidth of N depends on the particular values of ϕ and κ used in evaluating the state vectors at a cell interface. Even more important, however, is the fact that for the upwind methods, the bandwidth is also dependent upon the flow regime of the particular problem under consideration. This is in contrast to standard central-differenced schemes, which maintain the same structure of the coefficient matrix independent of the Mach number.

To be specific, equation (10) can be expanded to read

$$\begin{aligned} &\left(\frac{\delta q}{\delta t}\right)_{j,k} + \frac{1}{\Delta x}\Big\{\left[A^+\left(q_L\right)\delta q_L\right. \\ &\qquad \left. + A^-\left(q_R\right)\delta q_R\right]_{j+1/2,k} \\ &\qquad - \left[A^+\left(q_L\right)\delta q_L + A^-\left(q_R\right)\delta q_R\right]_{j-1/2,k}\Big\} \\ &\qquad + \frac{1}{\Delta y}\Big\{\left[B^+\left(q_B\right)\delta q_B + B^-\left(q_A\right)\delta q_A\right]_{j,k+1/2} \\ &\qquad - \left[B^+\left(q_B\right)\delta q_B + B^-\left(q_A\right)\delta q_A\right]_{j,k-1/2}\Big\} = -R_{j,k} \end{aligned} \quad (12a)$$

where

$$\left.\begin{aligned} A^{\pm} &= \frac{\partial f^{\pm}}{\partial q} \\ B^{\pm} &= \frac{\partial g^{\pm}}{\partial q} \end{aligned}\right\} \quad (12b)$$



Sketch B

The subscripts B and A imply the evaluation of q at the cell boundary from below and above, respectively. The structure of N is depicted in sketch B, using equations (6) and (7) and similar formulas for the state-variable interpolation in the y-direction. Any individual element on the nonzero diagonals represents a 4×4 matrix. Vertical-line Gauss-Seidel (VLGS) relaxation sweeping through lines in the positive x-direction would solve the linear system by replacing the coefficients on the upper diagonals, labeled $U1$ and $U2$, with zero. With the ordering shown in sketch B, the elements on the upper diagonals are given by

$$U1 = \frac{1}{\Delta x}\left[A^{+}_{j+1/2,k}\frac{\phi(1+\kappa_x)}{4} + A^{-}_{j+1/2,k}\left(1-\frac{\phi\kappa_x}{2}\right) + A^{-}_{j-1/2,k}\frac{\phi(1-\kappa_x)}{4}\right] \tag{13}$$

$$U2 = -\frac{1}{\Delta x}\left[A^{-}_{j+1/2,k}\frac{\phi(1-\kappa_x)}{4}\right] \tag{14}$$

Supersonic Flow

For a fully supersonic flow in the streamwise (x) direction A^{-} is zero everywhere; thus, $U2$ is zero. However, $U1$ is in general nonzero because of the presence of A^{+}. By choosing $\phi = 0$ (first-order scheme) or $\phi = 1$ and $\kappa_x = -1$ (fully upwind second-order scheme), the elements on $U1$ become zero, and vertical-line Gauss-Seidel becomes a direct solver of the linear problem, resulting in rapid convergence to the steady state. Even for subsonic flows, when the upper diagonals are nonzero, the line relaxation algorithm with alternating sweep directions results in an efficient solution method. (See refs. 7, 8, and 11.)

For supersonic problems, the classical implementation of VLGS or global iteration (solving the system of equations once on each line and then proceeding to the next line) is not the most efficient approach. A superior implementation of VLGS is to start at the first interior line of the grid, next to the inflow boundary, solve the linearized problem, update the dependent variables and residuals, and continue the process until the machine zero steady-state solution on the first line is obtained. The next step is to proceed to the next line and continue until the last line is completed, thus recovering the steady-state solution. This strategy is referred to as local, rather than global, iteration. In this approach, as $\delta t \to \infty$, Newton's method is being implemented by lines, so that the solution along each line converges quadratically. Note that the global strategy attempts to obtain the solution at a downstream field point before a fully converged solution has been obtained upstream. Based on the mathematical theory of characteristics, this is not possible, because the steady solution along the downstream lines depends on the steady state upstream. This dependence manifests itself in the fact that local iteration is more efficient than global iteration for this class of problems, although both converge quadratically.

Transonic Flow

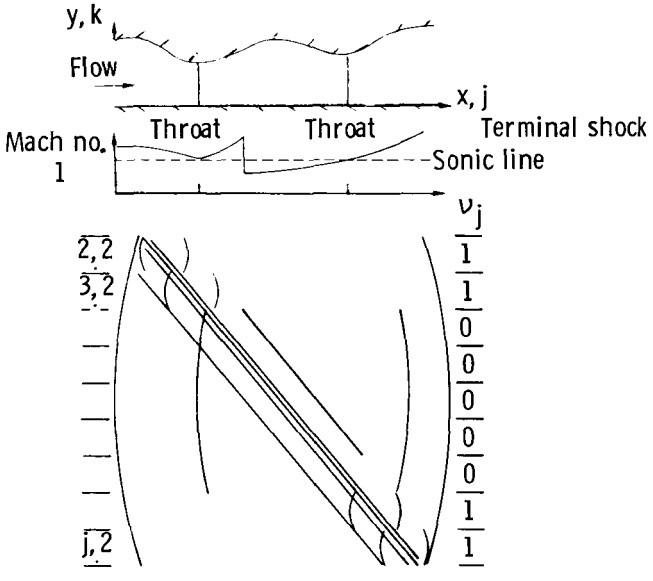
For problems containing separate regions of supersonic and subsonic flow, a hybrid strategy combining local and global iterations can be developed. For this type problem, where large subsonic regions can exist, global iteration may be used with back-and-forth sweeping. A more efficient hybrid strategy can be developed if the steady-state flow-regime interfaces can be located. As an example, consider the dual-throat inlet configuration in sketch C, a possible Mach number distribution for the inlet, and the coefficient matrix that would be obtained from a first-order upwind discretization of the problem, all of which are related to the iteration strategy that follows.

Assume that the steady-state location of the two interfaces separating the supersonic-subsonic states are known. Since the upper diagonal in the coefficient matrix has nonzero elements only on the rows corresponding to the subsonic region and along the preceding supersonic-subsonic interface (but not vice versa), the problem can be solved by local iteration in the supersonic region until a supersonic-subsonic interface is reached. Then, since the beginning and ending rows belonging to the submatrix are known, the large submatrix depicted in figure 3(c) can be solved iteratively until convergence. The last row included is the subsonic line just preceding the supersonic-subsonic cell interface

at the second throat. It is sufficient to close the submatrix with this row, because there is no influence on the solution from the downstream supersonic line. Solving the submatrix in this fashion is referred to as block iteration, because it involves the iterative solution of a group or block of lines in a global fashion. Having obtained the steady-state solution in this region, the remaining supersonic field can be solved by local iteration.

The automatic implementation of the hybrid strategy is very straightforward. All that is needed is a switch that determines whether a line can be solved locally or must be included in a block-iteration group. In sketch C, the parameter ν serves this purpose, and, for the first-order scheme, is defined by

$$\nu_j \begin{cases} 1, & \text{if } M(q_R)_{j+1/2,k} \geq 1 \text{ for all } k \\ 0, & \text{otherwise} \end{cases} \tag{15}$$



Sketch C

If $\nu_j = 1$, the line can be solved locally, otherwise it must be included in a larger submatrix. The value of ν_j at the last line depends on the type of boundary condition specified at the outflow. Obviously, for a supersonic outflow boundary condition, $\nu_j = 1$; for subsonic outflow, $\nu_j = 0$.

The situation is only slightly more complicated with a second-order, fully upwind approximation in the streamwise direction. The only additional requirement for local iteration on line j is that $M(q_R)_{j-1/2,k} \geq 1$; this ensures that the elements on the diagonal $U1$ in sketch B are zero. Therefore, the definition of ν_j for $\phi = 1$ is

$$\nu_j = \begin{cases} 1, & \text{if } M(q_R)_{j+1/2,k} \geq 1 \\ & \quad \text{and } M(q_R)_{j-1/2,k} \geq 1 \text{ for all } k \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

In practice, the steady-state location of the supersonic-subsonic interfaces (and vice versa) are not known a priori, but must be determined as part of the solution process. Two approaches were tried in this study. The first approach was to apply the global iteration technique first until ν_j became fixed in time and then to apply the line-block iteration technique for the final pass through the mesh. A slightly more sophisticated approach is to initially solve the problem on each line to a loose tolerance (e.g., a two- to three-order reduction in the linearization error and/or a fixed number of iterations). After one left-to-right sweep through the mesh, the values of ν_j are computed and the line-block-iteration strategy is applied on the next pass. Subsequent hybrid iterations are made with the line-block strategy until ν_j becomes independent of time. A machine zero tolerance is then set, and the steady-state solution is recovered on the next hybrid sweep through the grid. It is not known under what conditions (tolerance level, problem severity, etc.) this latter strategy will succeed or fail.

Initial Conditions for Local Iteration

To start a problem, a simple set of initial conditions are usually specified (e.g., uniform flow at free-stream conditions everywhere). However, in supersonic regions where local iteration is used, another choice that may reduce the computational effort is to use the solution from the previous line as an initial condition. Although this choice may be an improvement over that of the free-stream conditions, it is generally not a good enough guess to put one inside the domain of attraction of Newton's method without requiring additional iteration. This situation can be at least partially alleviated with only a minor coding effort by using an initial condition generated by solving the steady-state Euler equations approximately. As shown in the section entitled "Results," significant savings are obtained by this approach.

Implementation of this procedure is as follows. Consider the following differential form of the steady-state Euler equations in conservation-law form:

$$\frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0 \tag{17}$$

Let

$$\Delta_j h = h_{j+1} - h_j \qquad (h = f, g, \text{ or } q) \tag{18}$$

where the k subscript has been suppressed for simplicity. Using first-order differencing in the streamwise direction and using equation (17) yields

$$\Delta_j f + \Delta x \frac{\partial \Delta_j g}{\partial y} = -\Delta x \left(\frac{\partial g}{\partial y}\right)_j \tag{19}$$

Linearization of $\Delta_j f$ and $\Delta_j g$ with respect to q yields

$$\left[\left(\frac{\partial f}{\partial q}\right)_j + \Delta x \frac{\partial}{\partial y}\left(\frac{\partial g}{\partial q}\right)_j\right] \Delta_j q = -\Delta x \left(\frac{\partial g}{\partial y}\right)_j \tag{20}$$

In equation (20), the Jacobian matrices, $\frac{\partial f}{\partial q}$ and $\frac{\partial g}{\partial q}$, are already formed (eq. 12(a)); consequently, the implementation and solution of equation (20) is very straightforward. The spatial discretization in the y-direction is upwinded, as previously described. Then,

$$q_{j+1} = q_j + \Delta_j q \tag{21}$$

is set as the initial condition on the $j + 1$ line.

Operation Count, Time Stepping, and Boundary Conditions

Before proceeding to the test problems, a few additional remarks on overall implementation are in order. Frequently, questions arise concerning the computational effort associated with the solution of a system of equations characterized by a block pentadiagonal coefficient matrix as opposed to a block tridiagonal solution process. This is pertinent, since all the discretizations with $\phi = 1$ require a pentadiagonal solver. The operation count, without pivoting, is approximately $n[4m^2 + 2m - 1]$ for a block tridiagonal solver and $n[9m^2 + 3m - 1]$ for a block pentadiagonal solver, where n is the number of unknowns (mesh points) and m is the block size. (See ref. 12.) Thus, approximately 2.18 times more effort is required for that part of the iteration associated with the pentadiagonal solver than with a tridiagonal solver. The actual increase in effort has been measured on a Control Data CY170-855 computer, and a 2.154 ratio was obtained. Solving the linear system, however, represents only 30 percent of the computational time per iteration. The remaining 70 percent is the effort required to set up the linear problem and the boundary conditions. The total increase in CPU time per iteration is therefore only 35 percent. This increase is certainly worth the effort if quadratic convergence, as opposed to linear convergence, can be obtained.

The general time-stepping scheme is a variation of that described in reference 8. At any level of iteration, the time step is obtained from

$$\delta t = \frac{\delta t_o}{\|R^n\|_2} \tag{22}$$

where $\|R^n\|_2$ is the L_2-norm of the residual (the right-hand side of eq. (2)) normalized by the initial residual and δt_o is the initial time step. A common choice for δt_o would correspond to a maximum Courant number of the order of 10. This choice would allow the initial solution to adjust to the enforcement of the steady-state boundary conditions without incurring severe oscillations. As the residual approaches zero, very large time steps are obtained to mimick a steady-state Newton method.

Finally, it has been tacitly assumed that all boundary conditions are treated implicitly, which is required if quadratic convergence is to be obtained. Nonlinear, delta-form boundary conditions can be readily linearized by the techniques already presented and built into the discrete algebraic problem to yield a completely consistent formulation of Newton's method.

Results

To assess the relative performance of the various iteration strategies, the same test problem is solved by different means and CPU times are compared. The first problem considered is the inviscid shock reflection problem (ref. 13), in which a shock wave of prescribed strength reflects off a flat plate. The problem is simulated by prescribing free-stream conditions at the inflow boundary ($M_\infty = 2.9$) and extrapolating conditions at the supersonic outflow boundary (first order). Along the upper boundary, all conditions are fixed through the incident shock angle of 29°, which corresponds to an overspecification of the boundary conditions by setting the exact solution. The boundary conditions on the plate were flow tangency and first-order extrapolation of ρ, u, and e. The grid contained 61 equally spaced points in the x-direction, $0 \le x \le 4.1$, and 21 equally spaced points in the y-direction, $0 \le y \le 1$. Pressure contours for the first-order x and y scheme ($\phi = 0$), the fully upwind second-order x and y scheme ($\phi = 1$, $\kappa_x = -1$, $\kappa_y = -1$), and the hybrid second-order x, third-order y scheme ($\phi = 1$, $\kappa_x = -1$, $\kappa_y = 1/3$) are compared in figure 1. Clearly, a dramatic resolution improvement is obtained with the $\phi = 1$ schemes.

The convergence rates of the three methods using the global iteration procedure are depicted in figure 2. This problem was then solved by lines using the initial conditions given in table 1.

Table 1. Comparison of CPU Time for Shock Reflection Problem[1]
[Tolerance = 10^{-13}; L_2-norm of residual]

		CPU time, sec, for—				
ϕ	κ_y	Global iteration	Uniform flow	Previous line	Eq. (20) ($\phi = 0$)	Eq. (20) ($\phi = 1$)
0		20.6	7.9	7.8	3.8	3.8
1	-1	30.8	(2)	10.6	6.5	(2)
1	1/3	40.8	(2)	11.8	11.9	12.4

[1]Control Data VPS 32 Computer; FORTRAN 2.1.5; OPT = BE compilation; scalar code.
[2]Did not converge because of state vector interpolation near shock reflection.

The results shown in table 1 verify that local iteration is significantly better than global iteration for supersonic flow. Using free-stream initial conditions on a line does lead to difficulty with the higher order methods if discontinuities are present because of the nature of the interpolation. Using the solution from the previous line as the initial guess alleviates this problem, but this method can be further improved by using the first-order-accurate solution of equation (20) as initial conditions. Finally, for the higher order methods, one might expect that using a consistent y-discretization in equation (20) to generate the initial condition would be superior to the first-order discretization, but this has not been the case. The third-order discretization resulted in a marginal CPU increase, and the fully upwind second-order y-discretization resulted in failure. This failure to achieve a further CPU reduction was probably due to the fact that equation (20) is only first-order accurate in x.

The next problem considered is the supersonic and transonic flow in a dual-throat inlet. The nozzle contour is very mild and was generated by specifying a parabolic Mach number distribution between the first throat and the exit and by then solving for the area distribution from the one-dimensional area Mach number relation. Ahead of the first throat the area was constant. A 69×31 mesh was used with equal spacing in the x-direction and slight clustering in the y-direction. The upper-wall contour is sketched in figure 3 along with the converged centerline Mach number distribution from the first-order scheme for both supersonic ($M_\infty = 1.10$) and transonic ($M_\infty = 1.07$) flow. The global convergence histories for these two cases are also shown in figures 4(a) and (b). For illustrative purposes, figure 4(a) has two curves; one has all implicit boundary conditions, and the other has an explicit boundary treatment on the nozzle surface. The difference in the convergence is dramatic and clearly indicates the advantage of fully implicit boundary condition treatment.

The transonic case was solved using three different approaches: (1) global iteration only, (2) global iteration for the first 160 iterations followed by one pass with the hybrid-iteration strategy, and (3) one pass through the mesh assuming a fully supersonic flow followed by two passes with the hybrid-iteration strategy. (The first two passes had a loose (10^{-3}) tolerance, and the final pass had a tight (10^{-12}) tolerance. The results of these three approaches, along with the supersonic results, are summarized in table 2.

Table 2. Comparison of CPU Time for Dual-Throat Problem[1]
[Tolerance = 10^{-12}; L_2-norm of residual]

	CPU time, sec, for—	
Iteration strategy	Supersonic	Transonic
Global	68.9	489
Global/hybrid 160:1		381
Global/hybrid 1:2		151
Local	7.7	

[1]Control Data CYBER 203 time with FORTRAN 2.1.5 cycle 607 compilation.

The results again emphasize the fact that local iteration is significantly more efficient than global iteration for supersonic flows. It is also apparent that major gains for transonic flow can be made using the hybrid-iteration strategy.

Another simple problem that indicates the efficiency of the block-iteration strategy is the Mach reflection problem in an inlet-diffuser. The nozzle geometry and computational details are described in references 14 and 15. A 69×31 grid with slight clustering near the wall and centerline was used. With an inflow Mach number of 1.95, a Mach reflection occurs (see fig. 5). The flow is almost entirely supersonic, with only a small subsonic region existing behind the Mach stem. The global convergence of the first-order scheme is shown in figure 6. The CPU time to obtain a 10^{-12} residual was 48.9 sec, and, with the three-pass block-iteration strategy previously described, the CPU time was reduced to 17.8 sec. This result, however, was obtained using the solution from

the previous line as the initial condition, so a further improvement could likely be realized.

A more severe test of the solution strategy is described in this case, where the solution of the Euler equations in a highly contoured dual-throat inlet is sought. The nozzle contour and 69 × 31 grid are shown in figure 7. The wall is made up of five polynomial arcs. Wall slope is continuous at the arc interfaces, and curvature is discontinuous at two of the interfaces.

The flow entering the nozzle is subsonic and originates from an infinite reservoir, which is at rest. At the inflow boundary, two conditions are obtained from evaluation of locally one-dimensional Riemann invariants (see ref. 10), which are

$$R^{\pm} = V_n \pm \frac{2c}{\gamma - 1} \tag{23}$$

The velocity component V_n is the component of velocity normal to the boundary and is defined to be positive in the direction pointing outward from and normal to the boundary. For subsonic flow, R^- can be prescribed on the boundary and R^+ can be evaluated from the interior. The remaining two conditions that were prescribed at the inflow boundary were the entropy (p/ρ^γ), which was held constant at stagnation conditions, and the y velocity component v, which was set to zero based on a parallel-flow assumption at the entrance of the nozzle. The conditions for the wall and centerline were flow tangency and extrapolation of density, temperature, and total enthalpy. The flow at the exit station is fully supersonic; thus, all quantities were extrapolated there.

The initial conditions used to start the calculation were relatively simple. The Mach number/area relation (assuming quasi-one-dimensional isentropic flow and sonic conditions at the first throat) was used to obtain the initial entrance conditions. Similarly, initial conditions at the exit station were determined based on the exit-to-first-throat area ratio flow. The Mach number, density, and static pressure were then linearly interpolated into the x-direction. The individual velocity components were chosen such that the initial flow direction was parallel to the grid lines in the streamwise direction.

Mach number contours are shown in figure 8. The Mach number at the entrance to the nozzle converged to $M = 0.09$ (the Mach number area relations used $M \approx 0.1$). The flow accelerates to sonic conditions at the first throat, continues to accelerate downstream of the first throat, and a Mach reflection forms in the converging region ahead of the second throat. Just ahead of the second throat and immediately behind the Mach stem, a small subsonic region exists. The flow in this region accelerates to sonic conditions at the second throat, beyond which the flow becomes entirely supersonic. At the exit station, the Mach number varied from 1.96 on the centerline to 4.39 on the surface. Quasi-one-dimensional theory initialized the exit Mach number to 3.22.

The convergence histories of the first-order and fully upwind second-order schemes using global iteration are shown in figure 9. The sawtooth pattern is typical for the higher order upwind methods with alternate line Gauss-Seidel sweeping. In the calculation, the maximum time step was limited and resulted in limiting the maximum Courant number to 405. The results obtained with three hybrid-iteration sweeps are presented in table 3.

Table 3. Comparison of CPU Time for Global and Hybrid Iterations

Difference method parameters		CPU time, sec, for—	
ϕ	κ_y	Global iteration	Hybrid iteration
0	N/A	193	Not calculated
1	-1	257	168
1	1/3	259	171

The dual-throat problem contains two separate subsonic regions which together span 55 percent of the x = Constant grid lines. An approximation for the optimal performance improvement that could be obtained from the hybrid-iteration strategy would therefore be 45 percent, since this could be achieved only if the supersonic-subsonic interfaces were known a priori and if the CPU time associated with the supersonic regions were neglected. The results in table 3 show that a 35-percent reduction in CPU time was actually obtained. Thus, for this problem, the three-pass hybrid-iteration strategy attains approximately 78 percent of the optimal performance improvement that one could expect with this technique.

Concluding Remarks

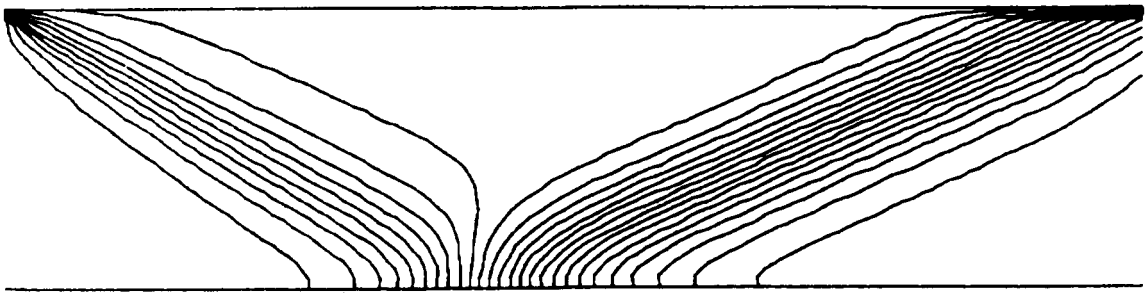
The family of algorithms presented in this study results in an efficient iterative technique for inviscid flow problems, particularly in the supersonic regime. The new iteration approach for problems containing separate regions of supersonic and subsonic flow also represents a significant savings in computational

effort. The optimal approach to implementing this strategy is still unknown.

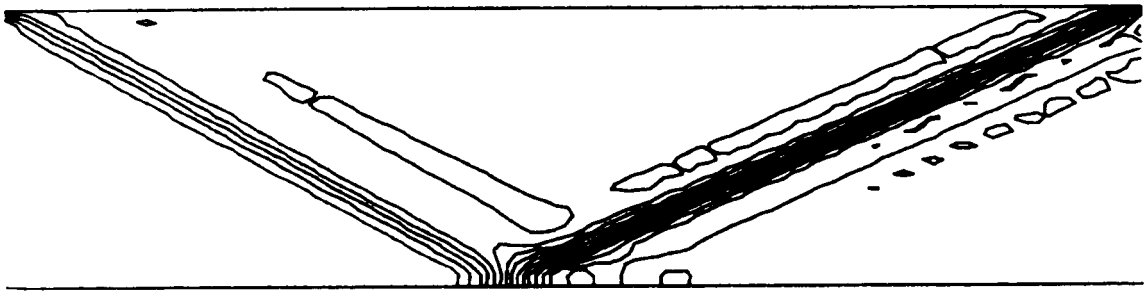
NASA Langley Research Center
Hampton, VA 23665-5225
September 19, 1986

References

1. Briley, W. R.; and McDonald, H.: Solution of the Multidimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method. *J. Comput. Phys.*, vol. 24, no. 4, Aug. 1977, pp. 372–397.
2. Beam, Richard M.; and Warming, R. F.: An Implicit Factored Scheme for the Compressible Navier-Stokes Equations. *AIAA J.*, vol. 16, no. 4, Apr. 1978, pp. 393–402.
3. Jameson, Antony; and Bakes, Timothy J.: Multigrid Solution of the Euler Equations for Aircraft Configurations. AIAA-84-0093, Jan. 1984.
4. MacCormack, Robert W.: The Effect of Viscosity in Hypervelocity Impact Cratering. AIAA Paper No. 69-354, Apr.–May 1969.
5. Abarbanel, Saul S.; Dwoyer, Douglas L.; and Gottlieb, David: Improving the Convergence Rate of Parabolic ADI Methods. *A Collection of Technical Papers—AIAA 6th Computational Fluid Dynamics Conference,* July 1983, pp. 99–109. (Available as AIAA-83-1897.)
6. Abarbanel, Saul S.; Dwoyer, Douglas L.; and Gottlieb, David: *Stable Implicit Finite-Difference Methods for Three-Dimensional Hyperbolic Systems.* ICASE Rep. No. 82–39, 1982.
7. Chakravarthy, Sukumar R.: Relaxation Methods for Unfactored Implicit Upwind Schemes. AIAA-84-0165, Jan. 1984.
8. Van Leer, Bram; and Mulder, William A.: *Relaxation Methods for Hyperbolic Equations.* Rep. 84-20, Dep. Math. & Inform., Delft Univ. of Technol., 1984.
9. Van Leer, Bram: Flux-Vector Splitting for the Euler Equations. *Eighth International Conference on Numerical Methods in Fluid Dynamics, Volume 170 of Lecture Notes in Physics,* E. Krause, ed., Springer-Verlag, 1982, pp. 507–512.
10. Anderson, W. Kyle; Thomas, James L.; and Van Leer, Bram: A Comparison of Finite Volume Flux Vector Splittings for the Euler Equations. AIAA-85-0122, Jan. 1985.
11. Lombard, C. K.; Venkatapathy, E.; and Bardina, J.: Universal Single Level Implicit Algorithm for Gasdynamics. AIAA-84-1533, June 1984.
12. Dahlquist, Germund; and Björck, Åke (Ned Anderson, transl.): *Numerical Methods.* Prentice-Hall, Inc., c.1974.
13. Yee, H. C.; Warming, R. F.; and Harten, A.: Implicit Total Variation Diminishing (TVD) Schemes for Steady-State Calculations. *A Collection of Technical Papers—AIAA 6th Computational Fluid Dynamics Conference,* July 1983, pp. 110–127. (Available as AIAA-83-1902.)
14. Bogar, T. J.; Sajben, M.; and Kroutil, J. C.: Characteristic Frequency and Length Scales in Transonic Diffuser Flow Oscillations. AIAA-81-1291, June 1981.
15. Walters, Robert W.: LU Methods for the Compressible Navier-Stokes Equations. Ph.D. Thesis, North Carolina State Univ., 1984.



(a) First-order x and y.



(b) Second-order x and y.

(c) Second-order x, third-order y.

Figure 1. Pressure contours for shock reflection problem.

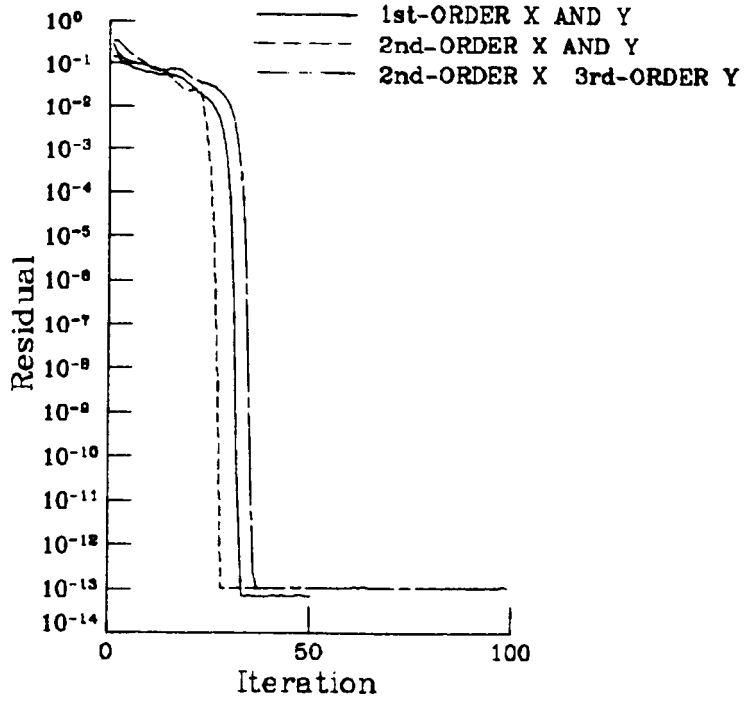


Figure 2. Convergence histories for shock reflection problem.

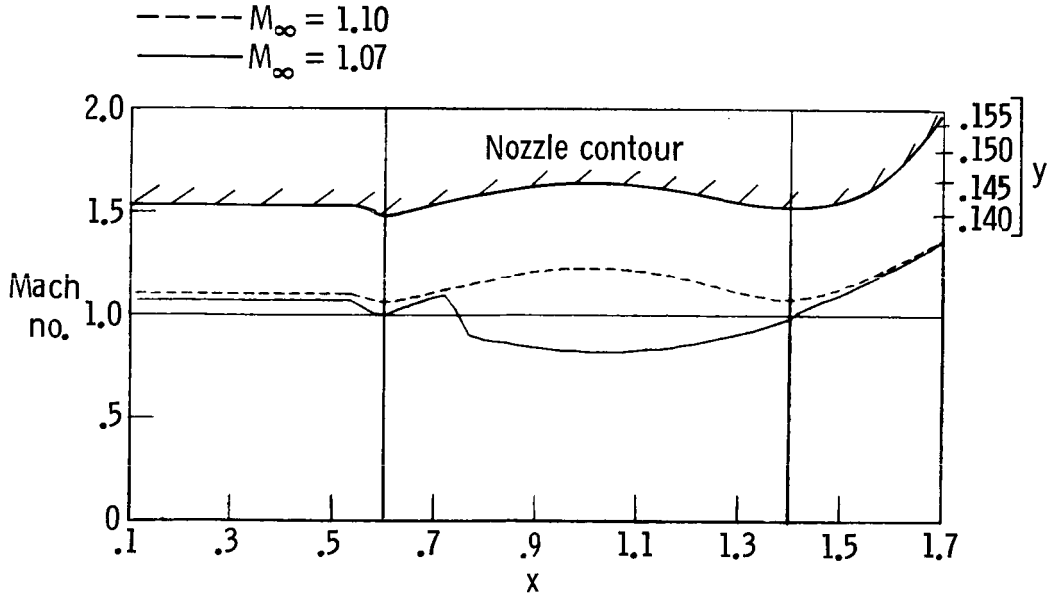
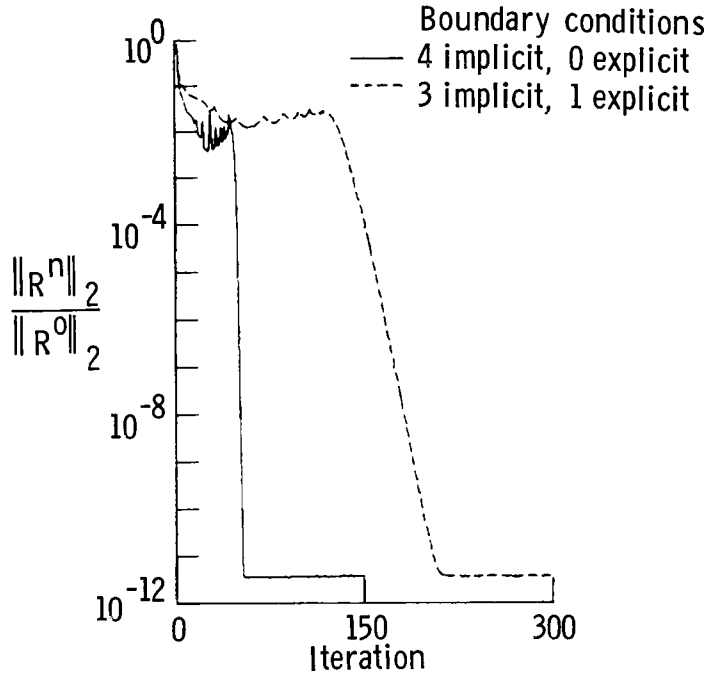
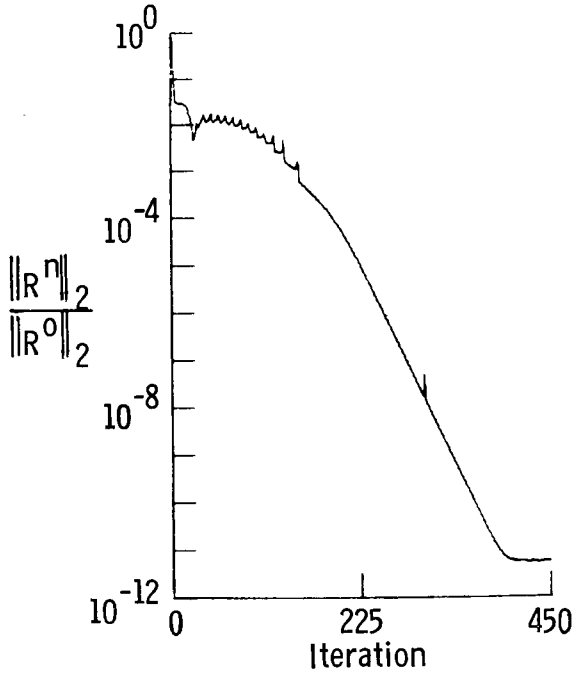


Figure 3. Centerline Mach number distribution and wall contour of mildly contoured dual-throat inlet.



(a) Supersonic; $M = 1.10$; 69×31 mesh.



(b) Transonic; $M = 1.07$; 69×31 mesh.

Figure 4. Convergence histories for dual-throat problem.

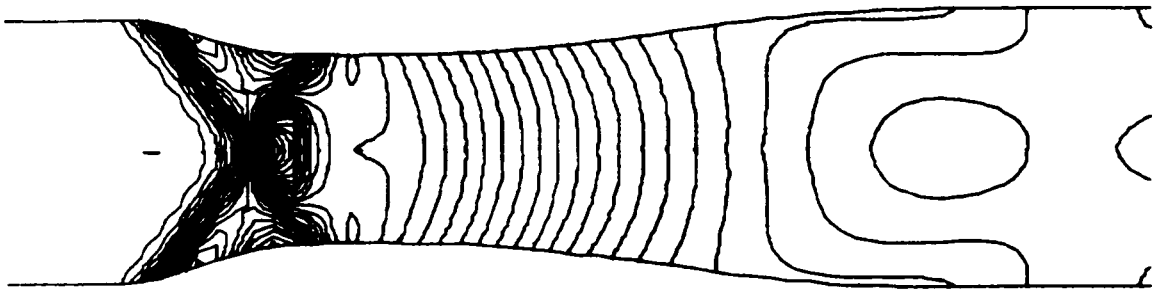


Figure 5. Pressure contours for Mach reflection problem.

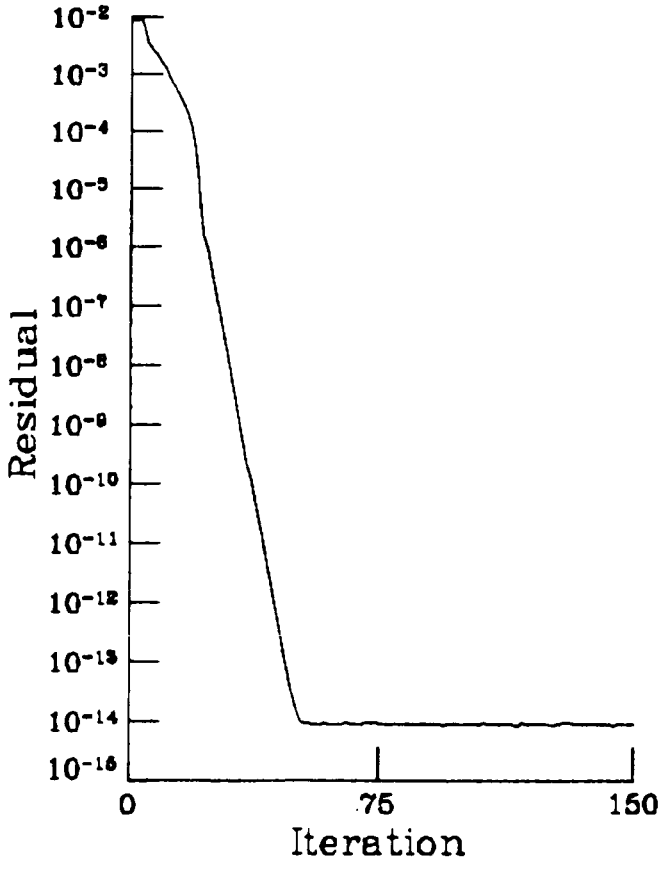


Figure 6. Convergence history for first-order scheme of Mach reflection problem.

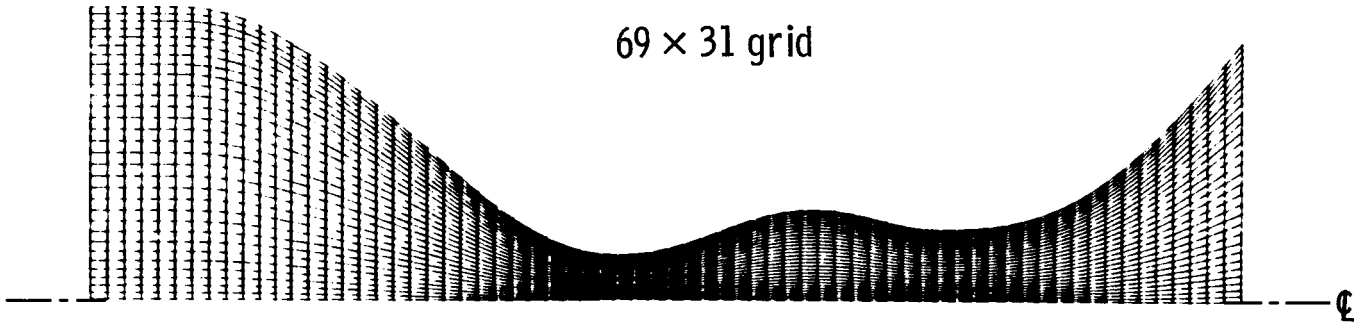


Figure 7. Mesh used for highly contoured dual-throat inlet problem.

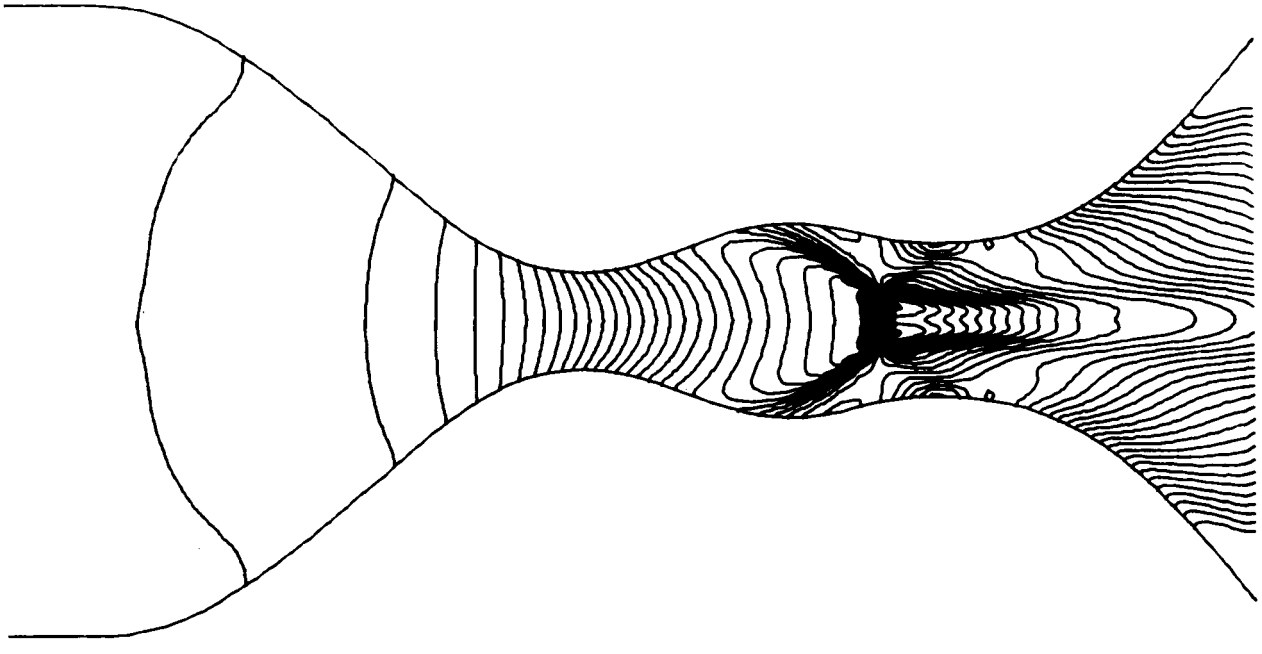


Figure 8. Mach number contours.

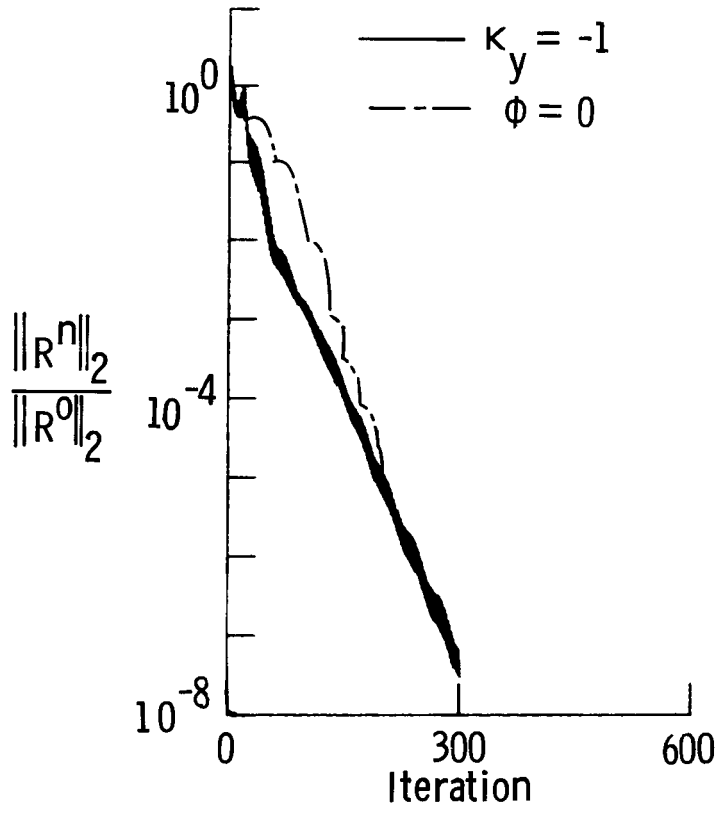


Figure 9. First- and second-order convergence histories for highly contoured inlet problem.

Standard Bibliographic Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TP-2523		

4. Title and Subtitle	5. Report Date
Efficient Solutions to the Euler Equations for Supersonic Flow With Embedded Subsonic Regions	January 1987
	6. Performing Organization Code: 505-31-03-02
7. Author(s): Robert W. Walters and Douglas L. Dwoyer	8. Performing Organization Report No.: L-15975
9. Performing Organization Name and Address: NASA Langley Research Center, Hampton, VA 23665-5225	10. Work Unit No.
	11. Contract or Grant No.
12. Sponsoring Agency Name and Address: National Aeronautics and Space Administration, Washington, DC 20546-0001	13. Type of Report and Period Covered: Technical Paper
	14. Sponsoring Agency Code

15. Supplementary Notes

Robert W. Walters: NRC-NASA resident research associate, now at Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

16. Abstract

A line Gauss-Seidel (LGS) relaxation algorithm in conjunction with a one-parameter family of upwind discretizations of the Euler equations in two dimensions is described. Convergence of the basic algorithm to the steady state is quadratic for fully supersonic flows and is linear for other flows. This is in contrast to the block alternating direction implicit methods (either central or upwind differenced) and the upwind biased relaxation schemes, all of which converge linearly, independent of the flow regime. Moreover, the algorithm presented herein is easily coupled with methods to detect regions of subsonic flow embedded in supersonic flow. This allows marching by lines in the supersonic regions, converging each line quadratically, and iterating in the subsonic regions, and yields a very efficient iteration strategy. Numerical results are presented for two-dimensional supersonic and transonic flows containing oblique and normal shock waves which confirm the efficiency of the iteration strategy.

17. Key Words (Suggested by Authors(s))	18. Distribution Statement
Upwind differencing Newton iteration Space marching	Unclassified—Unlimited Subject Category 02

19. Security Classif.(of this report)	20. Security Classif.(of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	16	A02

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

NASA-Langley, 1987