A COMPUTATIONAL FLUID DYNAMICAL CODE for the INVESTIGATION of RAM ACCELERATOR CONCEPTS

by

DEAN CARTER BRACKETT

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

University of Washington

1986

Approved by _____

Program Authorized

to Offer Degree Department of Aeronautics and Astronautics

Date August 8, 1986

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

Roman letters

A	area, Jacobian Matrix
c	speed of sound
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy number
CFS	Correction flux splitting technique
CFS1	first correction flux splitting technique
CFS2	second correction flux splitting technique
e	internal energy
e_t	total energy
EOS	equation of state
F	x-direction flux vector
G	y-direction flux vector
GOD1	first order Godunov technique
K	proportionality constant
m_i	mass fraction
n	timestep value
N	total number of computational cells
p	pressure
R_p	pressure ratio
RES	density solution residue
t	time
T	transformation matrix
u	x-direction velocity

U	state vector
v	y-direction velocity
V	computational cell volume
x	axial position
y	radial position

Greek letters

γ	ratio of specific heats
ρ	density
Λ	eigenvalue matrix
ζ	extrapolation/interpolation weighting factor

Subscripts and Superscripts

b	conditions at the cell boundary
c	corrector step
l	conditions to the left of the cell boundary
n	conditions at the nth timestep
p	predictor step
r	conditions to the right of the cell boundary
$+$	information traveling from the left
$-$	information traveling from the right

ACKNOWLEDGEMENTS

The author wishes to express sincere gratitude to Dr. David W. Bogdanoff, principal author of the computer code discussed in this thesis, for generously contributing his knowledge, experience, and many long hours. In addition, special thanks to Professors Abraham Hertzberg and Adam P. Bruckner for their great confidence in my abilities, and for providing the opportunity to be a part of the ram accelerator project. Thanks also to all the students associated with the project who made working in the lab a pleasure.

CHAPTER I

INTRODUCTION

Work on a high velocity device called the ram accelerator is in progress at the University of Washington. This device is capable, in principle, of accelerating payload masses up to metric tons to velocities of 8 to 14 km/sec and smaller masses to velocities up to about 25 km/sec by chemical means. In the lower speed range of operation (1 to 14 km/sec) the payload/projectile is propelled through a tube containing combustible gas mixtures. The projectile, analogous to a ramjet centerbody, experiences a thrust as combustion and/or detonation occurs in the surrounding flow. The various modes of device operation, which may be staged to produce ultrahigh velocities, are described in Reference 1. One such mode, involving an oblique detonation wave, is illustrated schematically in Figure 1. The projectile is injected supersonically into a tube containing premixed combustible gas. A conical shock wave develops on the projectile nosecone. Flow parameters such as pressure and composition can be set so that the reflected shock wave initiates combustion. The projectile travels at a velocity greater than the detonation speed of the fuel mixture. A detonation wave forms at the reflected shock front where combustion occurs in a thin layer. The heated gas expands through the nozzle formed by the projectile rear and tube wall, providing thrust. In this mode of operation, the ram accelerator can accelerate projectiles to velocities up to 14 km/sec. A research version of the ram accelerator, operating in the lowest velocity range (1 to 2.7 km/sec) ramjet mode, has been constructed at the University of Washington and is presently in operation.

Each mode of ram accelerator operation involves complex flows with energy release by the combustion and/or detonation of gas, or, in the higher velocity modes, the detonation of high explosives, involving dense media interaction. The need to

demonstrate the viability of advanced ram accelerator concepts has motivated the development of a computational fluid dynamical (CFD) computer program. This program, hereafter referred to as the "code" is the subject of this thesis. The code is written in the FORTRAN computer language and is presently run on a VAX 11/750 computer.

The code numerically solves the two-dimensional Euler equations using an explicit predictor-corrector time differencing scheme.[2] This code differs from most Euler solving codes in that it can successfully model ultrahigh velocity flows and it can use a completely general equation of state. Also, simple chemical reactions can be included. A new correction flux splitting (CFS) technique, capable of high velocity flow calculation, is available in the code. The two-dimensional version of the code can be run in either planar or axisymmetric geometries. A one-dimensional version is available to study high velocity plate slap problems. Viscous effects are not included in the code as yet. They will, however, be included at a later point.

Design features of the code and the numerical methods employed are described in Chapter II. Code results for special cases with analytic solutions are presented for comparison in Chapter III. The success of these preliminary trials provides a promising outlook for the application of the code to the ram accelerator problem. Some features which may be added to the code at a later date are described in Chapter IV.

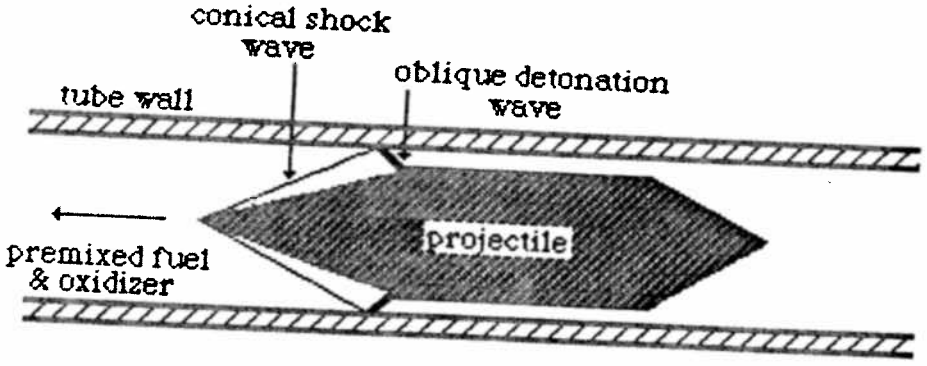


Fig.1 Oblique detonation ram accelerator

CHAPTER II

CODE DEVELOPMENT

A. Design Features

The code has been developed primarily as a tool for analyzing the flow interactions in various ram accelerator configurations under study. However, maximum flexibility is retained for ease in studying new or altered problems as project needs dictate. The major code features include:

1. Multiple Component Media

The working fluid may be composed of multiple arbitrary chemical species to model mixtures such as fuel/oxidizer/ diluent.

2. Combustion

Energy release in a combustible mixture is modeled using a simple global Arrhenius expression for the chemical reaction rates. Each computational timestep consists of an Euler solver step with frozen chemistry followed by a chemical reaction step in isolated cells without flow interaction. Each of these steps is predictor-corrector (second order accurate in time).

3. Multiple Zones

In the two-dimensional version of the code, both axisymmetric and planar problems can be analyzed. The domain may be divided into multiple zones with the necessary boundary conditions at each interface. This permits modeling of radial variations in gas mixtures or the inclusion of an explosive-driven liner, for example. The computational grid slides in the radial (y) direction as necessary to satisfy the prescribed boundary condition and preserve zone material integrity.

4. General Equation of State (EOS)

Consistent with the desire to analyze dense media as well as gaseous interactions, an arbitrary EOS may be employed in each computational zone. The use

of a correction flux splitting or Godunov technique in calculation of cell boundary fluxes (Sec B.2) lifts the restriction that the EOS be of ideal gas form.

B. Numerics

1. Governing Equations

The two-dimensional Euler equations, written in conservation form are:

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \qquad [1]$$

where U is the state vector

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e_t \\ \rho m_1 \\ \rho m_2 \\ \vdots \end{pmatrix}. \qquad [2]$$

and the flux vectors F and G in the x and y-directions are given by

$$F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e_t + p) \\ \rho u m_1 \\ \rho u m_2 \\ \vdots \end{pmatrix} \qquad G = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ v(e_t + p) \\ \rho v m_1 \\ \rho v m_2 \\ \vdots \end{pmatrix}. \qquad [3]$$

The code employs the finite volume calculation technique. The state variable U is calculated at the center of each computational cell characterized by density, ρ, total energy per unit volume, e_t, axial velocity, u, and radial velocity, v. ($e_t = \rho\left(e + \frac{u^2}{2} + \frac{v^2}{2}\right)$ where e is the internal energy.) Local speed of sound, c, temperature, and pressure, p, are obtained from the equation of state, and component mass fractions, m_i, are also calculated.

The flux vectors, evaluated at cell boundaries, are determined by the values of the primitive variables (ρ,u,v,e,p,m_i) on either side of the cell wall. A third

order extrapolation/interpolation of ρ, u, v, e, and m_i is used to obtain cell wall values. p is then obtained from the EOS. If, however, the extrapolated/interpolated cell wall boundary values for each variable are not between the two adjacent first order values, the corresponding first order value is chosen. This technique, which forces the code to revert to first order calculations in regions of strong curvature, avoids the instabilities which would occur if third (or second) order calculations were used throughout the flow field. Cell wall value calculations are illustrated in Figure 2. The final calculated cell wall quantities are weighted averages of third order interpolated and extrapolated values. The weighting factor can be "tuned" to enhance solutions by reducing numerical dissipation, provided that solution stability (Sec B.4) is retained.

2. Cell Wall Flux Calculations

The flux vectors, F and G, are calculated from the state vector U obtained at either side of the cell wall. Three methods are available. The code is written to employ these techniques, as options, discussed below. Test case comparisons were conducted to determine the optimum method.

Two correction flux splitting techniques and a first order Godunov method are available in the code. All observe the proper characteristic directions of information flow. In the following discussion, for clarity, these techniques are illustrated referring to the one-dimensional one component case.

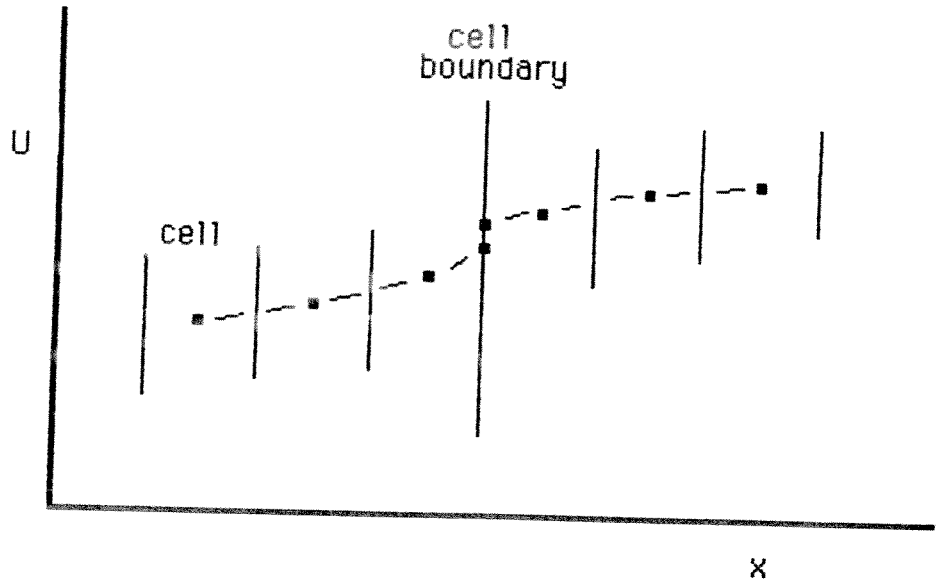
The Euler equations, if homogeneous of degree one, have the property[2] that

$$F = \frac{\partial F}{\partial U} U = AU. \qquad [4]$$

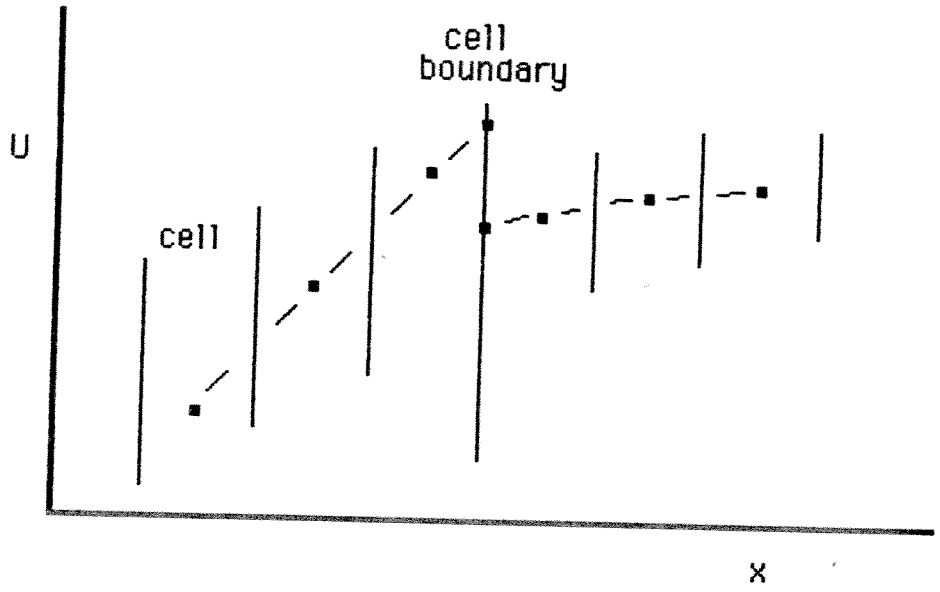
The Jacobian matrix, A, may be diagonalized by the similarity transformation

$$A = T\Lambda T^{-1} \qquad [5]$$

where Λ is a diagonal matrix containing the eigenvalues of A which are u, $u + c$, and $u - c$. Hence, the flux vector F can be "split" into two matrices, one containing information traveling from the left, denoted by the superscript $+$, and the other



(a) third order extrapolation to the cell boundary is satisfactory in this case.



(b) reversion to first order calculation is required in this case.

Fig. 2: Cell wall value calculations

containing information traveling from the right, denoted by the superscript $-$.

$$F^+ = A^+ U_l \qquad F^- = A^- U_r \qquad [6]$$

where subscripts l and r imply evaluation at the left and right sides of the cell wall, respectively. The matrix A may be evaluated alternately at the left side, A_l, or right side, A_r, for predictor-corrector methods, or at averaged conditions.

The standard flux split technique sets the cell boundary flux, F_b, to be the sum of the left and right fluxes.

$$F_b = F^+ + F^- = A^+ U_l + A^- U_r \qquad [7]$$

Although this method preserves the proper characteristic flow of information, it is not suitable for this code. The condition that the Euler equations be homogeneous of degree one requires an equation of state of the form $p = K\rho e$, precluding general EOS flow analysis.

However, it has been found that a correction, usually small, applied to the standard technique yields accurate results. This correction flux splitting (CFS) technique is

$$F_b = F_l + A_r^- (U_r - U_l) \qquad [8]$$

or

$$F_b = F_r + A_l^+ (U_l - U_r) \qquad [9]$$

where F_l and F_r are calculated directly from the left and right hand side primitive variables.

The code uses two versions of this correction flux splitting technique with the matrices A, A^+, and A^- calculated for a general equation of state. The first method (CFS1) calculates p, ρ, u, and e as for a one-dimensional problem. Velocity v at the cell boundary is set equal to v_l or v_r according to the direction of the u characteristic. Mass fractions for a multi-component region are treated in the same manner as the

v velocity. The second method (CFS2) includes v in the matrix calculations and mass fractions are treated as in the first method.

The third method available for flux calculation is the first order Godunov technique[3] (GOD1). It is not a flux splitting technique and does not require the Euler equations to be homogeneous of degree one. Basically, this method consists of solving the Riemann Problem at each cell boundary to first order accuracy. The equations[4] associated with each of the three characteristic directions are

$$\begin{aligned} dp &= \rho c\,du \qquad && (u + c \text{ characteristic}) \\ dp &= -\rho c\,du \qquad && (u - c \text{ characteristic}) \\ de &= p\frac{d\rho}{\rho^2} \qquad && (u \text{ characteristic}). \end{aligned} \tag{10}$$

These equations taken with the equation of state $p = p(\rho, e)$ can be solved for the four cell wall quantities p, ρ, u, and e directly. For example, in the case of an ideal gas, with a rightward moving shock wave and a leftward moving expansion zone, these equations become

$$\begin{aligned} p - p_l &= -\rho_l c_l (u - u_l) \\ p - p_r &= \rho_r c_r (u - u_r) \\ e - e_l &= p_l \left(\frac{\rho - \rho_l}{\rho_l^2} \right) \\ p &= (\gamma - 1)\rho e. \end{aligned} \tag{11}$$

Second order or even exact Riemann solutions can be obtained, but are not used in the present code. Velocity v and component mass fractions are calculated as in the first CFS techique.

3. Calculation in Time

The unsteady Euler equations are hyperbolic in nature for all flow regimes. The code solves them numerically using an explicit finite difference scheme marching in time. This approach is chosen as the calculations are more straightforward

than those associated with implicit schemes which require the "inversion" of a pentadiagonal matrix in the two-dimensional case. Also, since this code is intended to study a considerable number of unsteady flow development problems, the convergence time advantage of implicit techniques in obtaining steady state solutions is relatively unimportant.

The code employs the MacCormack predictor-corrector differencing scheme[2] which is second order accurate in time. The numerical solution to equation [1] is advanced in time from $n\Delta t$ to $(n+1)\Delta t$ in each cell as follows:

Predictor step:

$$U^{\overline{n+1}} V^{\overline{n+1}} = U^n V^n + \sum F_p A \Delta t \qquad [12]$$

Corrector step:

$$U^{n+1} V^{n+1} = \frac{1}{2}\left(U^n V^n + U^{\overline{n+1}} V^{\overline{n+1}} + \sum F_c A \Delta t\right) \qquad [13]$$

where U is the state vector, V is cell volume, A is cell wall area, fluxes F_p and F_c are given by equations [8] and [9] when CFS1 or CFS2 are used or directly by equation [2] when GOD1 is used. When CFS1 or CFS2 is used, in odd timesteps, equation [8] is used for the predictor and equation [9] is used for the corrector. In even timesteps, the use of equations [8] and [9] is reversed. This procedure avoids the generation of a preferred predictor-to-corrector direction and is found to improve the quality of the solutions. It should be noted that when the computational grid slides, $V^n \neq V^{\overline{n+1}} \neq V^{n+1}$ and the areas must be carefully averaged values, otherwise spurious source terms can be created. Barred quantities represent "predicted" values. Only fluxes in the x-direction have been included for clarity. The forward and backward difference operators normally contained in the predictor-corrector method are absent here since the fluxes, F_p and F_c are evaluated at the cell wall as described in the previous section. The third order extrapolation/interpolation used in the code increases the spatial accuracy of the code beyond that of standard first

or second order predictor-corrector methods, and greatly reduces the production of spurious numerical dissipation.

4. Stability

Explicit numerical solution of the Euler equations requires restriction of the length of the computational timestep to satisfy the von Neumann necessary condition for stability.[5] This condition states that the Courant-Friedrichs-Lewy (CFL) number be at most equal to unity, which for this code becomes $(|u| + c)\,\Delta t/\Delta x \leq 1$ and $(|v| + c)\,\Delta t/\Delta y \leq 1$ where $|u| + c$ and $|v| + c$ represent the largest eigenvalues of the Jacobian matrices $\partial F/\partial U$ and $\partial G/\partial U$.

Limiting timestep length, Δt, to satisfy these conditions ensures that solution errors do not grow in time without bound. This CFL condition applies to many hyperbolic systems of equations as it requires that the analytic domain of influence lie within the numeric domain of influence. In this way, the computational fluid model properly reflects the flow of information in physical space.

Reduction of the CFL number much below unity may degrade solution accuracy as the numeric domain of influence includes more unnecessary information. The test cases presented in Chapter III were run with a conservative value of CFL=0.6 .

5. Convergence

Minimizing the computation time required to produce converged steady state solutions is critical in constructing a useful computational code. Several techniques have been used in the code to shorten convergence times.

Convergence is determined by the behavior over time of the density solution residue given by

$$\mathrm{RES} = \left\{ \frac{1}{\mathrm{N}} \sum_{\mathrm{N\ cells}} \left(\frac{2(\rho^{n+1} - \rho^{n})}{(\rho^{n+1} + \rho^{n})} \right)^2 \right\}^{\frac{1}{2}} . \qquad [14]$$

The residue is calculated at each timestep. Typical plots of residue behavior are presented in Chapter III. As the solution converges, the residue decreases. The nature of the MacCormack predictor-corrector method is such that rather than

falling to machine zero, the residue oscillates about some finite value.[2] This provides an estimate of the discretization error inherent in the finite volume modeling of the flow field.

One method used in the code to accelerate solution convergence is the ability to initiate calculation on a coarse grid containing relatively large cell volumes. Requiring fewer calculations, the coarse grid solution converges rapidly. The finite volumes are then divided into smaller elements (by a factor of 2 or 3 in each dimension) which retain the converged solution of the parent cell. Calculation resumes in the refined grid requiring less time to converge than if started from initial conditions. The cell splitting process may be repeated to increase code speed.

Another convergence acceleration technique employed concerns initial flow velocity. Inlet axial velocities of many kilometers per second impacting the projectile profile cause large transient numerical disturbances in the surrounding grid, which increase the convergence time. To avoid this unnecessary time expenditure, the initial velocity direction is distributed across the entire computational region to be everywhere parallel to the grid which conforms to the projectile shape and tube wall. For some projectile shapes, use of this technique is found to be necessary to prevent code failure at high velocities due to void formation as the flow expands at the projectile rear.

6. Calculation Near Flow Boundaries

Extrapolation of interior flow variables to solid boundaries and zone interfaces has proven to be a most important factor affecting solution quality. Much effort has been spent investigating various techniques for boundary calculations. Only those producing high quality results are discussed here.

One of the following boundary conditions is imposed at each zone boundary:

i. Normal velocity specified
(e.g. zero normal velocity at a solid boundary)

ii. Normal velocity and pressure matched across zone boundary
(e.g. at a fluid interface)

iii. Pressure specified

(e.g. zero pressure at a free surface)

For boundary condition (i), second order extrapolations of pressure and normal velocity are made to the boundary, and a first order Godunov technique (Riemann solver) is used to obtain the pressure at the boundary. First order extrapolations to the boundary produce significant errors (about 10 to 15%) consistently and third order extrapolations tend to produce erratic results. For boundary condition (ii), third order extrapolation/interpolation of pressure and normal velocity are made using reduction to first order in regions of strong curvature in a way similar to that used for the internal cell boundaries. A first order Godunov technique is then used to obtain the boundary pressure and velocity. For boundary condition (iii), the technique used is of the same type used for boundary condition (i). Second order Godunov solvers were tested for boundary condition (i), but do not offer any significant improvement over first order techniques.

For the internal cell boundary one cell removed from the zone boundary, two additional "virtual" cells are created by "reflecting" the two cells nearest the zone boundary about the boundary. Using the additional "virtual" cells, second or third order extrapolations/interpolations can be made to obtain the fluxes at this particular type of cell boundary. For these conditions, second order calculations are found to give much more consistent results than third order.

CHAPTER III

TEST CASES

Several test cases have been conducted to qualify the code. Test cases having analytical solutions were chosen to permit direct comparison with code solutions. Each case provides opportunities to evaluate a number of code features. These test cases involve ideal gas flow with a specific heat ratio, γ, of 1.4 . A single computational zone is used. Test case results are presented graphically as overlays of corresponding code and analytic solutions of the state variables ρ, u, v, e, and p. These results demonstrate the most promising techniques incorporated in the code to date.

A. Riemann Shock Tube Problem

Riemann's Problem consists of describing the one-dimensional flow field produced in a shock tube. Flow is initiated by the instantaneous rupture of a diaphragm separating high and low pressure ideal gas volumes initially at the same temperature. As the flow develops, a shock wave moves into the low pressure region while a centered expansion fan moves into the high pressure region. A moving contact surface separates the gas volumes. The flow configuration associated with Riemann's Problem is shown in Figure 3. Equations relating the properties of the resulting flow regions are given in Reference 6.

Comparison of code solutions with the analytic shock tube solutions provides an excellent test of the accuracy of the code. The ability of the code to model important flow phenomena such as normal shocks, expansion waves, and contact surfaces is also evident from this comparison. These tests also provide the necessary feedback for "tuning" the code to ensure solutions of the highest quality are obtained when applied to more complex higher dimensional flow problems. Figures 4 to 9 illustrate

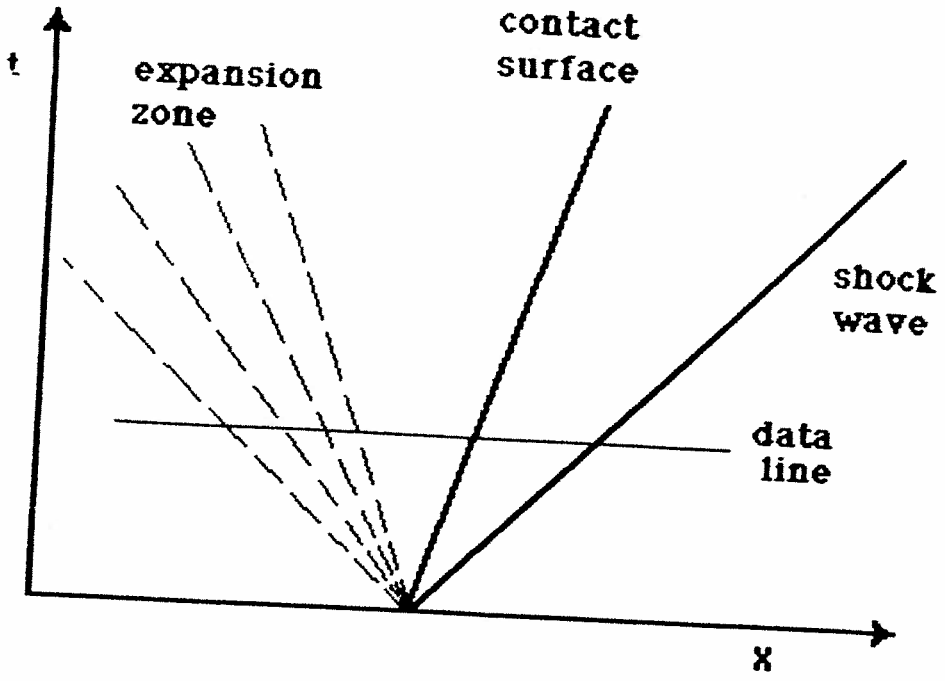


Fig. 3 *x-t diagram of the Riemann shock tube problem*

this tuning process as applied to a correction flux splitting method (CFS2). The cell boundary value interpolation/extrapolation weighting factor, ς, is varied, coupled with the inclusion or exclusion of a sharpening technique described below. The value of ς ranges from zero (only interpolated values used) to one (only extrapolated values used).

Internal energy, e, is chosen as the monitored output since it has proven to be a more difficult parameter to characterize by CFD codes of this type. The contact (entropy) surface between the gas volumes, invisible in the pressure and velocity solutions, is apparent in the energy solution. Reducing numerical diffusion across this surface is a challenging CFD problem. Thus, scrutiny of the energy solution provides a critical means of evaluation of solution techniques.

Figure 4 shows the energy solutions plotted against shock tube position following diaphragm rupture at a pressure ratio (R_p) of 100. A leftward moving expansion fan and rightward moving shock wave and contact surface are characterized by the solution. The code solution in this case involves only extrapolation in the calcula-

tion of cell wall values. Although the most numerically stable technique, dispersion causes discontinuities, which should be sharp, to be spread out over several cells. This is most evident at the contact surface and in the rounding of the expansion fan leading edge. The normal shock wave, however, is well modeled, typical of codes using higher order techniques nearly everywhere and reverting to first order in regions of large curvature or large gradients. Figures 5 and 6 show the effect of increasing the weighting of values interpolated across the cell wall in the flux calculation. Numerical dispersion may be reduced at the expense of solution stability as is evident in Figure 6 where pure interpolation is used.

Figures 7 to 9, corresponding to Figures 4 to 6, show the result of a sharpening technique. When reduction from higher order extrapolation/interpolation techniques to first order calculation is called for, the value from the opposite side of the cell boundary is chosen as the boundary value if it is closer to the value predicted by the higher order calculation. Figure 7 illustrates the introduction of unacceptable stability degradation as the sharpening technique counteracts the reduction of order. Figure 8 shows the optimum result of combining interpolation weighting and sharpening as applied to the correction flux splitting technique.

The first order Godunov method is also tested by the Riemann shock tube problem as shown in Figure 10. This method produces flow features of the same quality as those of the CFS2 method. Both methods require equivalent computation time.

Riemann's Problem becomes a more stringent test at higher pressure ratios. The resulting large flow velocities increase errors in the internal energy calculation ($e = [e_t/\rho] - [u^2/2]$) which propagate throughout the solution. Figures 11 to 14 show solution comparisons of the four state variables ρ, u, e, and p for the tuned (ς=0.37 with sharpening) CFS2 method at R_p=100. Figure 15 shows the degradation of the solution at R_p=1000. Neither of the CFS methods survive to produce solutions at R_p=3000. However, the GOD1 method remained stable at pressure ratios as high as R_p=10,000 as illustrated by Figure 16. Solution degradation is ap-

parent but quite acceptable considering the severity of the test conditions. GOD1 solution quality at more moderate pressure ratios is equal to that of the CFS2 method.

The superiority of the Godunov method at high velocities is confirmed by a related one-dimensional test in which the two gas volumes are given initial velocities toward each other at a combined speed of 200 km/sec. Only the Godunov method produces acceptable results, shown in Figures 17 and 18. The velocity errors produce large spurious energy drops in front of shock waves when the CFS methods are used. To enable the code to analyze extreme Mach number flows, a special safety feature is included. This feature overwrites ρ and e values which are negative at the end of a timestep with some fraction (usually 0.2) of the value at the beginning of the timestep. By examining conservation of energy (which this technique violates) and the ρ and e values in detail, the frequency with which this feature is invoked can be determined. Using the Godunov flux solver, this feature is very rarely invoked, even at impact velocities of hundreds of kilometers per second. The few instances when the code uses this feature are usually at the beginning of the impact process, when the 2 to 3 cell wide shock waves are first developing. Under more moderate conditions, the overwriting feature is never used.

Since the code is intended to be applied to very high speed flows, the first order Godunov method is chosen as the standard computational technique for the remaining test cases.

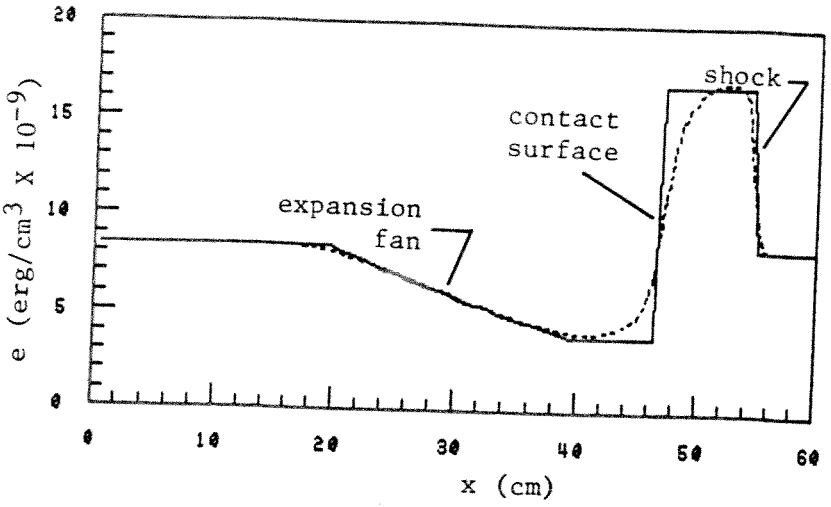


Fig. 4 Riemann's problem energy solution comparison:
CFS2 ς=1 without sharpening
(solid line, analytic solution;
dashed line, code solution)

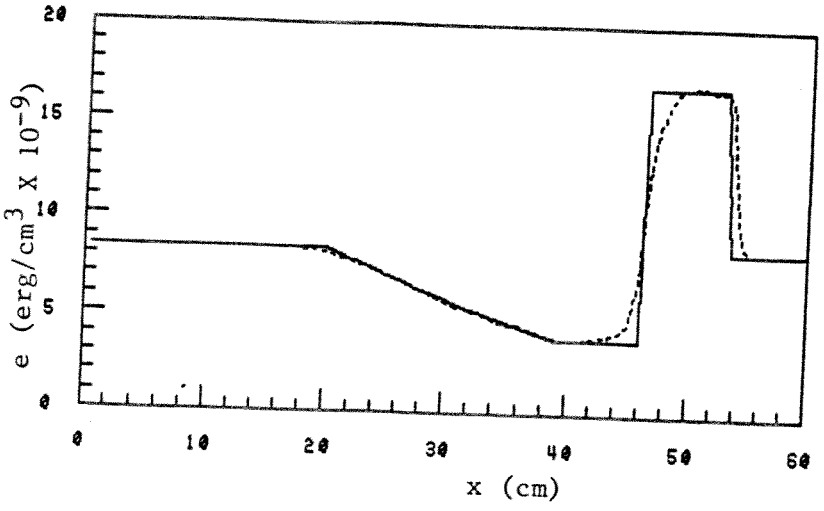


Fig. 5 Riemann's problem energy solution comparison:
CFS2 ς=0.37 without sharpening

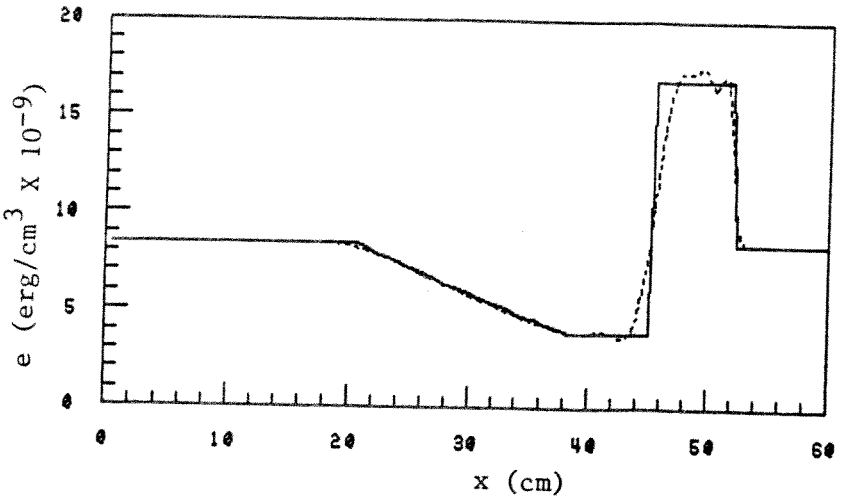


Fig. 6 Riemann's problem energy solution comparison:
CFS2 ς=0 without sharpening

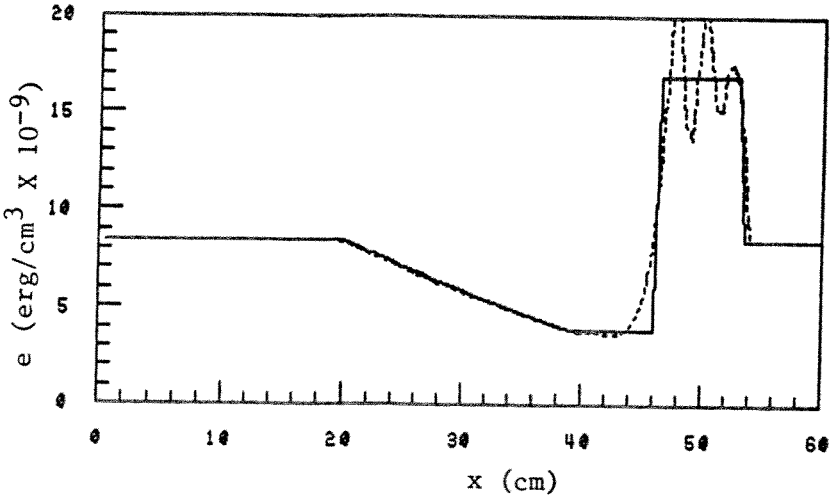


Fig. 7 Riemann's problem energy solution comparison:
CFS2 ς=1 with sharpening

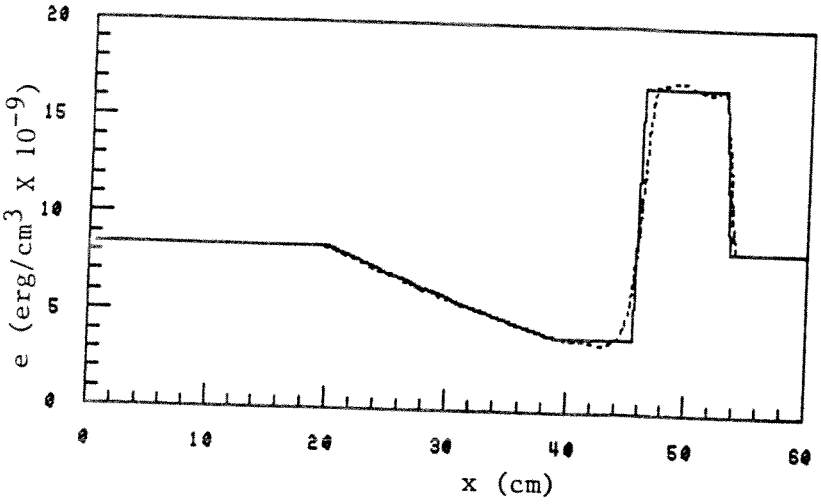


Fig. 8 Riemann's problem energy solution comparison:
CFS2 ς=0.37 with sharpening

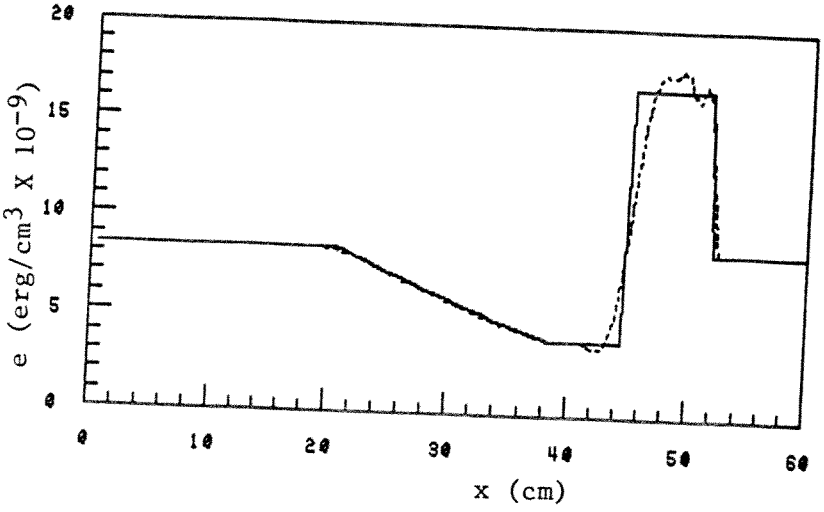


Fig. 9 Riemann's problem energy solution comparison:
CFS2 ς=0 with sharpening

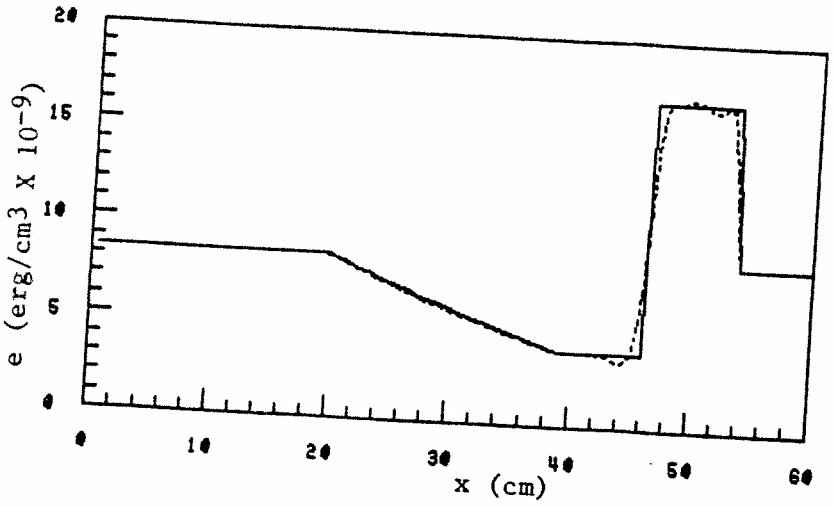


Fig. 10 Riemann's problem energy solution comparison: GOD1

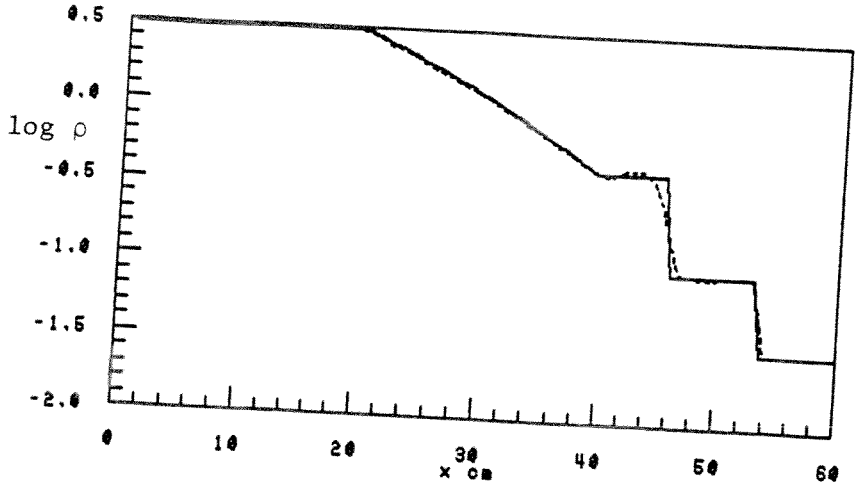


Fig. 11 Riemann's problem density solution comparison:
CFS2 R_p=100

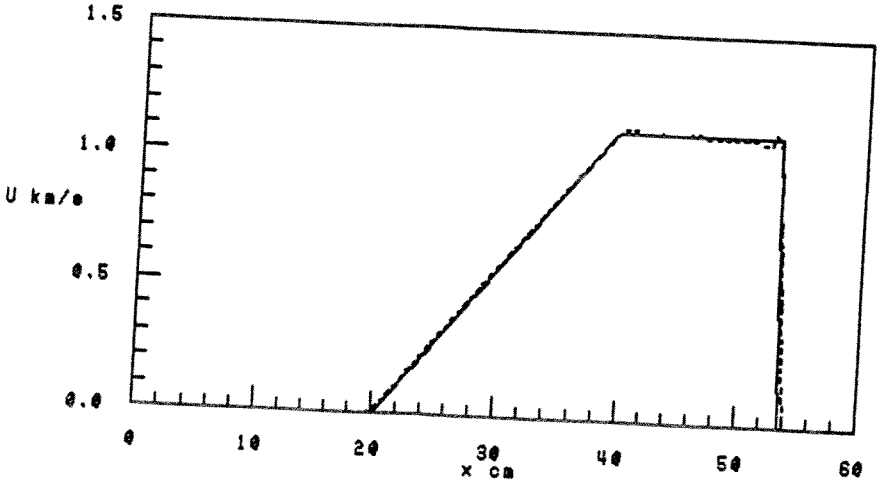


Fig. 12 Riemann's problem velocity solution comparison:
CFS2 R_p=100

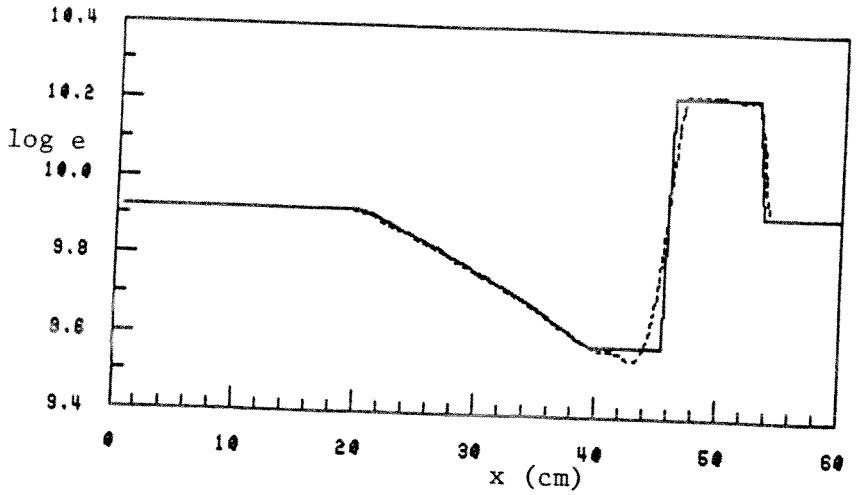


Fig. 13 Riemann's problem energy solution comparison:
CFS2 R_p=100

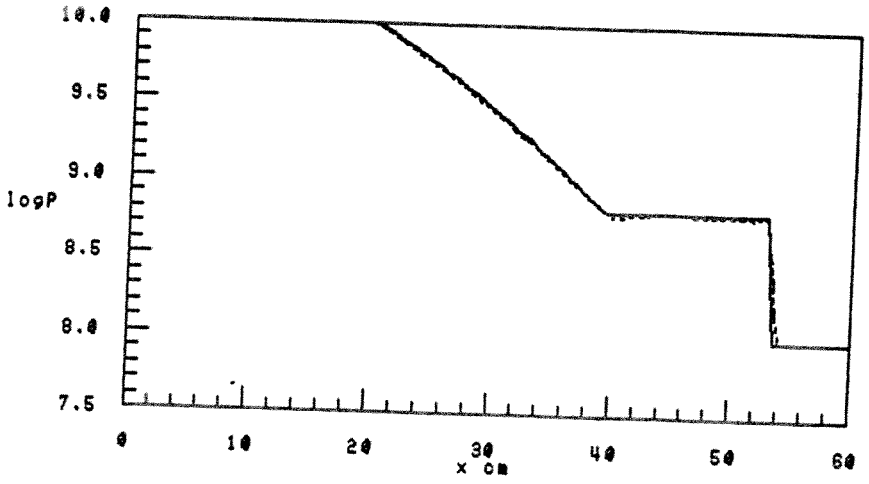


Fig. 14 Riemann's problem pressure solution comparison:
CFS2 R_p=100

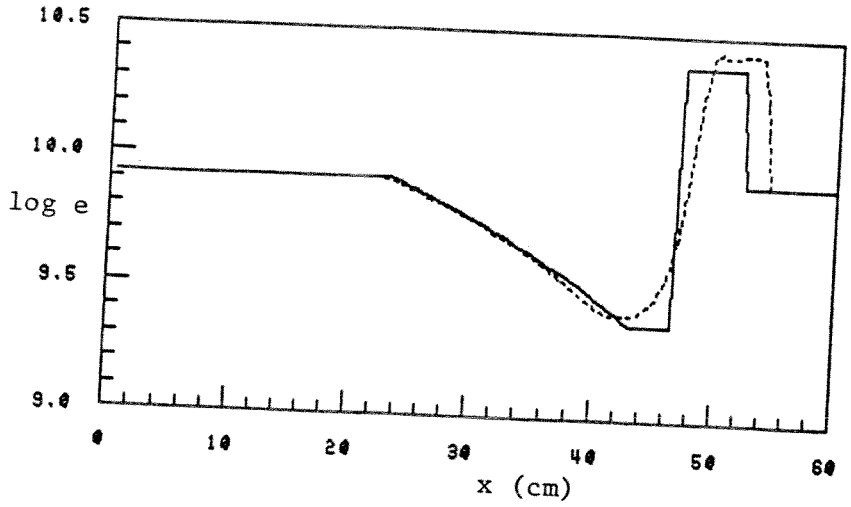


Fig. 15 Riemann's problem energy solution comparison:
CFS2 R_p=1000

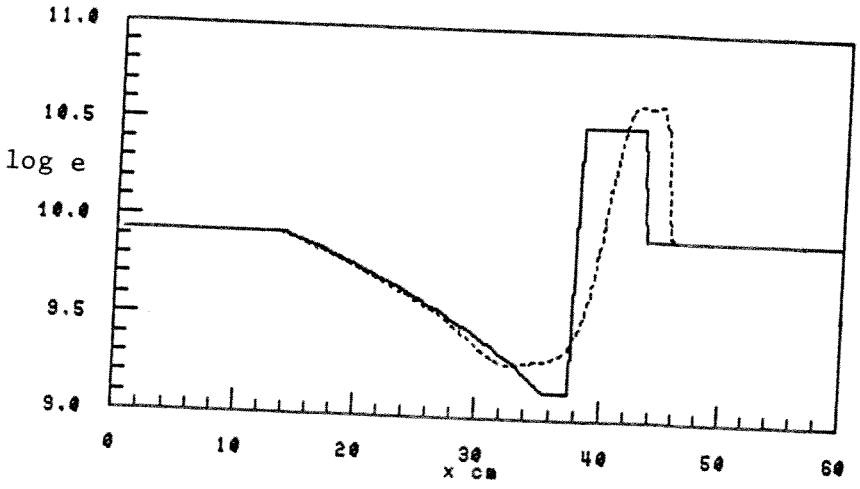


Fig. 16 Riemann's problem energy solution comparison:
GOD1 R_p=10,000

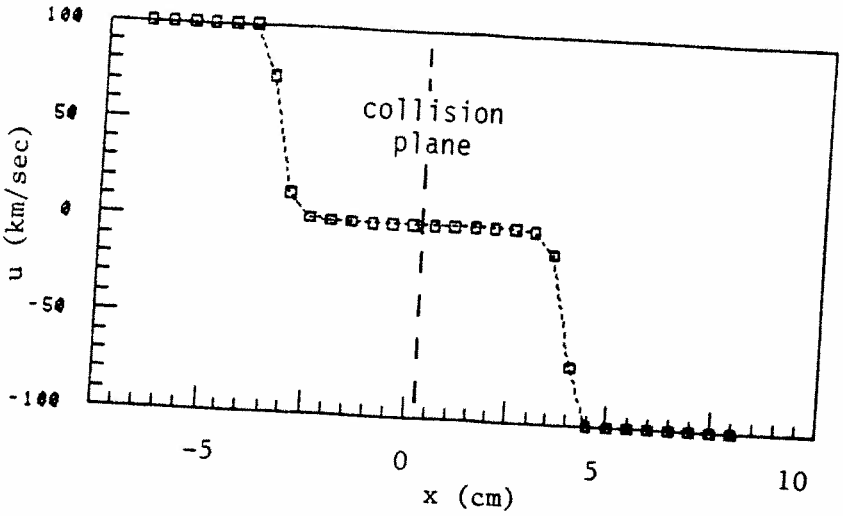


Fig. 17 High velocity impact results: GOD1 velocity solution

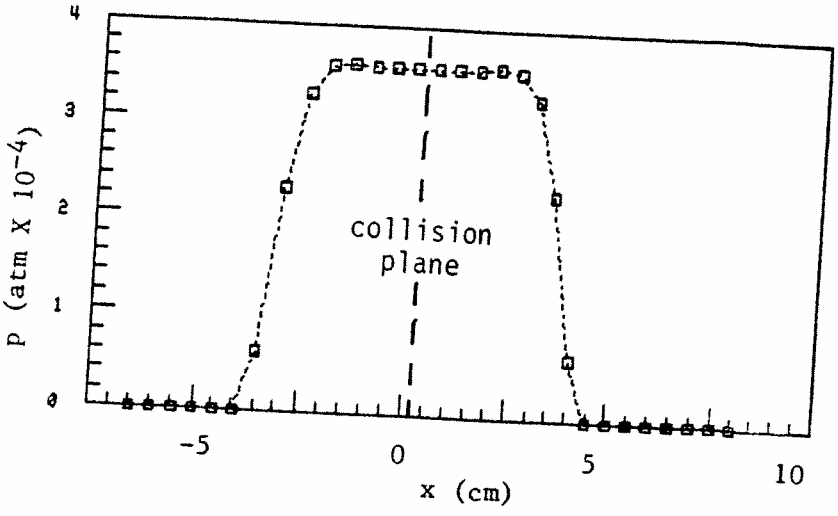


Fig. 18 High velocity impact results: GOD1 pressure solution

B. Oblique Shock Wedge Flow

Two-dimensional supersonic flow over a wedge produces an oblique shock wave attached to the wedge vertex. Placed in a duct, the shock wave reflects from the wall, keeping the flow parallel to the wall. The resulting flow field can be described analytically.[7] This wedge flow configuration is readily modeled by the code. It provides an opportunity to test the code's two-dimensional computation capability and accuracy in modeling flow interaction with solid surfaces. Also, this test permits examination of solution degradation near shock waves not aligned with the principal grid directions, a problem associated with many CFD codes.

Test conditions consist of a Mach 5.5 inlet flow, with γ=1.4 , passing over a 24° wedge and bounded by a solid wall. (These parameters were chosen to correspond to previous analysis done on the oblique detonation ram accelerator concept.) Solution data points are taken along vertical lines, (a) and (b), intersecting the initial and reflected shock waves as illustrated in Figure 19. The state variables ρ,u, v, e, and p are plotted against distance from the wedge surface for the initial shock wave in Figures 20 to 24, and for the reflected shock wave in Figures 25 to 29. The width of the oblique shock wave is limited to three computational cell widths. It should be noted that the analytic solutions, represented by solid lines in Figures 20 to 29 actually contain jump discontinuities rather than the finite thickness shock regions shown in these computer generated graphs. Maximum errors in uniform flow regions away from shock waves and boundaries are about 5% for p, u, and v behind each shock wave. Maximum errors in ρ and e are 2% behind the initial shock wave and 7% behind the reflected shock wave. Errors at the duct wall downstream of the shock wave reflection are slightly higher, but remain acceptable.

The wedge flow test cases are run using the grid refinement technique to accelerate convergence. The solution converges initially on a coarse grid (21×5). The number of cells is quadrupled and the code is restarted. This process is repeated in reaching the finest grid size (84×20). Figure 30 shows the resulting residue history. The oscillatory nature of the converged residue is evident.

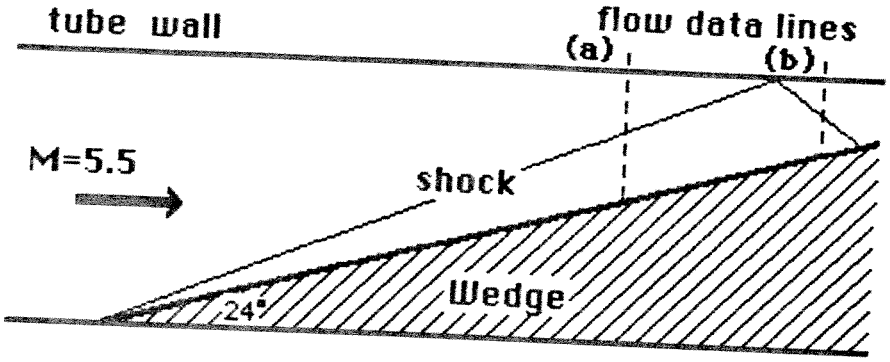


Fig. 19 Wedge flow configuration

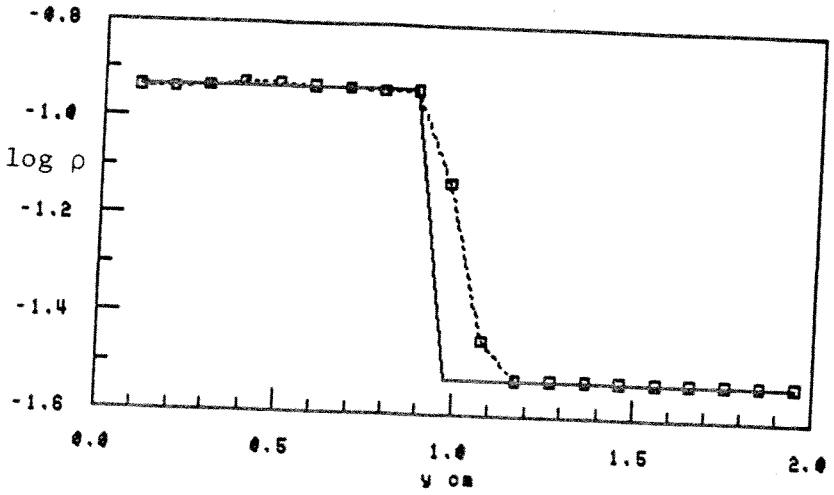


Fig. 20 Wedge flow density solution comparison:
initial shock wave
(solid line, analytic solution; dashed line, code solution)
Note: actual analytic solutions contain jump discontinuities

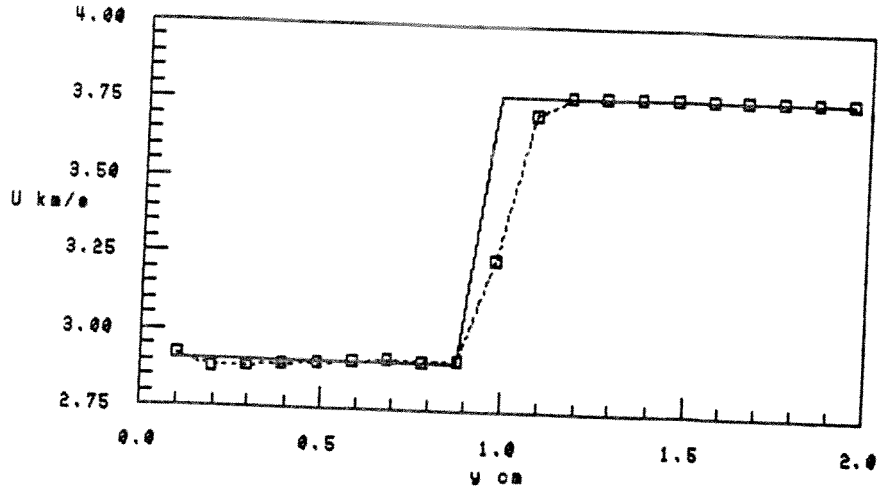


Fig. 21 Wedge flow x-velocity solution comparison:
initial shock wave

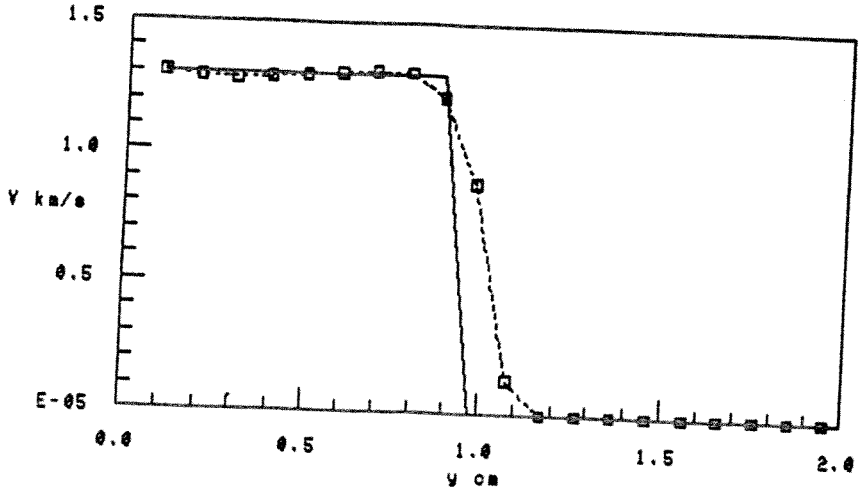


Fig. 22 Wedge flow y-velocity solution comparison:
initial shock wave

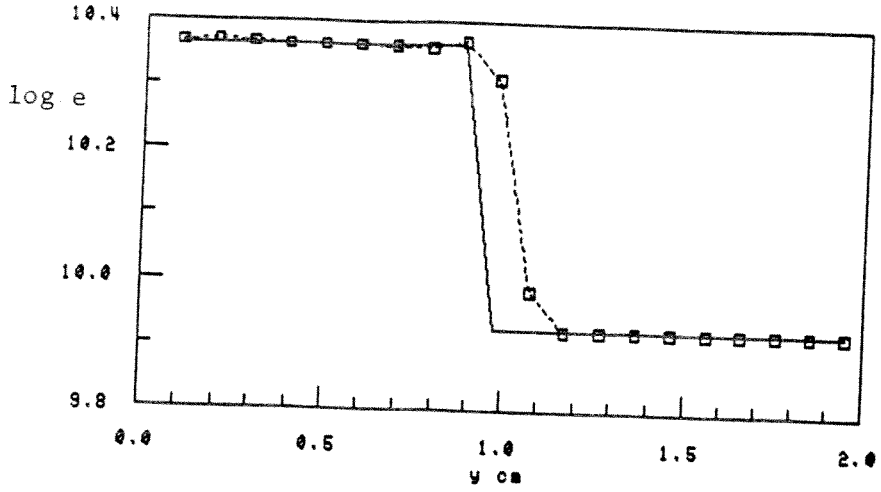


Fig. 23 Wedge flow energy solution comparison:
initial shock wave

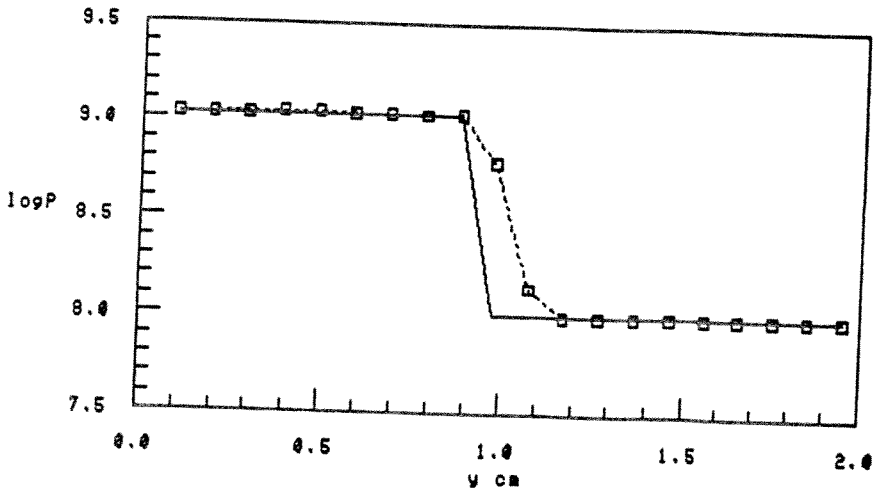


Fig. 24 Wedge flow pressure solution comparison:
initial shock wave

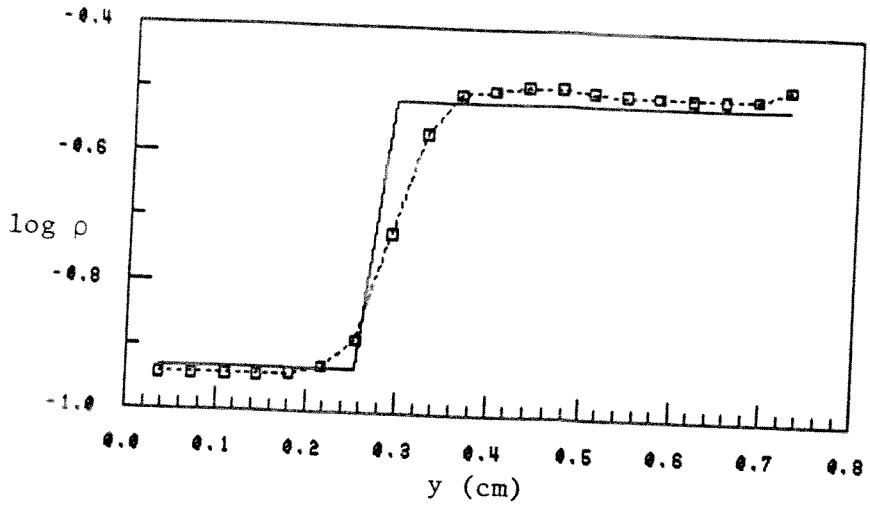


Fig. 25 Wedge flow density solution comparison:
reflected shock wave

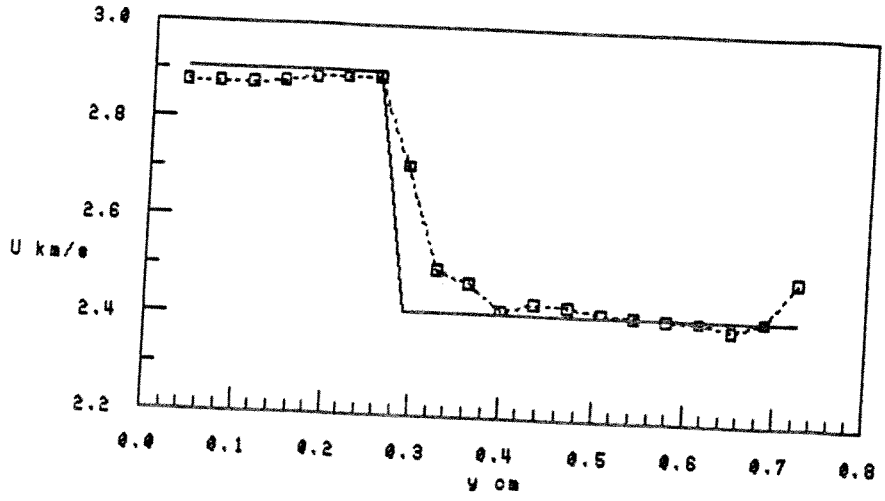


Fig. 26 Wedge flow x-velocity solution comparison:
reflected shock wave

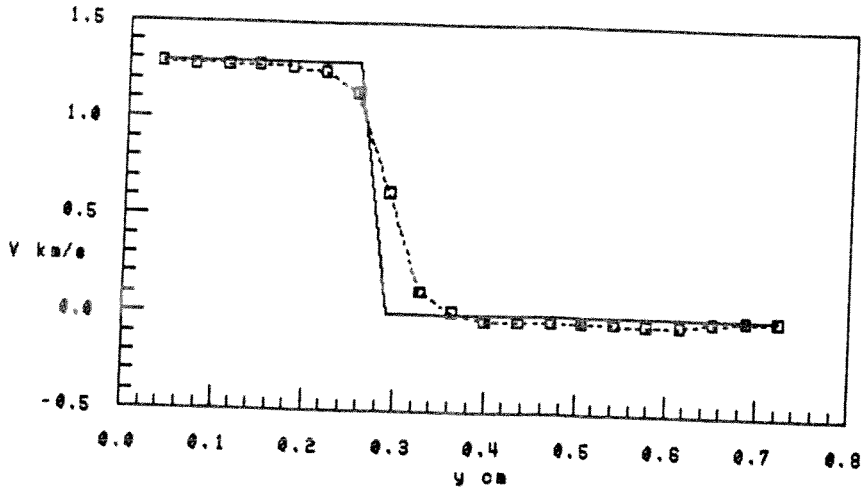


Fig. 27 Wedge flow y-velocity solution comparison:
reflected shock wave

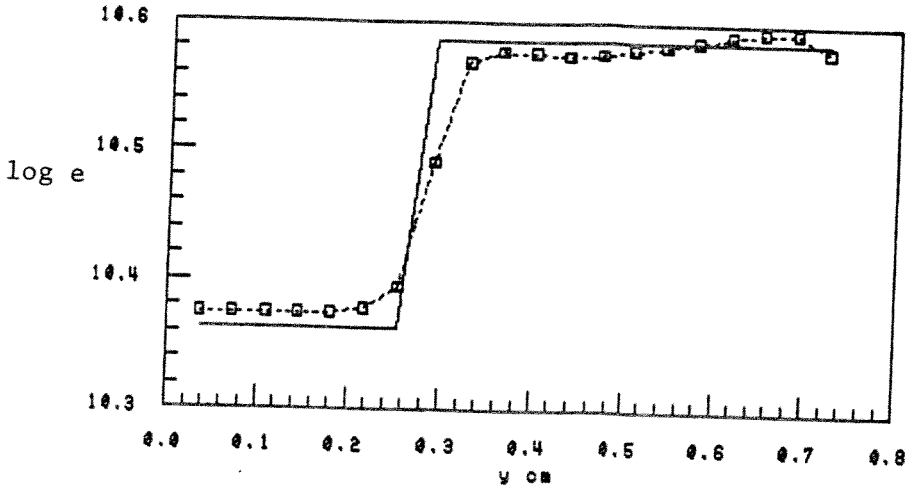


Fig. 28 Wedge flow energy solution comparison:
reflected shock wave

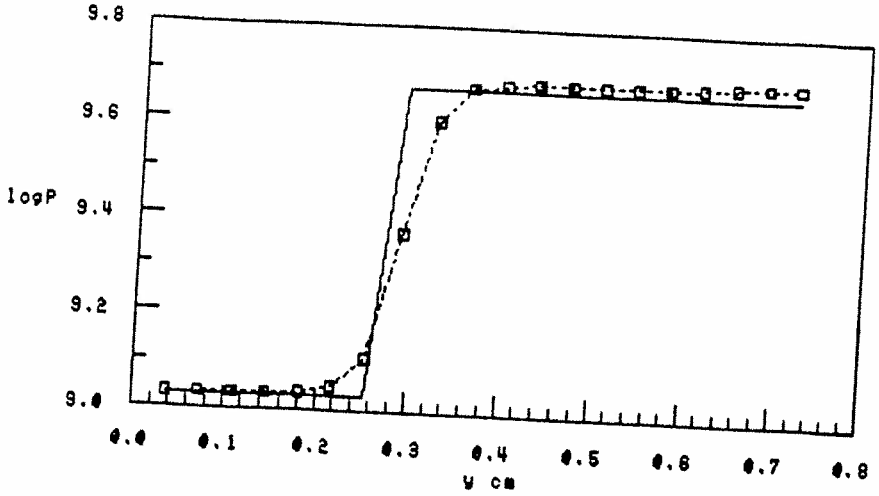


Fig. 29 Wedge flow pressure solution comparison: reflected shock wave

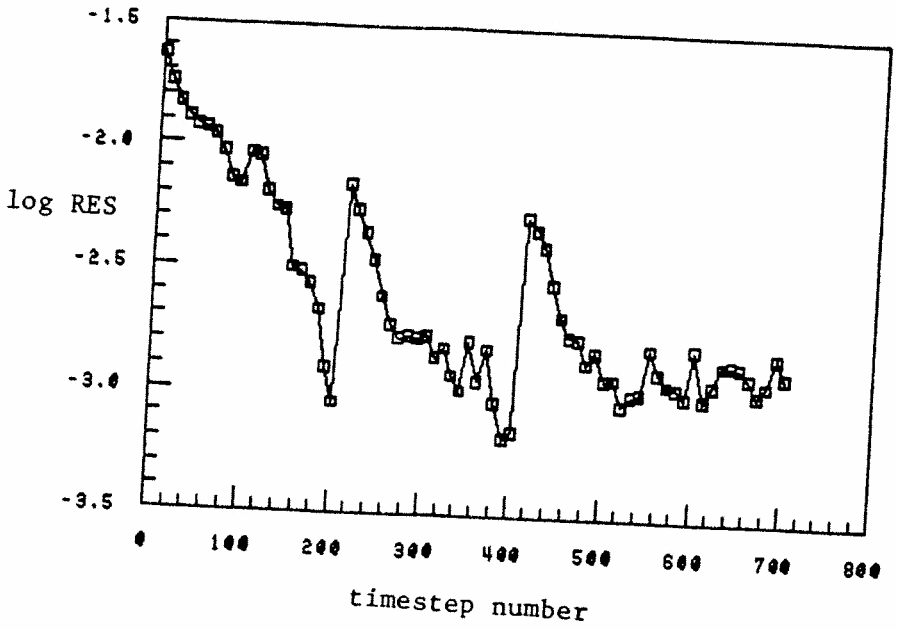


Fig. 30 Wedge flow residue history

C. Conical Shock Flow

The conical shock flow comparison is conducted to test the two-dimensional axially symmetric flow modeling capability of the code. For this test, a cone having a half-angle of 30° is subjected to freestream conditions of Mach 3 with γ=1.4 . The supersonic flow develops a conical shock wave attached to the cone vertex, resembling the wedge flow shown in Figure 19. Flow parameters between the shock wave and the cone surface are constant on concentric cones having a common vertex with the body.[8] The analytic solution used for comparison is interpolated from graphs contained in Reference 9. Figures 31 to 35 show the state variables ρ, u, v, e, and p plotted against angle from the cone axis. It should be noted that the analytic solutions, represented by solid lines in Figures 31 to 35 actually contain jump discontinuities rather than the finite thickness shock regions shown in these computer generated graphs. Figures 36 to 40 are "close-up" plots of Figures 31 to 35 with expanded vertical scales to facilitate solution comparison in the region behind the shock wave. Computational grid refinement (13×5, 26×10, 52×20) is used to accelerate convergence.

Close agreement is again obtained between analytic and code results. The shock wave is spread over three computational cells, as in the wedge flow case. Maximum errors in the region behind the shock are limited to about 1% for u, v, and p and 2% for ρ and e.

The grid refinement option is also evaluated using the conical flow test case. Figure 41 shows the solution residue history in which no grid size change occurs. Only the finest grid is used. Timestep number 600 is chosen as the "convergence point", as the residue values clearly demonstrate a stable oscillation about a final level. 505.5 CPU (central processing unit) minutes are required by the VAX 11/750 computer to reach timestep number 600 in this "fine grid only" case. Figure 42 shows the residue history in which larger computational cells are subdivided into quarters at timestep numbers 200 and 400 in reaching the fine grid size. Timestep number 660 is chosen as the convergence point for comparison. The use of grid

refinement reduces the convergence time to 270.5 CPU minutes which is 47% less than the no-refinement time. Although the converged solution residue is less by approximately a factor of two in the constant grid case, the slight decrease in accuracy is an acceptable penalty to pay for a substantial reduction in convergence time. Fine tuning the exact number of timesteps calculated for each grid size may allow some further increase in code speed. Raising the CFL number to 0.9 is also under investigation presently as another means of increasing code speed.

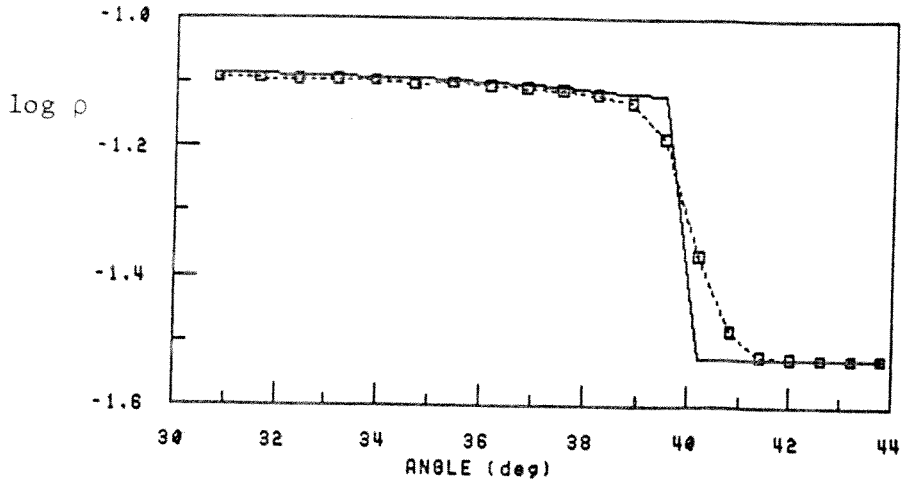


Fig. 31 Conical flow density solution comparison:
across the shock wave
(solid line, analytic solution;
dashed line, code solution)
Note: actual analytic solutions contain jump discontinuities

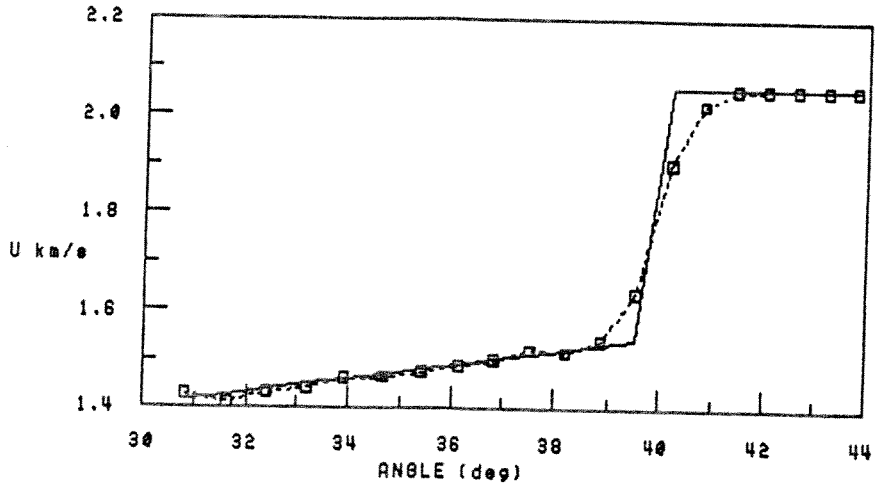


Fig. 32 Conical flow x-velocity solution comparison:
across the shock wave

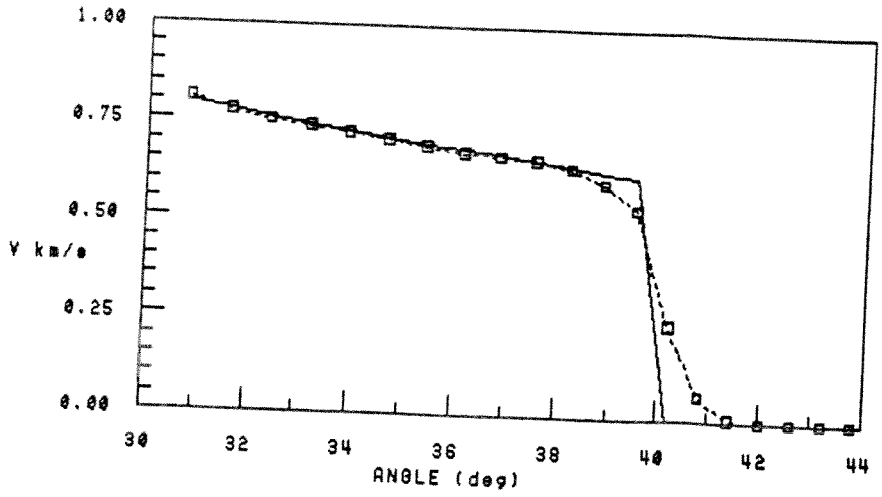


Fig. 33 Conical flow y-velocity solution comparison: across the shock wave

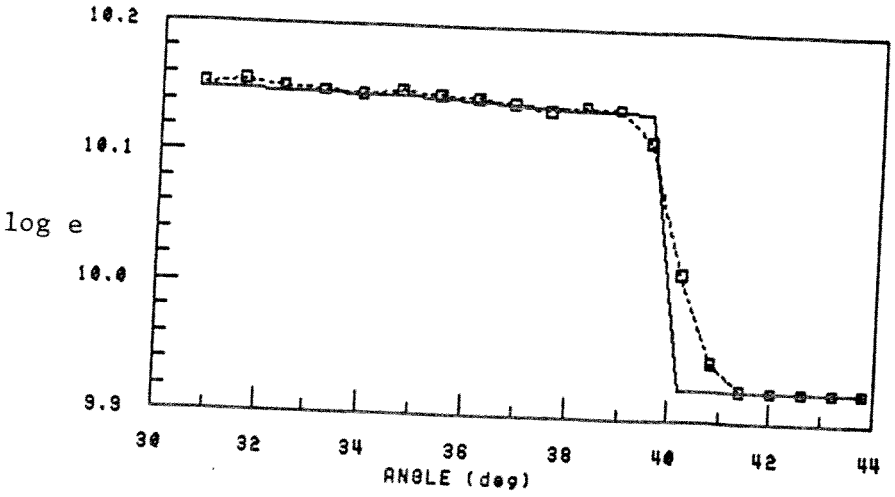


Fig. 34 Conical flow energy solution comparison: across the shock wave

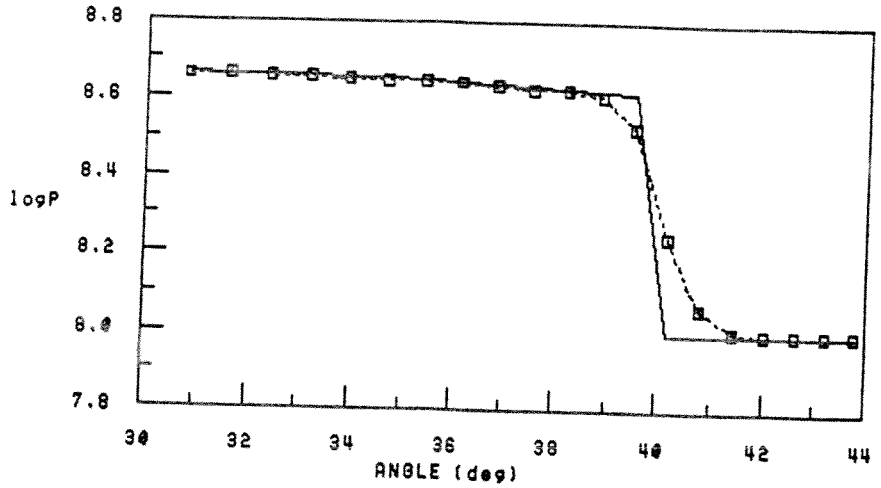


Fig. 35 Conical flow pressure solution comparison:
across the shock wave

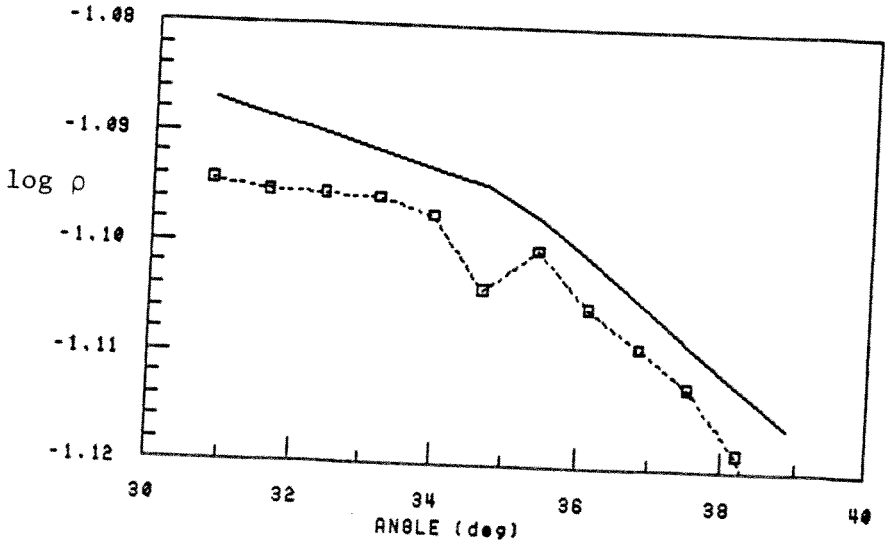


Fig. 36 Conical flow density solution comparison:
behind the shock wave

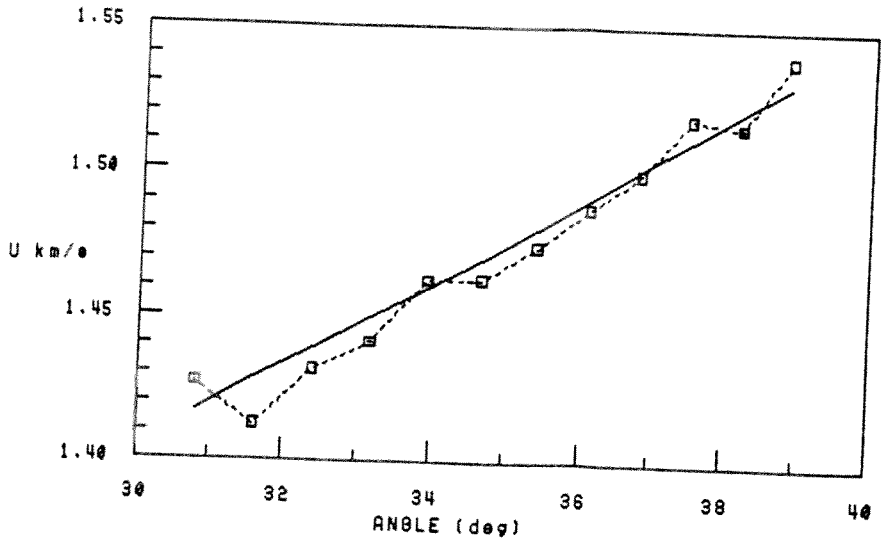


Fig. 37 Conical flow x-velocity solution comparison: behind the shock wave

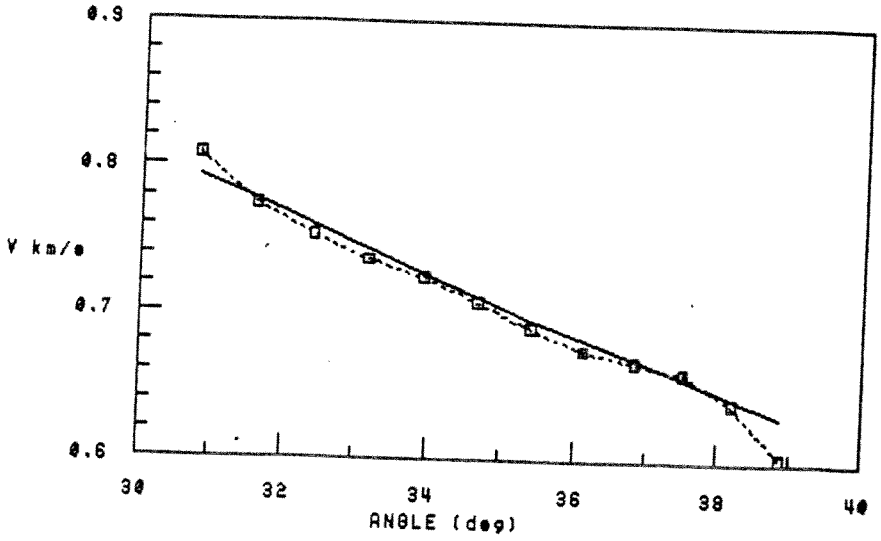


Fig. 38 Conical flow y-velocity solution comparison: behind the shock wave

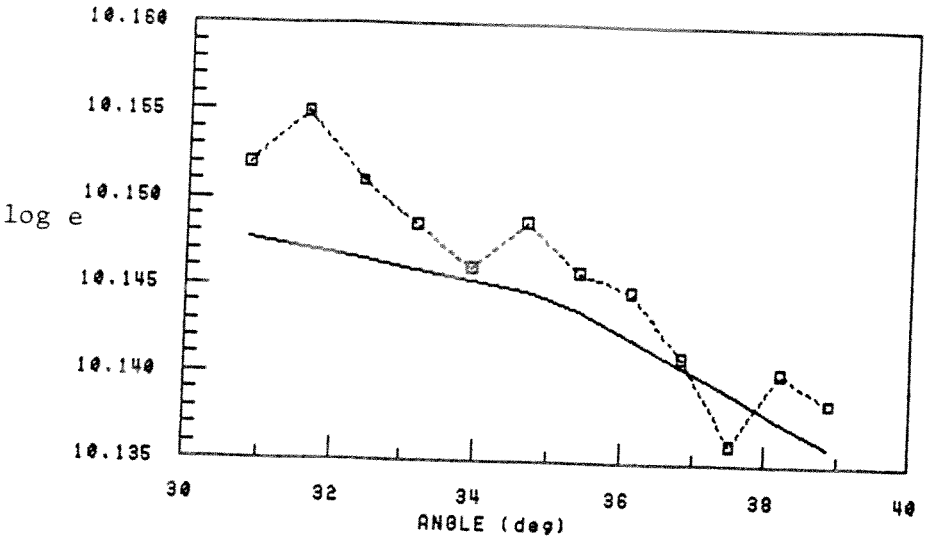


Fig. 39 Conical flow energy solution comparison:
behind the shock wave

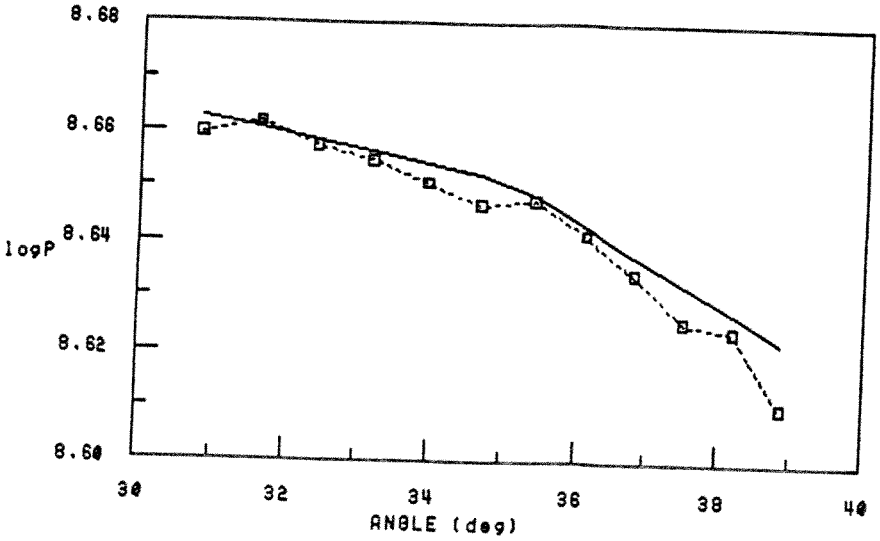


Fig. 40 Conical flow pressure solution comparison:
behind the shock wave

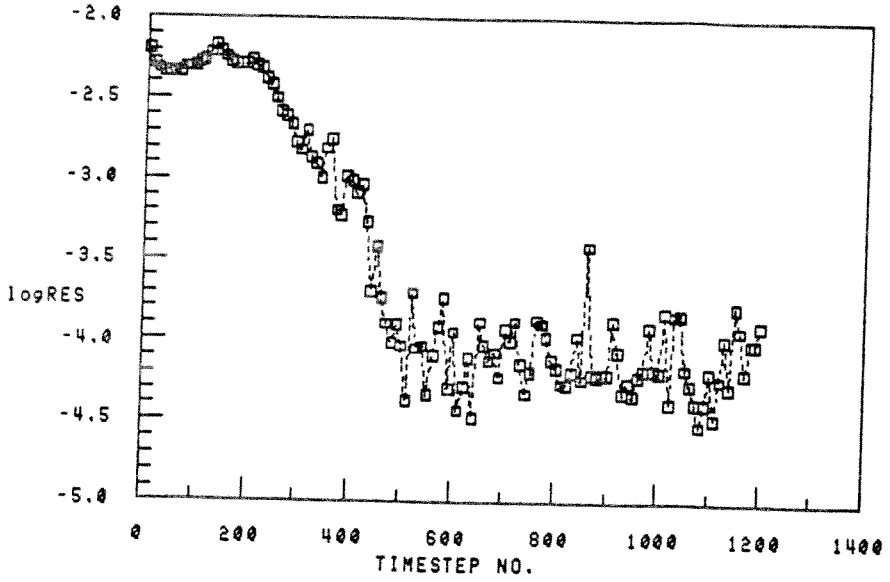


Fig. 41 Conical flow residue history: without grid refinement

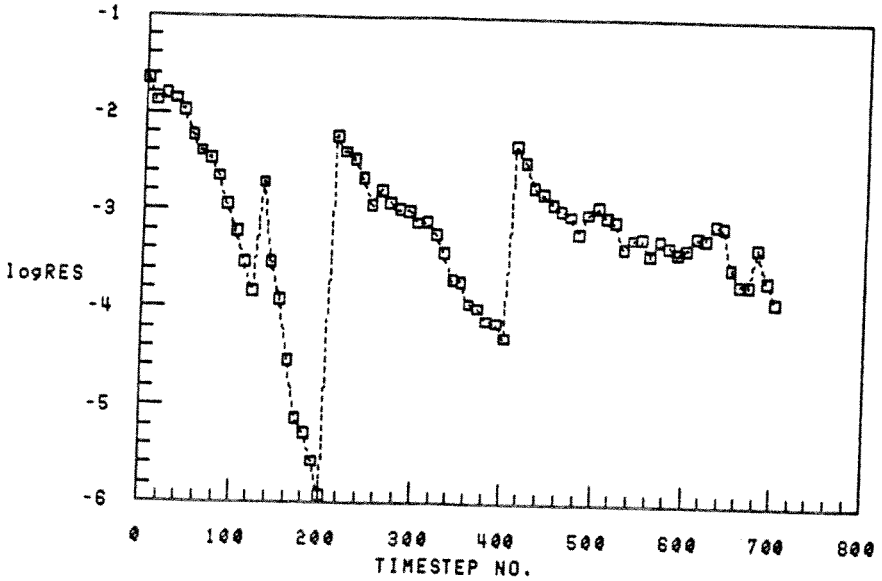


Fig. 42 Conical flow residue history: with grid refinement

CHAPTER IV

CONCLUSIONS

Much has been accomplished toward achieving the goal of developing a sophisticated yet economical computational tool, capable of analyzing axially symmetric or planar multi-component combustible flows up to hypersonic speeds. Many tasks remain in further code development. Work in progress includes testing of the code's ability to model the combustion and detonation processes. One-dimensional problem solving involving the impact of reactant gas volumes is yielding promising results. Non-ignition, Chapman-Jouget detonation, and overdriven detonation as impact velocity is increased have been satisfactorily modeled.

Preliminary work on expansion of the code to include viscous transport and radiation effects has begun. This will permit boundary layer, separated flow recirculation zones and projectile ablation to be modeled. Also, inclusion of tabulated thermodynamic data (from the JANAF tables or from the SESAME[10] tables obtained from the Los Alamos National Laboratory) will soon allow general equation of state calculations to be performed.

The code, in its present form, provides satisfactory results for the stringent test cases presented herein. The outlook is promising for its application to the ram accelerator project as well as other related fluid dynamics problems.

REFERENCES

1 Hertzberg,A., Bruckner, A. P. and Bogdanoff, D. W. "An Exploratory Study of Ram Accelerator Principles." AERP-RC-84-1 University of Washington, Seattle. Sept. 30, 1984.

2 MacCormack,R. W. "Current Status of Numerical Solutions of the Navier-Stokes Equations." AIAA Paper No. 85-0032, Reno, Nevada. Jan. 14, 1985.

3 Eidelman,S., Colella, P. and Shreeve, R. P. "Application of the Godunov Method and Its Second-Order Extension to Cascade Flow Modeling." AIAA Journal Vol. 22, No. 11, Nov. 1984.

4 Shapiro,A. H. The Dynamics and Thermodynamics of Compressible Fluid Flow Vol. I, p. 46. New York: The Ronald Press Company, 1954.

5 Anderson,D. A., Tannehill, J. C. and Pletcher, R. H. Computational Fluid Mechanics and Heat Transfer New York: McGraw-Hill Book Company, 1984.

6 Liepmann,H. W. and Roshko, A. Elements of Gasdynamics, Chap. 6. New York: John Wiley & Sons Inc., 1957.

7 Shapiro,A. H. The Dynamics and Thermodynamics of Compressible Fluid Flow Vol. I, Chap. 15. New York: The Ronald Press Company, 1954.

8 Shapiro,A. H. The Dynamics and Thermodynamics of Compressible Fluid Flow Vol. II, p. 653. New York: The Ronald Press Company, 1954.

9 Rosenhead,L. et al. A Selection of Graphs for use in Calculations of Compressible Airflow Oxford: The Clarendon Press, 1954.

10 Bennett,B. I., Johnson, J. D., Kerley, G. I. and Rood, G. T. Los Alamos Scientific Laboratory Report LA-7130 (UC-34) 1978.

BIBLIOGRAPHY

Addessio,F. L. et al. "CAVEAT: A Computer Code for Fluid Dynamics Problems with Large Distortion and Internal Slip." Los Alamos National Laboratory Report LA-10613-MS (UC-32) Feb. 1986.

MacCormack,R. W. and Paullay, A. J. "The Influence of the Computational Mesh on Accuracy for Initial Value Problems with Discontinuous or Nonunique Solutions." Computers & Fluids Vol. 2, 1974.

Ribe,F. L., Christiansen, W. H. and MacCormack, R. W. "Analysis of Impact Fusion Target Dynamics." UWFPP-35 University of Washington, Seattle. Sept. 15, 1983.

Ribe,F. L., Christiansen, W. H. and MacCormack, R. W. "Analysis of Impact Fusion Target Dynamics." UWAERP/3 University of Washington, Seattle. March 4, 1985.

Warming,R. F. and Beam, R. M. "On the Construction and Application of Implicit Factored Schemes for Conservation Laws." SIAM-AMS Proceedings Vol. 11, 1978.

Yee,H. C., Beam, R. M. and Warming, R. F. "Boundary Approximations for Implicit Schemes for One-Dimensional Inviscid Equations of Gasdynamics." AIAA Journal Vol. 20, No. 9, Sept. 1982.