A NUMERICAL ANALYSIS OF NON-STEADY SUPERSONIC DUCT FLOWS WITH HEAT ADDITION

by

John Mahlon Zulauf

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

Master of Science in Aeronautics and Astronautics

University of Washington

1985

Approved by ______________________________
(Chairperson of Supervisory Committee)

Program Authorized
to Offer Degree Aeronautics and Astronautics

Date ______________________________

University of Washington

Abstract

A NUMERICAL ANALYSIS OF NON-STEADY SUPERSONIC DUCT FLOWS WITH HEAT ADDITION

By John Mahlon Zulauf

Chairperson of the Supervisory Committee: Professor Abraham Hertzberg
Department of Aeronautics and Astronautics

The transient response of supersonic duct flows to the sudden initiation of combustion is of interest for many practical gasdynamic devices. Such flows may be considered "generalized" ramjets, for which steady-state solutions are well known. The transient response is approximated as a solution to the quasi-one-dimensional Euler equations. These equations are modeled by a finite difference technique that is second order accurate in time and first order accurate in space. The numerical model is applied to a fixed geometry convergent-divergent nozzle at various inlet velocities and with three different working fluids. The transient response is classified as stable or non-stable and the heat addition rate is compared to the maximum heat addition leading to a stable steady state solution. Typical results for stable and unstable transients are presented.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

<u>Roman letters</u>

A	Area
C_p	Specific heat at a constant temperature
CFL	Courant-Friedrichs-Lewy number
c	Speed of sound
e	Total energy per unit of volume
F	Flux vector
F_I	Generalized flux vector
H	Pressure source vector
M	Mach number
p	Pressure
Q	Generalized heat addition vector
q	Heat release per unit mass
T	Temperature
t	Time
U	State variable vector
u	Local material velocity
V	Velocity
W	Shock speed
x	Physical space variable

Greek letters

β	Defined as $\gamma - 1$
γ	Ratio of specific heats
Λ	Diagonalize Jacobian Matrix
λ	Eigenvalue (characteristic speed)
ν	Dummy argument for eigenvalue in F_I

Subscripts and Superscripts

i	Grid point
n	Time step value
x	Conditions upstream of shock wave
y	Conditions downstream of shock wave
*	Critical value
+	Positive eigenvalues
-	Negative eigenvalues
max	Maximum value
0	Total (stagnation) quantity

Special Symbols

∇_x	Backward difference operator
Δ_x	Forward difference operator

ACKNOWLEDGMENTS

The author wishes to express his deepest appreciation to Prof. Abraham Hertzberg for his encouragement toward clear analysis and good engineering. The contribution of Dr. Adam Bruckner has been invaluable. The frequent discussions, edits, praise, and critisisms were crucial to the succesful completion of this work. Special thanks to David W. Bogdanoff for input, frequent editing, and problem solving. Also, thanks to all those who through their comradeship and conversation helped keep the author on an even keel.

I. INTRODUCTION

Supersonic duct flows may be found in a number of gasdynamic devices. Among them are jet engines, shock tubes, rocket engines, and light-gas guns. Of these, only the jet engine has significant heat addition downstream of regions of supersonic flow. In turbojet or turbofan engines regions of heat addition are typically separated from regions of supersonic flow by turbo-machinery. Thus the supersonic duct with heat addition may be considered a "generalized" ramjet. This "generalized" ramjet is composed of a duct with varying area containing a region or regions of heat addition, or combustion. This corresponds to a ramjet's diffuser, combustor, and exit nozzle. For subsonic combustion a normal shock is required at some point in the flow. The location of this shock is dependent on nozzle exit conditions and heat addition rates.

The known steady-state results for the ramjet are based on an ideal quasi-one-dimensional analysis. "Ideal quasi-one-dimensional" is defined as an analysis which ignores viscous effects and allows entropy increases only at normal shock waves or within regions of heat addition. Also, all flow variables are considered constant across each cross-section. Velocities normal to the main flow direction are considered small with respect to the main flow velocity. The only two-dimensional effect included is that of the area change. Despite the number of simplifying assumptions stated above, the analysis still yields a good first-order approximation of the actual device behavior. The steady-state ideal ramjet equations form a solution to the steady-state quasi-one-dimensional Euler equations. The transient response is a solution to the time-dependent quasi-one-dimensional Euler equations. These equations may be modeled in several ways, such as the method of characteristics or various types of finite difference techniques. The method of characteristics fails in mixed supersonic-subsonic flows. Finite difference techniques may be applied with an accuracy comparable to the steady-state solutions, and can be used in all types of flows. Steady-state solutions to ramjet combustion and shock location are given in Chapter III.

The transient response of a supersonic diffuser to the initiation of combustion downstream of the diffuser does not lend itself to simple solution. Combustion processes that yield stable steady-state solutions may have transient responses that cause the diffuser to "unstart". The "unstart" condition causes the ramjet to be a net drag producer instead of a net thrust producer. For a ramjet this is totally unacceptable, however, there may exist gasdynamic devices in which shock movement through a diffuser or nozzle may be part of the normal operation of the device. For many cases, the transient response is of interest.

In order to model this transient response a finite difference model of the Euler equations of fluid motion is developed (Chapter III). This model is implemented by a FORTRAN program included as Appendix A. To verify the accuracy of the finite difference technique used to model the equations, results from this technique are compared (Chapter IV) with analytic solutions to two classical gasdynamic problems.

A single ramjet configuration is introduced in Chapter V for extensive analysis of the effects of working fluid and inlet velocity on the transient response. A graphical representation of a typical stable and unstable response is given. The results of three different working fluids at various inlet velocities are presented in tabular form. These results yield insight into the relative importance of parameters describing the ramjet configuration to the behavior of the transient response. Conclusions based on these results are given and recommendations are made for further investigation (Chapter VI).

II. BACKGROUND

A. RAMJET PARAMETERS

1) Description

In Chapter I it was noted that a supersonic duct flow may be characterized as a "generalized" ramjet. Several parameters are used to describe the configuration of any particular ramjet. These parameters are the area profile of the diffuser, the distribution and timing of the energy release in the combustion region, the nature of this energy release, and the exit condition of the ramjet. For the "generalized" ramjet these parameters may take on values significantly different from those considered in propulsion applications.

2) The Diffuser

The diffuser, for subsonic combustion in steady operation, is typically a convergent-divergent nozzle with a normal shock located downstream of the nozzle throat. The minimum entropy increase is obtained using a throat Mach number of unity and an infinitesimally weak normal shock located at the throat. In the ideal analysis, perturbations to the steady-state solution are absent, thus this configuration is neutrally stable, on the boundary of stability. Since the quasi-one-dimensional ideal relations exclude boundary layer, turbulence, and other two-dimensional effects, this configuration is, for real devices, unstable. In practical configurations the Mach number at the throat must be greater than unity and the shock must be some distance downstream of the throat. Typical throat Mach numbers for real, stable devices are in the neighborhood of 1.3 . Typical normal shock locations are such that the Mach upstream of the shock is 1.5 .

The unstable behavior is called an "unstart". The nature of the instability is that if the flow is perturbed such that the normal shock moves upstream of the throat the shock continues to propagate upstream through the diffuser and is disgorged.

3) Heat Addition Region

The second parameter describing a particular ramjet configuration is the type of heat addition or combustion. Heat addition may take place subsonically or supersonically. Supersonic combustion ramjets (SCRAMJET) are the subject of considerable study. However, in practice, all combustion is done subsonically. In a generalized ramjet the heat may be added by chemical reaction, radiation, or by electric discharge, perhaps forming a plasma. In all of these cases the character of the working fluid may be altered dramatically through dissociation, chemical reaction or ionization. This introduces an additional complexity into the analysis, especially if a finite difference technique is contemplated.

4) Downstream Exit Condition

The downstream exit condition may also be subsonic or supersonic. To maximize jet velocity and isolate the ramjet from the downstream conditions, the exit condition is most often chosen to be supersonic, or exactly sonic. This can be achieved in two ways; by using a choked convergent-divergent nozzle downstream of the combustion region, or by adding sufficient heat to thermally choke the flow downstream of the combustion region. The choke point Mach number is maintained at unity by the location of the upstream normal shock in the diffuser for either case. Steady-state solutions yield unique values for shock location for a given heat addition and downstream area profile for supersonic or sonic exit conditions only. Subsonic exit conditions require some knowledge of downstream conditions beyond the exit. With a subsonic exit, the nozzle exit pressure usually can be assumed to be known.

B. TRANSIENT RESPONSE

1) Qualitative Description

In light of the preceding discussion, a rough description of the transient response of a supersonic duct flow to the initiation of combustion can be formulated. If sufficient heat is added to choke the flow in the combustion region, a shock will form and propagate upstream into the diffuser. Depending on the configuration of the diffuser, the upstream conditions, and the amount of heat released, the shock may or may not propagate through the throat of the diffuser. If the transient response does not unstart the diffuser, the flow should settle into a steady-state solution.

2) Solution Technique

The rough description of the preceding paragraph, however, does not answer the question of whether or not the diffuser unstarts. Similarly, the steady-state, ramjet equations do not answer this question. To answer this question with at least the same order of approximation as that of the steady-state solution, a solution of the non-steady, quasi-one-dimensional Euler equations is required. A finite difference method is employed to provide numerical solutions.

With the advent of super-computers, finite difference techniques to solve the full Navier-Stokes equations have been developed[1]. The computational cost for accurate modeling using these techniques, however, is still quite prohibitive. Many of the techniques are still developmental and not readily applicable to engineering problems. The Euler equations are thus chosen as the basis of the present analysis, and as with the steady-state equations, the flow is taken to be quasi-one-dimensional with a varying flow area. The numerical method, given sufficient numerical accuracy, should yield transient solutions of the same level of accuracy as the steady-state one-dimensional ramjet equations.

III. THEORY AND METHOD

At this point the governing equations of this analysis are stated explicitly. The steady-state ramjet equations, the quasi-one-dimensional Euler equations, and the finite difference method used are discussed. To verify the accuracy of the finite difference technique, results from this technique are compared (Chapter IV) with analytic solutions to two classical gasdynamic problems. The equations leading to the analytic solutions of these two test problems are given in Chapter IV.

A. STEADY STATE RAMJET EQUATIONS

1) Isentropic Flow Relationships

The solution of the steady-state duct flow problem assumes isentropic flow except at normal shocks. The relationships between static and stagnation quantities are given by the following equations (Kuethe & Chow 1976, p. 207)[2].

$$\frac{p}{p_0} = \left[1 + \frac{\gamma - 1}{2} M^2 \right]^{-\gamma/(\gamma-1)} \qquad (1)$$

$$\frac{\rho}{\rho_0} = \left[1 + \frac{\gamma - 1}{2} M^2 \right]^{-1/(\gamma-1)} \qquad (2)$$

$$\left[\frac{c}{c_0}\right]^2 = \frac{T}{T_0} = \left[1 + \frac{\gamma - 1}{2} M^2\right]^{-1} \qquad (3)$$

Where γ is the ratio of the specific heats. Implicit in these equations is the assumption of the ideal gas equation of state, Eq. 4. Also it is assumed that the gas is calorically perfect. The speed of sound of the gas is given by Eq. 5.

$$p = \rho RT \qquad (4)$$

$$c = \sqrt{\gamma RT} \qquad (5)$$

2) Area change Relations

The equations describing the ramjet problem are required to satisfy both the continuity and momentum equations. These may be written in the forms of Eq. 6 and Eq. 7 respectively.

$$\frac{dV}{V} + \frac{d\rho}{\rho} + \frac{dA}{A} = 0 \qquad (6)$$

$$d\left(\frac{V^2}{2}\right) + \frac{dp}{\rho} = 0 \qquad (7)$$

Satisfaction of these two conditions, as well as the isentropic flow relationships, yields the following result.

$$\frac{A}{A^*} = \frac{1}{M}\left[\frac{2}{\gamma+1}\left(1 + \frac{\gamma-1}{2} M^2\right)\right]^{(\gamma+1)/(2(\gamma-1))} \qquad (8)$$

3) Normal Shock Relations

The isentropic relations above do not apply across normal shock waves, with the exception of Eq. 3, which only requires that the flow be adiabatic. In order to find the change in flow variables across the shock, the normal shock relations(Shapiro 1954, p.995)[3], or "jump conditions" must be used. These are given by Eqs. 9 - 11.

$$M_y^2 = \frac{M_x^2 + \dfrac{2}{\gamma - 1}}{\dfrac{2\gamma}{\gamma - 1} M_x^2 - 1} \tag{9}$$

$$\frac{p_y}{p_x} = \frac{2\gamma}{\gamma + 1} M_x^2 - \frac{\gamma - 1}{\gamma + 1} \tag{10}$$

$$\left(\frac{c_y}{c_x}\right)^2 = \frac{T_y}{T_x} = \frac{\left(1 + \dfrac{\gamma - 1}{2} M_x^2\right)\left(\dfrac{2\gamma}{\gamma - 1} M_x^2 - 1\right)}{\dfrac{(\gamma + 1)^2}{2(\gamma - 1)} M_x^2} \tag{11}$$

These jump conditions satisfy the Euler equations and Eq. 3.

4) Constant Area Heat Addition

a. Thermally Choked Flow

For the purposes of the current model only constant area, subsonic heat addition is considered. Application of continuity, the one-dimensional momentum equation, and the enthalpy form of the energy equation gives the following relationships(Oates 1984, p.48)[4].

$$\frac{T_{02}}{T_{01}} = \frac{f(M_2^{\,2})}{f(M_1^{\,2})} \tag{12}$$

$$f(M^2) = \frac{M^2\left(1 + \frac{\gamma - 1}{2} M^2\right)}{(1 + \gamma M^2)^2} \tag{13}$$

$$\frac{T_{02}}{T_{01}} = \left[1 + \frac{q_{1-2}}{C_p\, T_{01}} \right] \tag{14}$$

For the case of thermally choked flow the Mach number after combustion is unity. Eq. 12 then reduces to Eq. 15.

$$\frac{T_{02}}{T_{01}} = \frac{(1 + \gamma M^2)^2}{2(\gamma+1)\left(1 + \frac{\gamma - 1}{2} M^2\right)} \tag{15}$$

b. Unstart Heat Addition Limit

The maximum heat addition before unstart corresponds to a shock location at the throat. Solving the isentropic relations for each side of the diffuser and the normal shock relations at the throat, we may obtain the Mach number at the entrance to the combustion section. Knowing this, the heat addition for unstart is given by Eq. 16.

$$q_{1-2} = C_p\, T_{01} \left(1 - \frac{T_{02}}{T_{01}} \right) \tag{16}$$

Several of the equations governing supersonic duct flows with heat addition and area variation are not analytically invertible. Iterative techniques(Conte & de Boor 1980, pp. 74-81)[5], such as the bisection method and Newton's method, provide the required solutions in those cases.

B. FINITE DIFFERENCE METHOD

1) Overview

The finite difference method is a computational fluid dynamics (CFD) approximation of the Euler equations of fluid motion. The differential equations are approximated by one-sided difference operators in a predictor-corrector flux-split scheme. The method is explicitly factored and second order accurate in time and first order accurate in space.

2) Quasi-One-Dimensional Euler Equations

The one-dimensional Euler equations may be stated in the conservation law form as Eqs. 17 a - c.

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} = 0 \qquad (17a)$$

$$U = \begin{pmatrix} \rho \\ \rho u \\ e \end{pmatrix} \quad (17b) \; ; \; F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ (e + p)\ \rho \end{pmatrix} \quad (17c)$$

To include the effect of area change requires modification of the second term and the inclusion of a momentum source term. The quasi-one-dimensional Euler equations may then be written as Eqs. 18a and 18b.

$$\frac{\partial U}{\partial t} + \frac{1}{A}\frac{\partial FA}{\partial x} = \frac{H}{A}\frac{\partial A}{\partial x} \quad (18a) \; ; \; H = \begin{pmatrix} 0 \\ p \\ 0 \end{pmatrix} \quad (18b)$$

3) Flux-Split Technique

Linear stability analysis of explicit one-sided difference operators requires that they be applied such that the differencing follows the direction of wave propagation. Thus for one-sided differencing applied in a single direction, the difference must be taken upwind and the flow must be supersonic. For subsonic and/or reversing flows some intelligence must be invested in the method to determine the proper differencing direction. The flux split technique developed by Steger and Warming (1979)[6] provides this capability by applying the stability analysis to the uncoupled form of the quasi-one-dimensional Euler equations (Eqs. 19 a - d).

$$\frac{\partial U}{\partial t} + \frac{Q\Lambda Q^{-1}}{A}\frac{\partial UA}{\partial x} = \frac{H}{A}\frac{\partial A}{\partial x} \quad (19a)$$

$$Q\Lambda Q^{-1} = \frac{\partial F}{\partial U} \quad (19b)$$

$$\Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ 0 & & \lambda_3 \end{bmatrix} \quad (19c)$$

$$\lambda_1 = u \; , \; \lambda_2 = u + c \; , \; \lambda_3 = u - c \quad (19d)$$

The result of this analysis is that the flux vector F is split into two parts on the basis of the local characteristic slopes -- the eigenvalues. Since the eigenvalues give the direction on information propagation the one-sided difference operators may be applied stably. This correlates to the gasdynamic idea of following characteristic curves of both families. However it is important to stress that this technique is not the method of characteristics. Also it should be noted that the flux-split technique requires that the equations be homogeneous of degree one. This requires that pressure (p) be a linear function of the total energy (e), which is satisified by the ideal gas equation of state. The flux-split Euler equations are given by Eqs. 20 a - f.

$$\frac{\partial U}{\partial t} + \frac{1}{A}\left(\frac{\partial F^{+}A}{\partial x} + \frac{\partial F^{-}A}{\partial x}\right) = \frac{H}{A}\frac{\partial A}{\partial x} \tag{20a}$$

$$\lambda_i^{+} = \frac{\lambda_i + |\lambda_i|}{2} \quad ; \quad \lambda_i^{-} = \frac{\lambda_i - |\lambda_i|}{2} \tag{20b}$$

$$F^{+} = F_I(\lambda_1^{+},\lambda_2^{+},\lambda_3^{+}) \tag{20c}$$

$$F^{-} = F_I(\lambda_1^{-},\lambda_2^{-},\lambda_3^{-}) \tag{20d}$$

$$F_I(\nu_1,\nu_2,\nu_3) = \frac{\rho}{2\gamma}\begin{bmatrix} 2\beta\nu_1 + \nu_2 + \nu_3 \\ 2\beta\nu_1\lambda_1 + \nu_2\lambda_2 + \nu_3\lambda_3 \\ \beta\nu_1\lambda_1^{2} + \frac{\nu_2}{2}\lambda_2^{2} + \frac{\nu_3}{2}\lambda_3^{2} + w \end{bmatrix} \tag{20e}$$

$$\beta = \gamma - 1 \; ; \qquad w = \frac{(3-\gamma)(\nu_2 + \nu_3)c^2}{2\beta} \tag{20f}$$

Eq. 20b defines the flux-split eigenvalues; Eq. 20c and 20d define the flux-split flux vectors in terms of those eigenvalues. The eigenvalues and flux vectors defined above also have the following qualities.

$$F = F^{+} + F^{-} \tag{21}$$

$$F = F_{I}(\lambda_1, \lambda_2, \lambda_3) \tag{22}$$

$$\lambda_i = \lambda_i^{+} + \lambda_i^{-} \tag{23}$$

With the two flux terms consistently stable in one-sided spatial differencing, any number of time differencing schemes may be employed to advance the solution in time.

4) Description of Predictor-Corrector Technique

a. Overview

The algorithm used in the FORTRAN program used for this research (Appendix A) uses a predictor-corrector scheme and spatial differencing treated as a finite volume model, as shown in Fig. 1. The finite difference equation for the predictor-corrector scheme is given by Eqs. 24 - 28.

Predictor Step:

$$\overline{U_i^{n+1}} = U_i^{n} - \frac{\Delta t}{\Delta x\, A_{i-1/2}}\left(\nabla_x F_i^{+} A_{i+1/2} + \Delta_x F_i^{-} A_{i-1/2} - H_i \nabla_x A_{i+1/2}\right) + \Delta t Q_i \tag{24}$$

Corrector Step:

$$\overline{\overline{U_i^{n+1}}} = U_i^{n} - \frac{\Delta t}{\Delta x\, A_{i+1/2}}\left(\nabla_x \overline{F_i^{+}} A_{i+1/2} + \Delta_x \overline{F_i^{-}} A_{i-1/2} - \overline{H_i} \nabla_x A_{i+1/2}\right) + \Delta t \overline{Q_i} \tag{25}$$

$$U_i^{n+1} = \frac{1}{2}\left(\overline{\overline{U_i^{n+1}}} + \overline{U_i^{n+1}} \right) \quad (26)$$

$$\nabla_x Q_i = Q_i - Q_{i-1} \text{ Backward Difference Operator} \quad (27)$$

$$\Delta_x Q_i = Q_{i+1} - Q_i \text{ Forward Difference Operator} \quad (28)$$

The "barred" quantities are the "predicted" values of the flow variables. The "double barred" quanities are the "corrected" values and are calculated on the basis of the "predicted" values. The net change in the U vector for a single time step is the average of the change in U for the predictor step and the change in U for the corrector step due to the averaging of the "predicted" and "corrected" values. The Q term is a generalized heat addition per cell, per unit time. This term is defined by the type of heat addition used and is specific to that type. A specific Q term is defined in Chapter V for use in the flow modeled in that chapter.

The fluxes through any surface are calculated in terms of the eigenvalues of the finite volumes (cells) on one side of the surface. This prevents the fluxes at any particular surface being determined by more than three characteristics. The cell used to determine the eigenvalues is alternated between the two sides of the surface on prediction and correction steps to prevent a

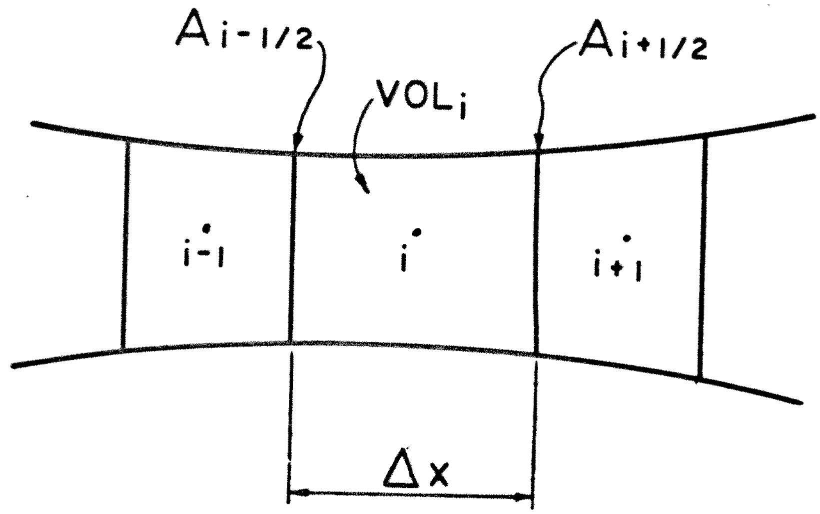


Figure 1. Finite Volume Model

numerically preferred direction. The technique is also strictly conservative in that a flux out of a cell on one side of a surface is the flux into the cell on the other.

b. Accuracy

The numerical model can be shown to be second order accurate in time and first order accurate in space. A simple inspection of the each element of the algorithm verifies this result. The time differencing is a predictor-corrector scheme, in itself second order accurate. The spatial derivatives are modeled by first differences, well understood as first order accurate. The rigorous analysis of the numerical accuracy of the algorithm is beyond the scope of this work. Instead the algorithm is checked against two known solutions to gasdynamic problems (Chapter IV).

c. Stability

Linear stability analysis of the predictor-corrector flux-split finite difference equations yields the stability condition of Eq. 29.

$$1 \geq \frac{\Delta t}{\Delta x \, |u+c|} \tag{29}$$

The right hand group in Eq. 29 is also known as the CFL number. The value of 1.0 for the CFL number is only neutrally stable and, thus, for actual calculations is never used. Typical values for an upper bound on the CFL number are on the order of 0.9. This value is used for all calculations in Chapters IV and V. The actual time step is the Δt such that Eq. 30 is satisfied for every cell.

$$CFL_{max} \geq \frac{\Delta t}{\Delta x \; |u+c|} \qquad (30)$$

A local CFL number less than one introduces numerical dissipation and diffusion. These effects can be shown to broaden the numerical representation of shock waves, cause sharp material boundaries, such as contact surfaces, to average and diffuse, and increase the entropy of the fluid, thus raising temperature and energy levels in the fluid. When analysing CFD results it is important to note these effects in order to correctly interpret the results.

IV. VERIFICATION OF NUMERICAL METHOD

A. TEST GASDYNAMIC PROBLEMS

The rigorous theoretical analysis of the numerical accuracy of the CFD technique employed here is beyond the scope of this paper. Instead the algorithm is checked against two known solutions. The two test problems chosen are: 1) Riemann's problem; 2) a shock-wave area-discontinuity interaction. The first problem is chosen to verify the ability of the computer program to correctly follow the transient response of various one-dimensional gasdynamic phenomena. The second is chosen to check the accuracy and ability of the program to model the area-change terms in Eq. 20a.

This type of verification has two major advantages over theoretical analysis. First, it verifies that the computer program accurately models the intended equations. Theoretical analysis only yields information on the the behavior of a particular algorithm if the algorithm is programmed into the machine correctly. Benchmark testing is the only way to truly verify any given computer program. Second, it identifies not only accuracy in general, but also yields information on which particular physical phenomena (such as normal shocks, contact surfaces, and expansion fans) are best modeled by the technique and which are not represented as well. This information can help build a family of solution techniques for various problems with greater insight as to which technique will yield the best solution for a given problem.

B. RIEMANN'S PROBLEM

Riemann's problem (Shapiro 1954, pp. 1007-1009), also known as the shock tube problem, involves a long tube divided into two sections by a diaphragm (Fig. 2.). The left-hand side of the tube is filled with high pressure gas; the left with low pressure gas. In our test example, the gas has the same temperature and sound speed on each side of the diaphragm. When the diaphragm is burst, a right moving shock propagates into region 1 and an insentropic expansion fan propagates into region 3. The pressure ratio across the shock is determined by an iterative solution to Eq. 31.

$$\frac{P_1}{P_3} = \frac{P_1}{P_2}\left[1 - \frac{\gamma - 1}{2\gamma}\,\frac{\dfrac{P_2}{P_1} - 1}{\sqrt{1 + \dfrac{\gamma + 1}{2\gamma}\left(\dfrac{P_2}{P_1} - 1\right)}}\right] \qquad (31)$$

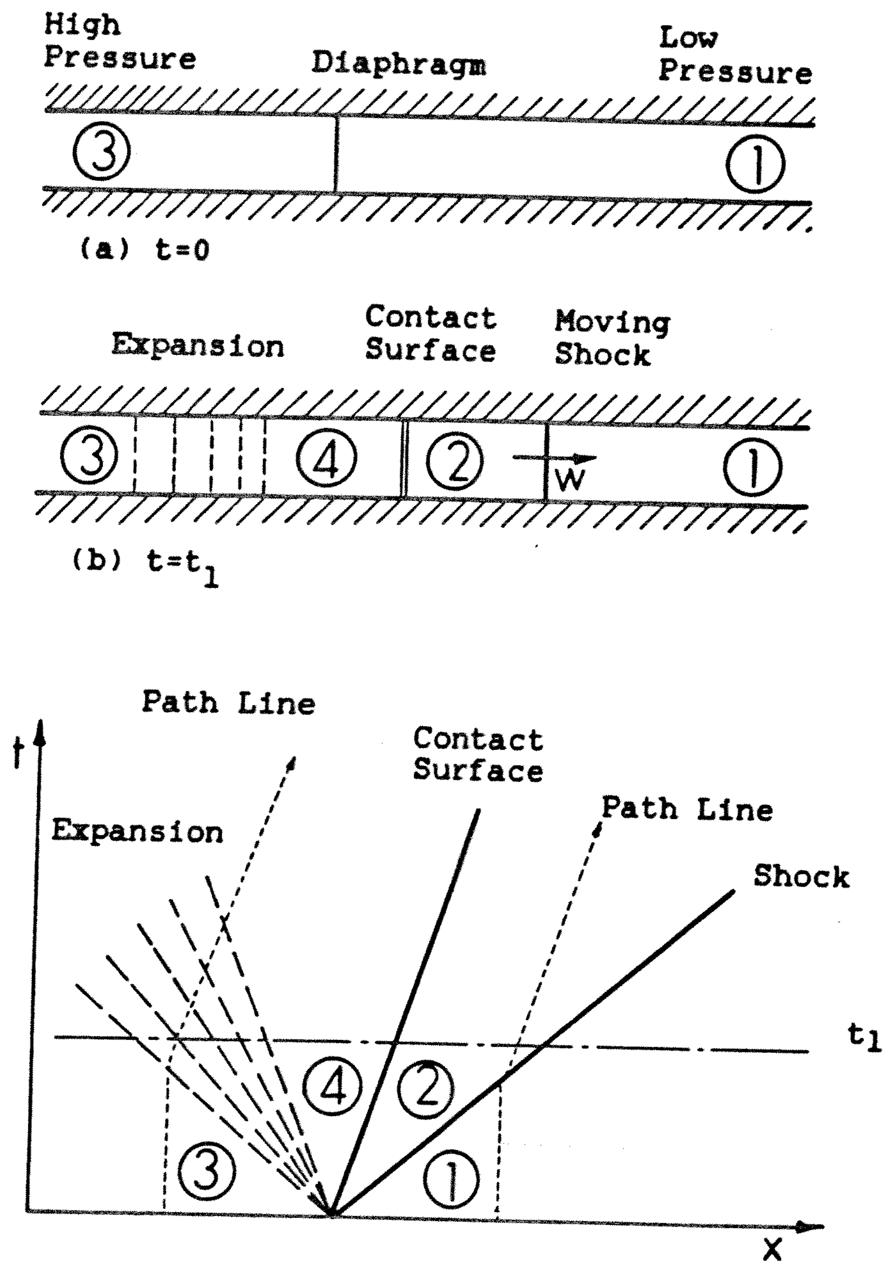
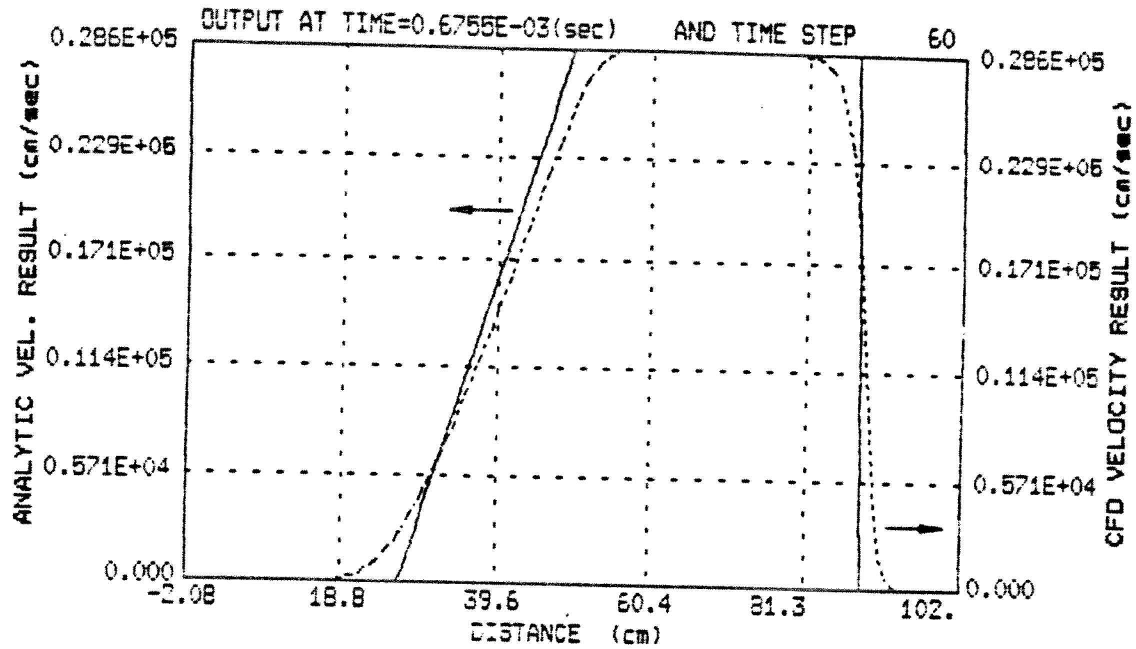


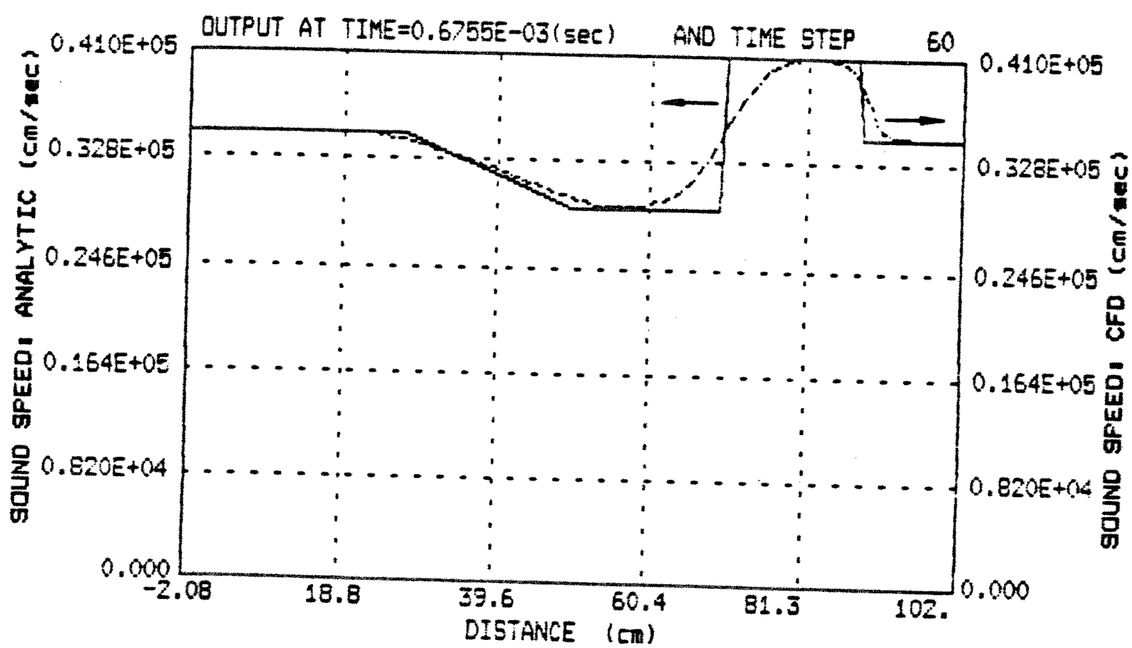
Figure 2. The Shock Tube Problem
(a) Configuration at $t = 0$
(b) Configuration at $t = t_1$
(c) Physical plane, showing shock wave expansion wave, contact surface, and path lines.

The remainder of the flow variables are determined by the appropriate use of the moving shock equations (Shapiro 1954, pp. 1000-1002) and the isentropic flow and normal shock relations given in Chapter III.

The performance of the CFD program is presented in Fig. 3. The intial pressure ratio of the shock tube problem shown is 10. It is apparent from Fig. 3 that the program follows the expansion fan and shock speed very well, and represents the shock with reasonable steepness. A typical value for the shock width is 10 cells. The maximum error for the sound speed in the expansion fan is 2.0%. The velocities and values of sound speed in the constant value regions vary by a maximum of 0.7% for velocity and 0.5% for sound speed from their respective analytic values. However, the ability to model the contact surface between the shocked and the expanded gas is severely limited, with errors on the order of 10%. This limitation is due to numerical averaging of the cell in which the contact surface should exist discretely. This numerical diffusion or mixing may be eliminated by use of a "sliding grid" tranformation (Ribe, Christiansen, and MacCormack 1983, pp. 3-5)[7] in which cell boundaries move with material boundaries. For the flow investigated in Chapter V this is not an important limitation as no material boundaries or contact surfaces exist within the flow.



(a) Velocity Result Comparison



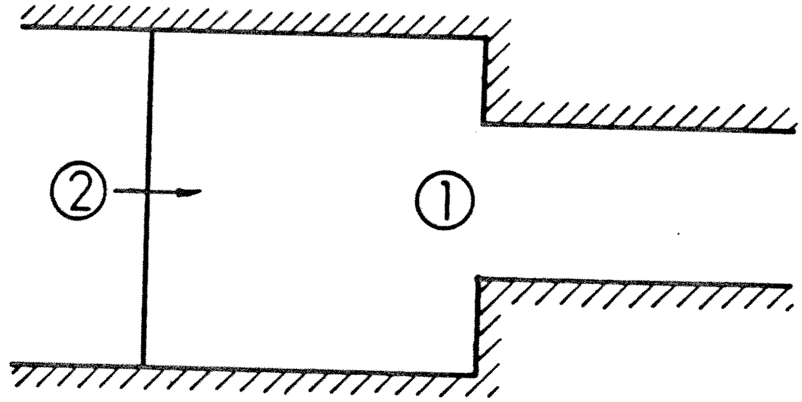
(b) Sound Speed Result Comparison

Figure 3. Comparison of Numerical and Analytic Results for Riemann's Problem

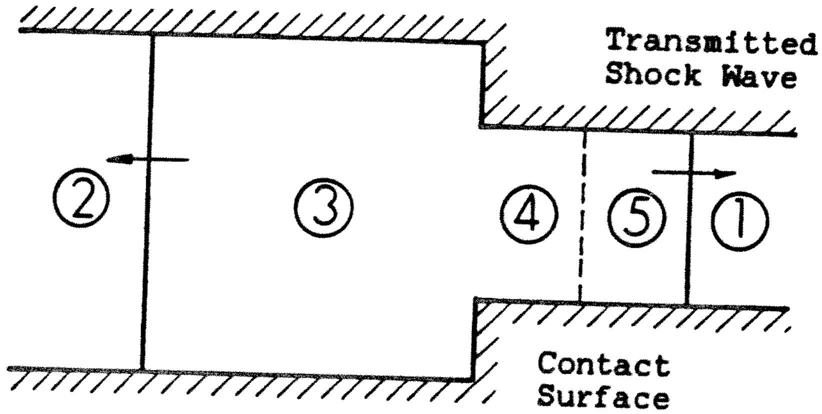
C. SHOCK-WAVE AREA-DISCONTINUITY PROBLEM

The second test problem is the interaction between a shock wave and a discontinous area change. Since the Riemann problem is for a constant area tube, a test of the area-change effects is required. The area-change problem (Shapiro 1954, pp. 1026-1027) involves a normal shock wave propagating into still air and encountering a discrete change in the area profile (Fig. 4). A "transmitted" shock continues to propagate into the region of reduced area, while a "reflected" shock propagates upstream into the previously shocked flow. Good performance in this test gives a good indication of the ability of the code to handle area-change effects in general. The analytic solution to this problem requires extensive use of the moving shock relations, and an iterative technique, the details of which are outside the focus of this paper.

Fig. 5 shows the computational result for this problem. Table 1 shows the comparison of computational and analytic solutions to the area discontinuity problem. Results are given for two different cases. For the first, the area change is modeled by two cells, for the second the area change is modeled by five cells. Errors in general are small, on the order of a few percent, reasonable for the first order accurate area-change terms. The contact

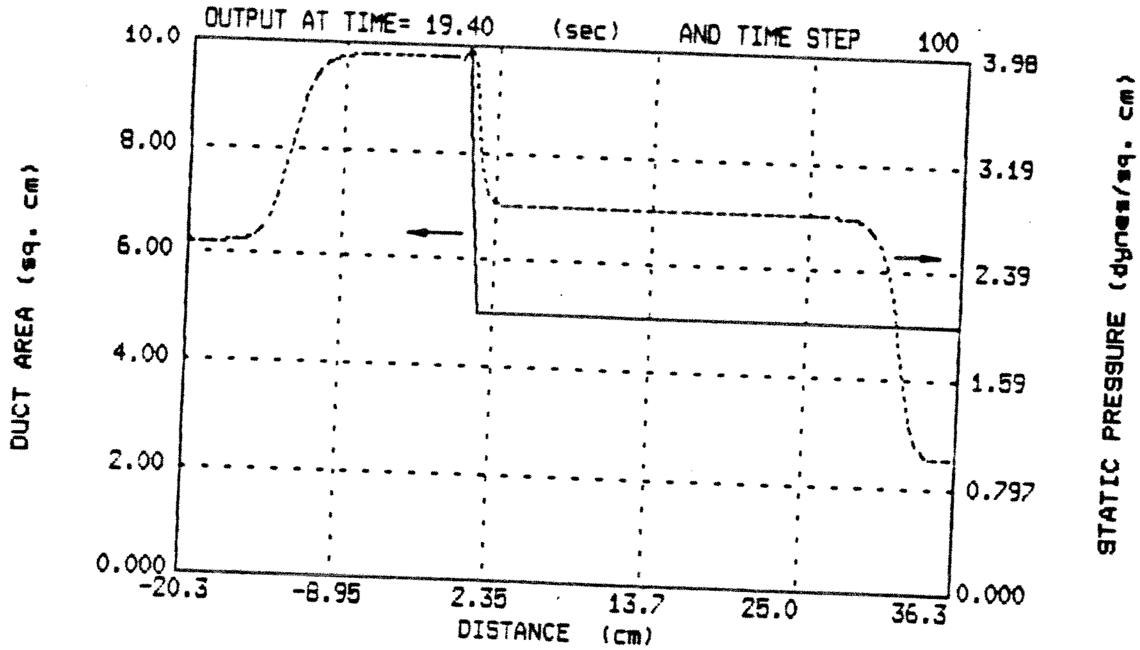


(a) Configuration Prior To Shock-Wave Area-Discontinuity Interaction

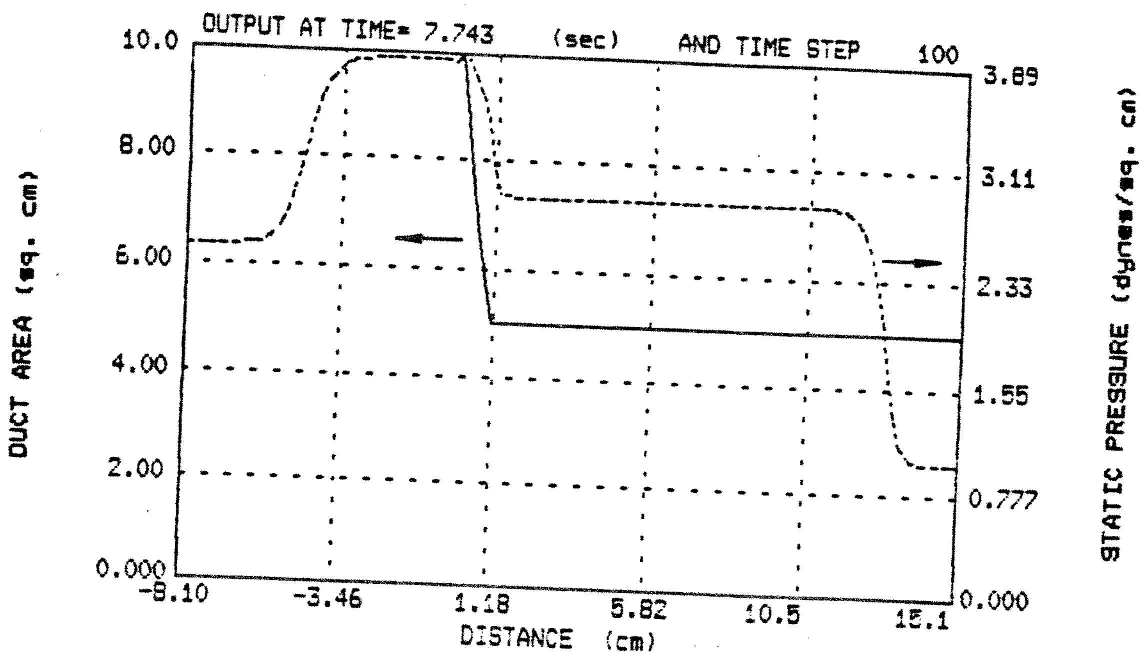


(b) Configuration After To Shock-Wave Area-Discontinuity Interaction

Figure 4. Area Discontinuity Problem



(a) Pressure And Area (Two-Cell Model)



(b) Pressure And Area (Five-Cell Model)

Figure 5. Numerical Results for Area Discontinuity Problem

TABLE 1

COMPARISON OF ANALYTIC AND NUMERICAL RESULTS TO THE SHOCK-WAVE AREA-DISCONTINUITY PROBLEM

Region	Pressure Error %	Density Error %	Sound Speed Error %	Shock Speed Error %
Two-Cell Model				
1	exact	exact	exact	0.9
2	< 0.1	< 0.1	< 0.1	2.7
3	3.1	< 0.1	0.7	-
4	3.0	4.5	0.6	-
5	3.0	4.5	0.6	-
Five-Cell Model				
1	exact	exact	exact	6.9
2	< 0.2	< 0.1	< 0.01	1.0
3	1.6	1.1	0.7	-
4	< 0.2	1.0	< 0.3	-
5	< 0.2	1.0	< 0.3	-

surface between regions 4 and 5 does not appear in the CFD result. The change in sound speed across this contact surface is substantially less than the error of the approximation, thus this feature is lost.

As Table 1 shows, the values of flow variables within the constant value regions vary from the respective analytical values by 0.1% to 6.0%. The shock speed for the two-cell model is accurate to within 2.7% for both the transmitted and reflected shocks. The shock speed for the transmitted shock for the five cell model is significantly different, an error on the order of 6.9%. There are two major reasons for this discrepancy. First, the area change is now significantly different from discontinous, and second, the shock speed is calculated by a simple time difference of shock location from two output times. This technique itself is only first order accurate and uses a large value of Δt. The shocks are represented over approximately eleven cells for both transmitted and reflected shocks for both cases.

D. CONCLUSIONS

The two problems of this chapter comprise a good test of the ability the code to solve the quasi-one-dimensional Euler equations. The computational results compare favorably with the analytic solutions. The exception to this is the contact surface which spreads unacceptably due to numerical diffusion and mixing. The problem addressed in Chapter V does not contain this feature. The algorithm, however, is shown to correctly model the moving shock, expansion, and area-change effects. Thus the results of Chapter V can be viewed with reasonable certainty, and accuracy to within a few percent.

V. APPLICATION OF NUMERICAL METHOD TO TRANSIENT RESPONSE PROBLEM

A. PHYSICAL CONFIGURATION OF DEVICE MODELED

A complete description of a generalized ramjet requires a large number of unknowns varying in almost limitless combination. To provide preliminary understanding of the time-dependent response of the device a single geometry is chosen. As seen the Fig. 6, the device investigated in this chapter consists of steady, supersonic, upstream flow encountering a convergent-divergent nozzle. The area at the nozzle throat is 40% of that of the inlet. Following the nozzle section is the zone in which heat is released. The heat release is such that in steady flow the duct is thermally choked.

With only limited prior knowledge of the transient response to combustion inititiation, the downstream boundary condition must be able to correctly model both supersonic and subsonic exit conditions. For the purposes of the model, no pressure waves are allowed to propagate upstream from beyond the downstream boundary. This condition is called the "no reflection condition." This is automatically true if the Mach number at the exit is

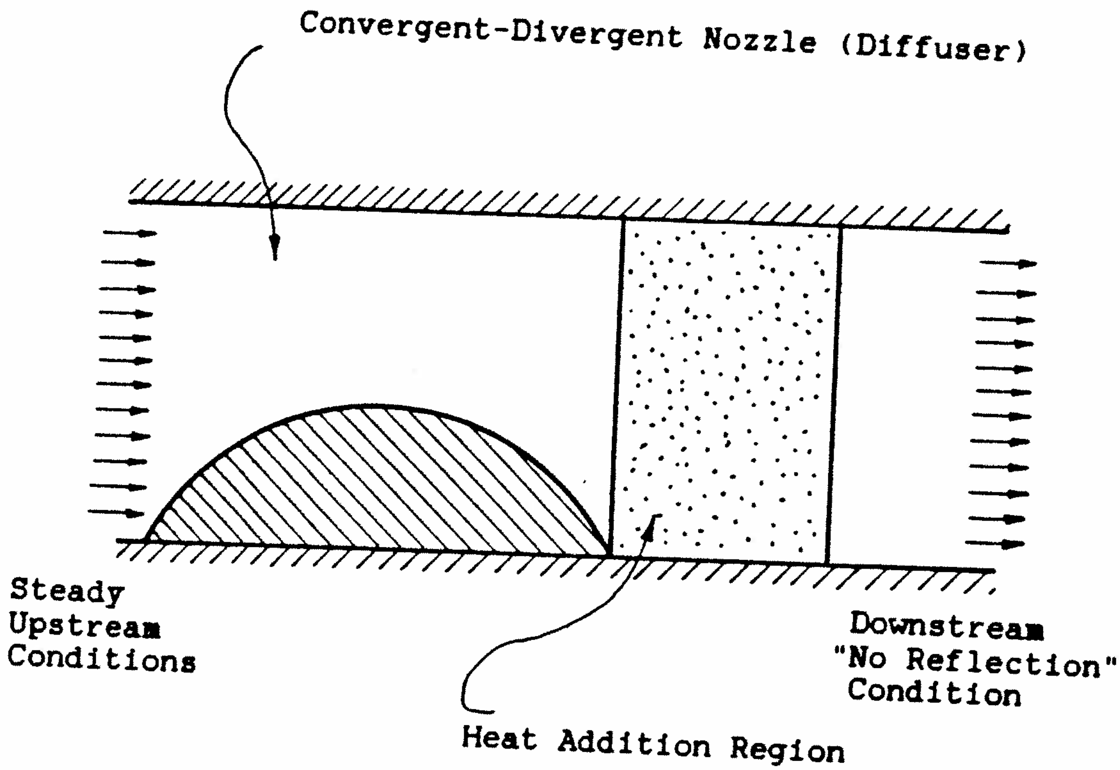


Figure 6. Configuration of Modeled Device

greater than unity, and enforced for Mach numbers less than unity.

Although the device investigated in this chapter has a fixed geometry, the area profile, the gas characteristics, and the inlet velocity are all input variables to the CFD program.

B. HEAT ADDITION MODEL

1) Definition of the Q Term

The heat addition is modeled by an idealization of the combustion process. The Q term from Eq. 24 and Eq. 25 is constructed by multiplying the heat release per unit mass by the incoming mass flow, which is constant. This number is divided by the total volume of the combustion section to yield the energy per unit volume per cell per time step. This value is added as a source term to the energy equation by means of the Q term. The value of Q is set to zero for all cells outside the combustion region.

2) Ignition and Chemical Composition

An extensive model of the ignition process is outside the focus of this work. For this analysis, combustion is modeled as beginning in a single cell and propagating at the local flow velocity to a fixed downstream boundary of the heat addition region. At the flow velocities of this analysis, however, the time required for this ignition process is small in comparison to the time for shock movement, thus the effect on the transient is considered to be small. The chemical composition of the gas is assumed to remain constant throughout the combustion process. The molecular weight and γ are taken to be the average of the actual values before and after combustion.

C. STABILITY OF AREA CHANGE TERMS

1) Numerical Unstart

The effect of numerical dissipation plays an important role in determining the finite difference grid used for a particular configuration. The dissipative terms cause both an increase in temperature of the fluid and a decrease in the flow velocity, both yielding a decrease in local Mach number. This is especially important at the throat of the nozzle where the Mach number may already be near unity. For coarse grids, this effect may yield the numerical result of a throat Mach number less than unity. This causes a numerical instability in the form of a "spontaneous unstart" for configurations whose ideal quasi-one-dimensional result is stable supersonic flow.

2) Area Profiles

Area profiles may be classified into three major groups, log profiles, linear profiles, and Mangler tranformed profiles. The log profile is defined as one in which the area of the next grid point is a constant fraction (or multiple) of the previous one. This profile tends to be both the least dissipative and the least similar to real devices. The linear area profile is one for which the area is a linear function of distance along the device. The Mangler transformed area profiles correspond to the quadratic functions describing the area profile of a cylindrical duct with a conical center-body. For all of these types of area profiles, the function or constant may change at the throat of the device for convergent-divergent nozzles. For each of the last two profiles, special attention is required at the sharp corner produced at the throat by the functions used. A thorough understanding of the stability characteristics of the various area profiles yields a family of area profiles suitable to model different device configurations accurately, stably, and efficiently. The area profiles used for this analysis are log profiles, both for their stability and to allow minimum grid sizes for maximum computational efficiency.

D. THE HEAT ADDITION RATIO (HR)

For any particular set of upstream conditions and working fluid, a maximum steady-state heat addition per unit mass ΔH_M is uniquely defined by the steady-state ramjet equations given in Chapter III. A flow configuration is quantified by the ratio of the actual heat release per unit mass for the combustible mixture ΔH_A divided by the maximum steady-state heat addition ΔH_M. This ratio is called the heat addition ratio (HR) defined by Eq. 32.

$$HR = \frac{\Delta H_A}{\Delta H_M} \tag{32}$$

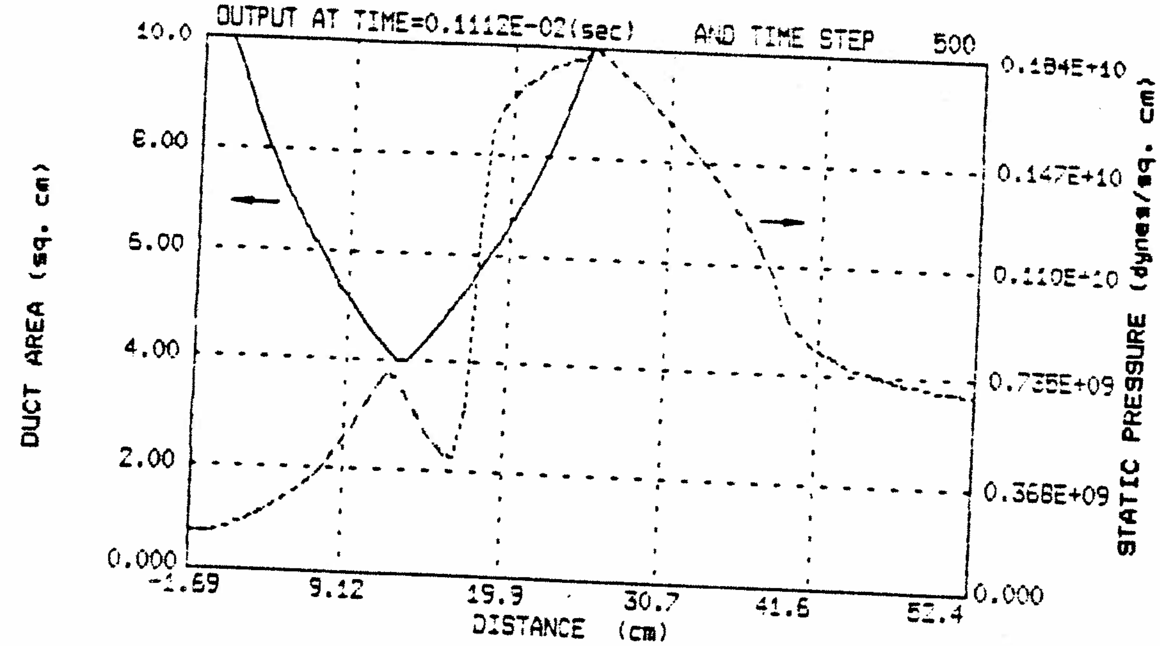
A value of unity for HR is only neutrally stable even for the steady-state case. For analysis of the transient response several values of HR less than unity are investigated to find stable configurations. HR values greater than unity are investigated to study the response of strongly unstable configurations.

E. TYPICAL TRANSIENT RESPONSE

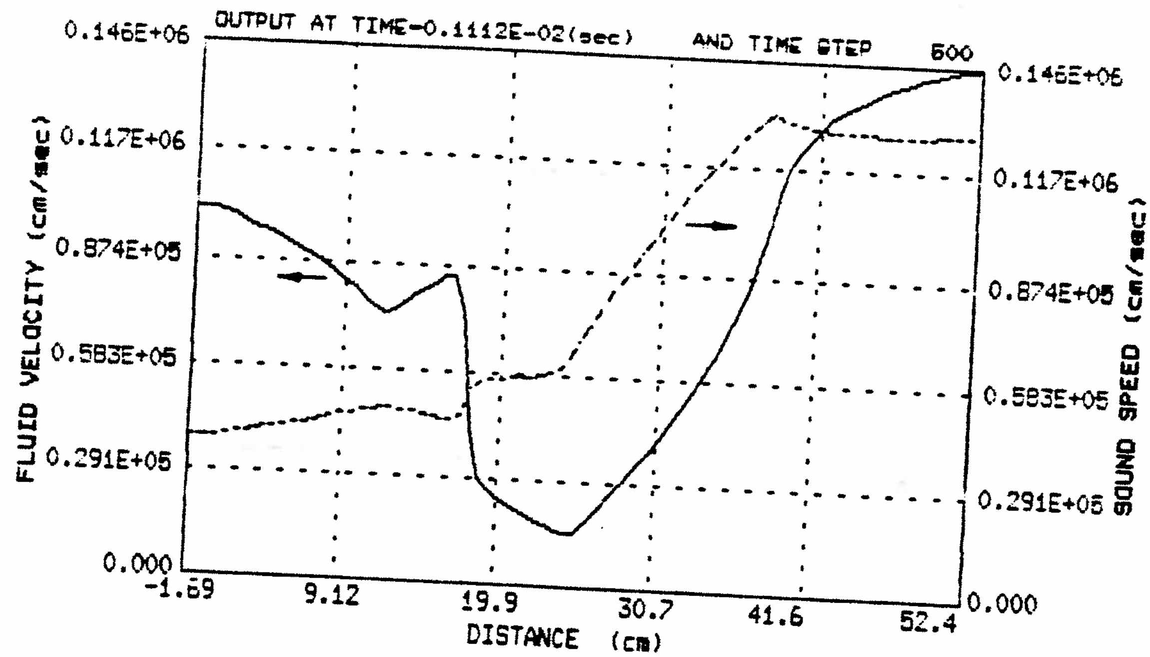
As noted above, the transient response may be classified as stable or unstable. The numerical results clearly show this behavior and follow the qualitative description of Chapter II. The results from various working fluids are sufficiently similar to allow presentation of a single "typical" response for stable and unstable cases. These typical results are given by Figs. 7 - 12. Shock strength and speed, final shock location (for stable configurations), and characteristic time for the transient response may all vary for different configurations. The essential characteristics of the response are, however, the same. The typical responses shown use the air-hydrogen mixture defined below. The stable result is for HR = 0.8. The HR value for the unstable result is 1.0.

Figs. 7 - 9 show the typical stable transient response for the modeled device. Each figure contains graphical output for duct area, static pressure, local flow velocity, and sound speed at a single time. The upper graph of each figure contains the duct area and pressure information, which shows the shock wave and the heat addition effects clearly. The lower graph shows velocity and sound speed, which are displayed together to show the shock and choke point more clearly than if they were presented separately. The three figures are the ouput at three different times, giving a "motion picture" representation of the transient.

The four quantities chosen for graphical display form

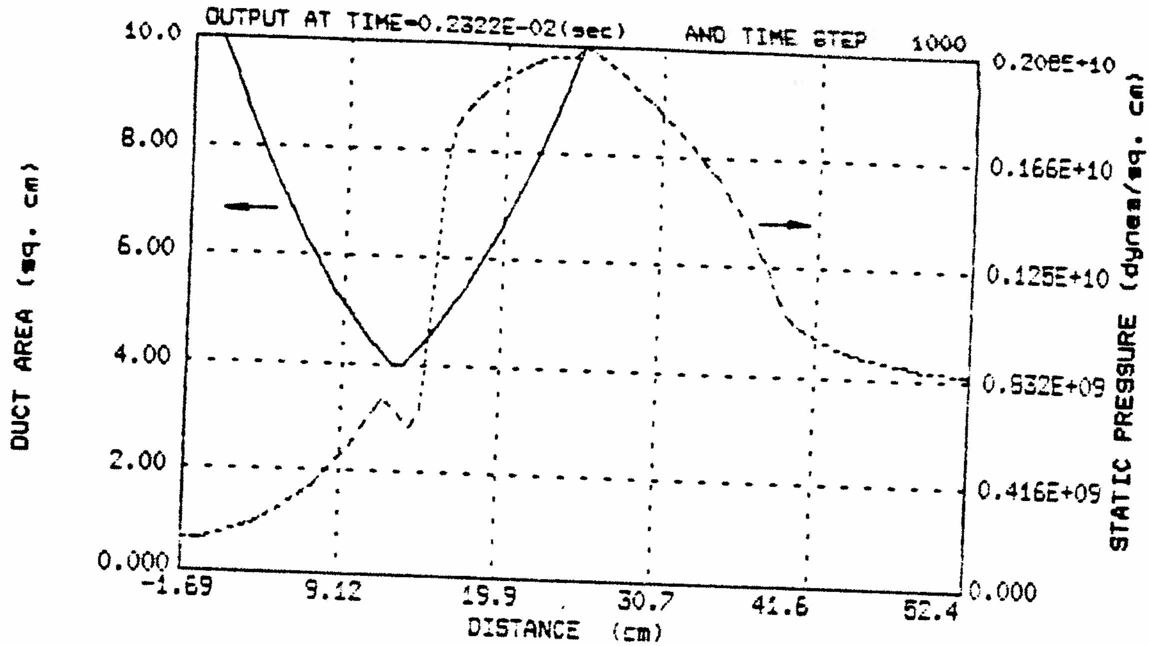


(a) Stable Transient Response: Pressure and Area

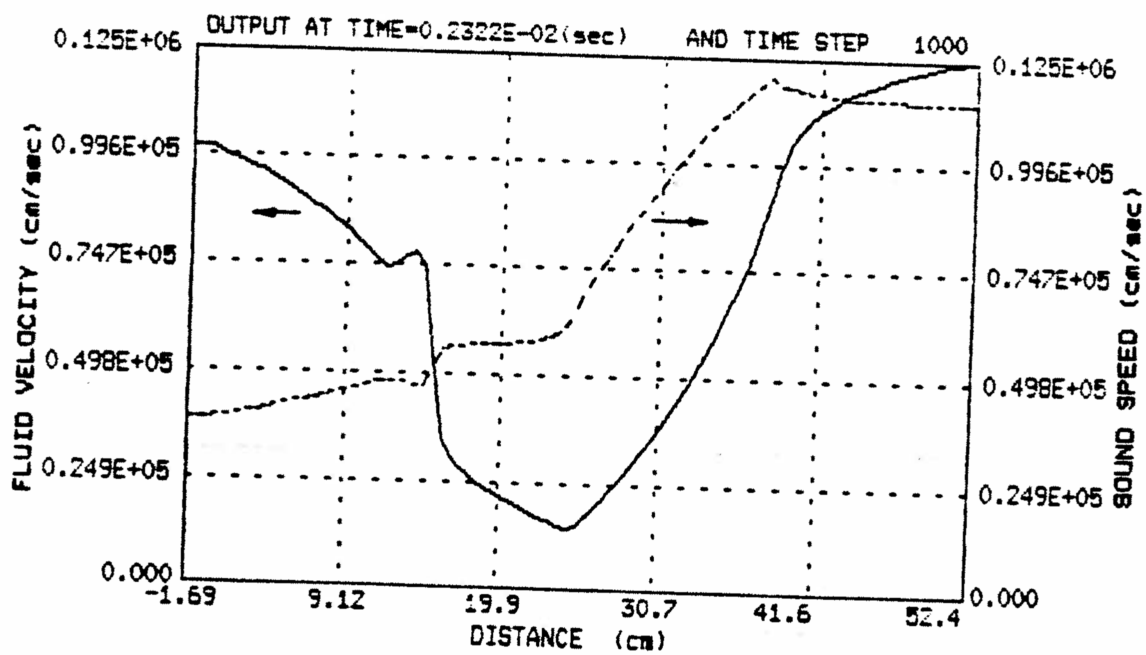


(b) Stable Transient Response: Velocity and Sound Speed

Figure 7. Typical Stable Response (Frame One)

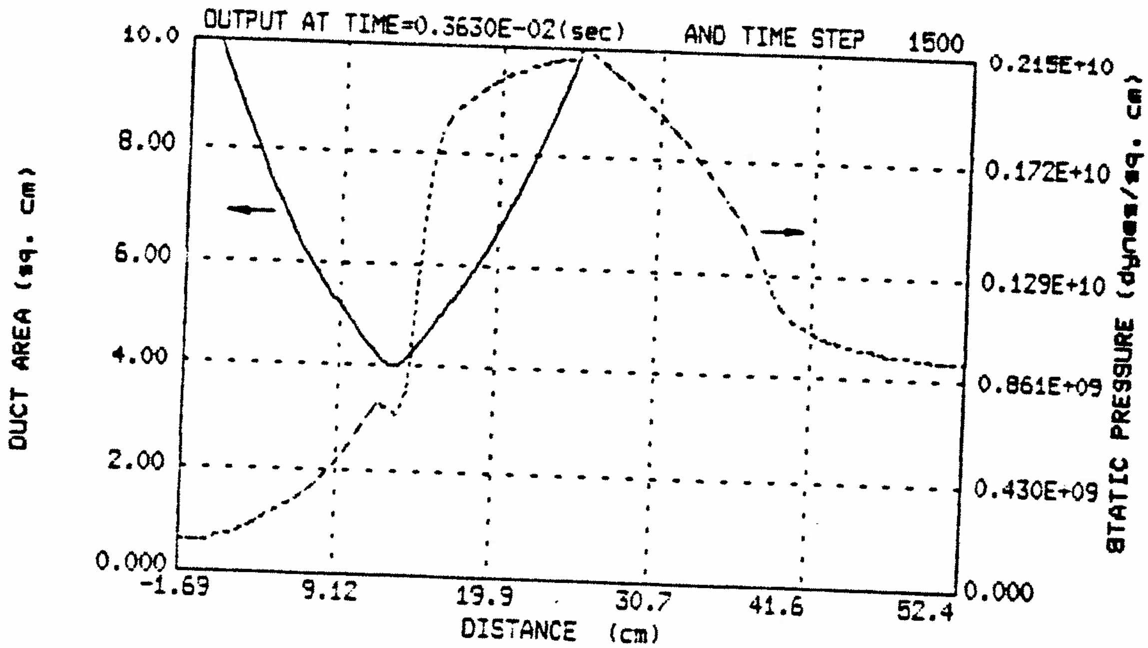


(a) Stable Transient Response: Pressure and Area

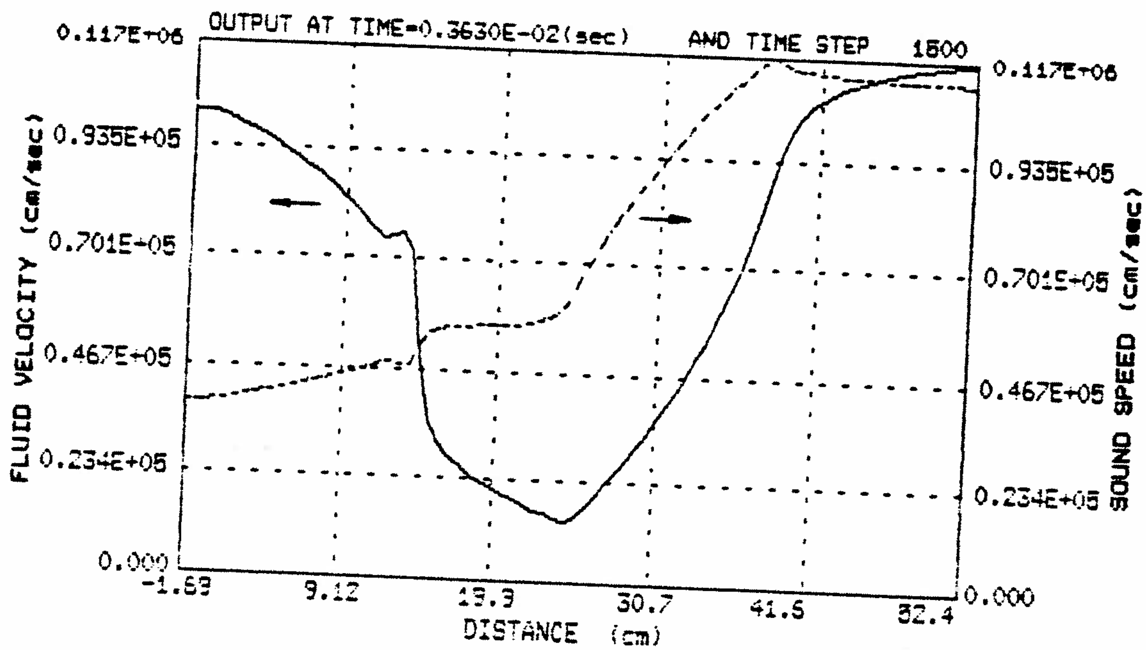


(b) Stable Transient Response: Velocity and Sound Speed

Figure 8. Typical Stable Response (Frame Two)

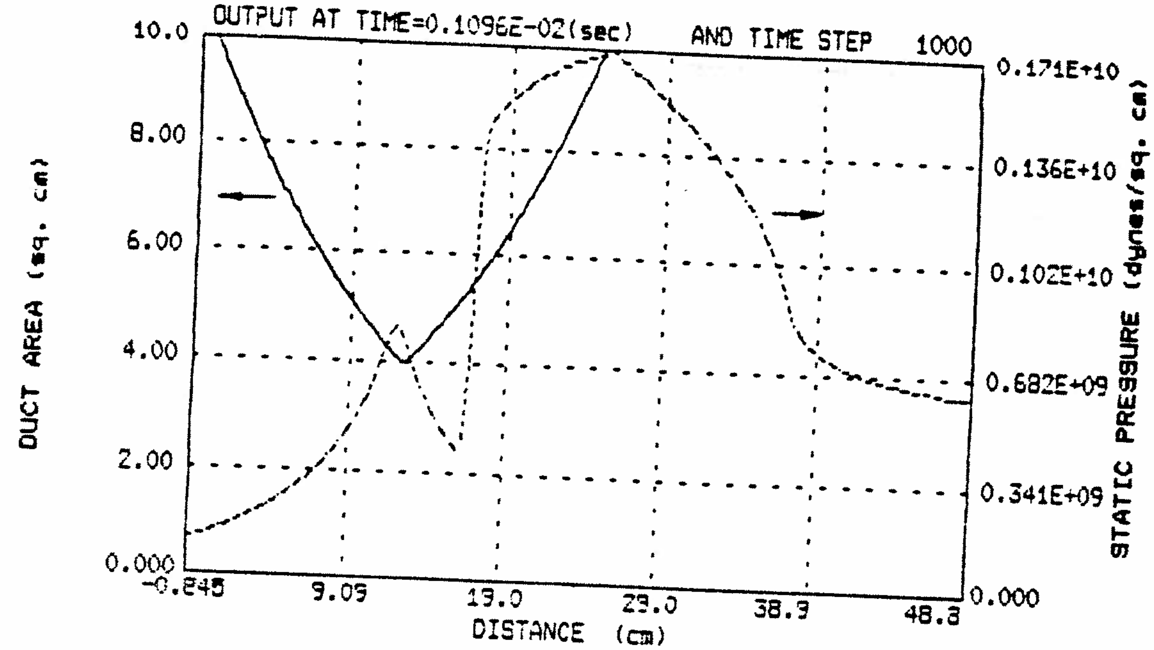


(a) Stable Transient Response: Pressure and Area

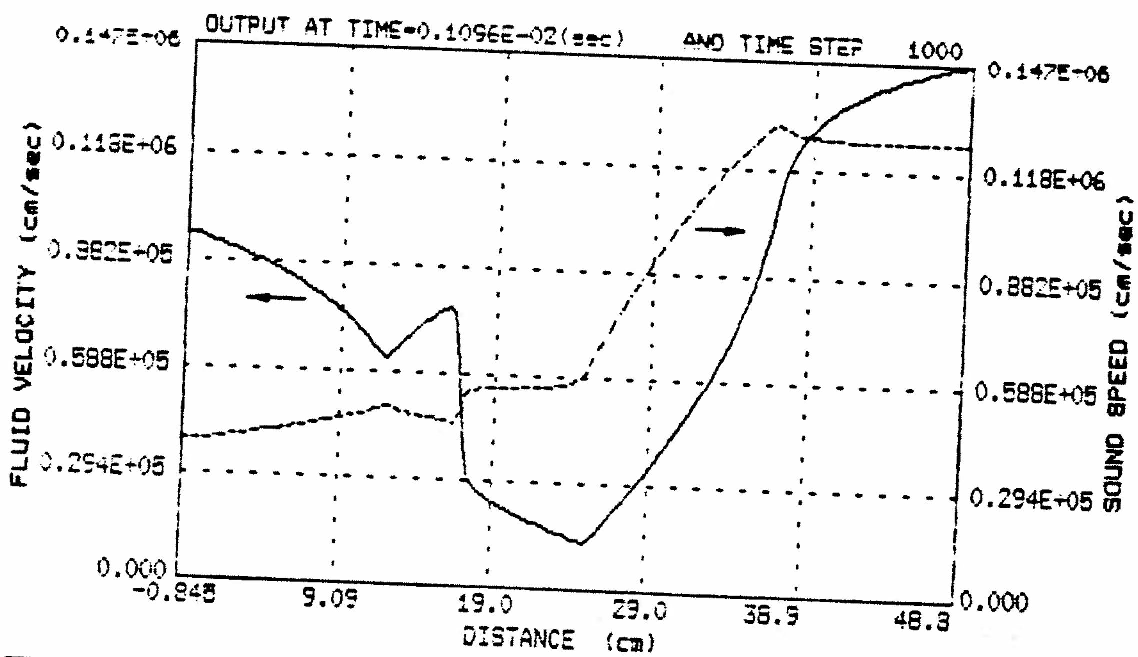


(b) Stable Transient Response: Velocity and Sound Speed

Figure 9. Typical Stable Response (Frame Three)

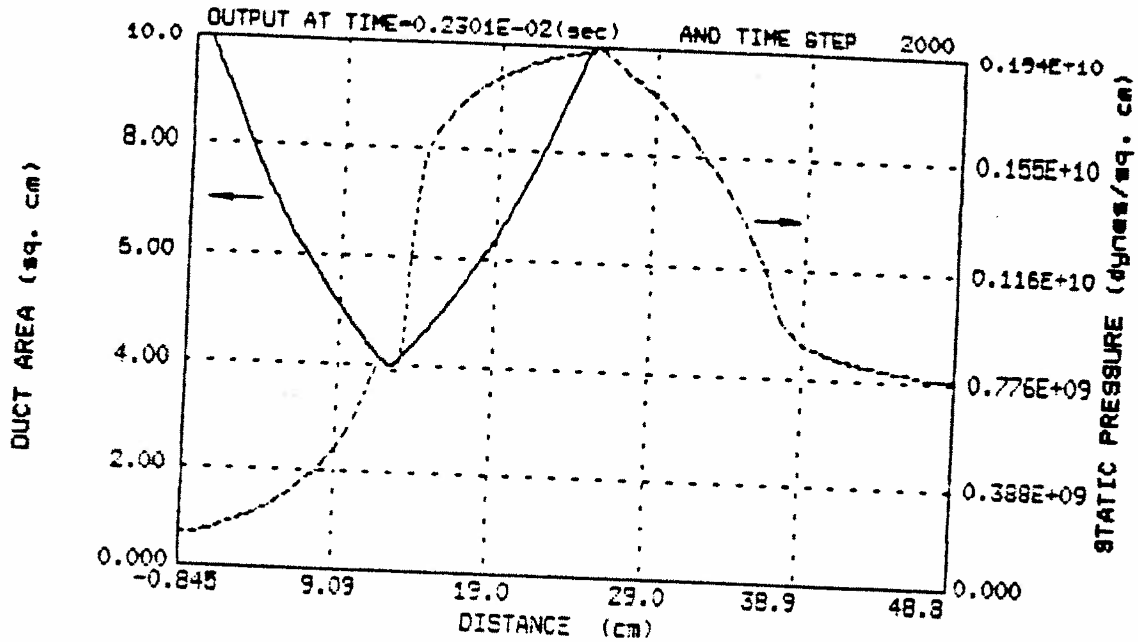


(a) Unstable Transient Response: Pressure and Area

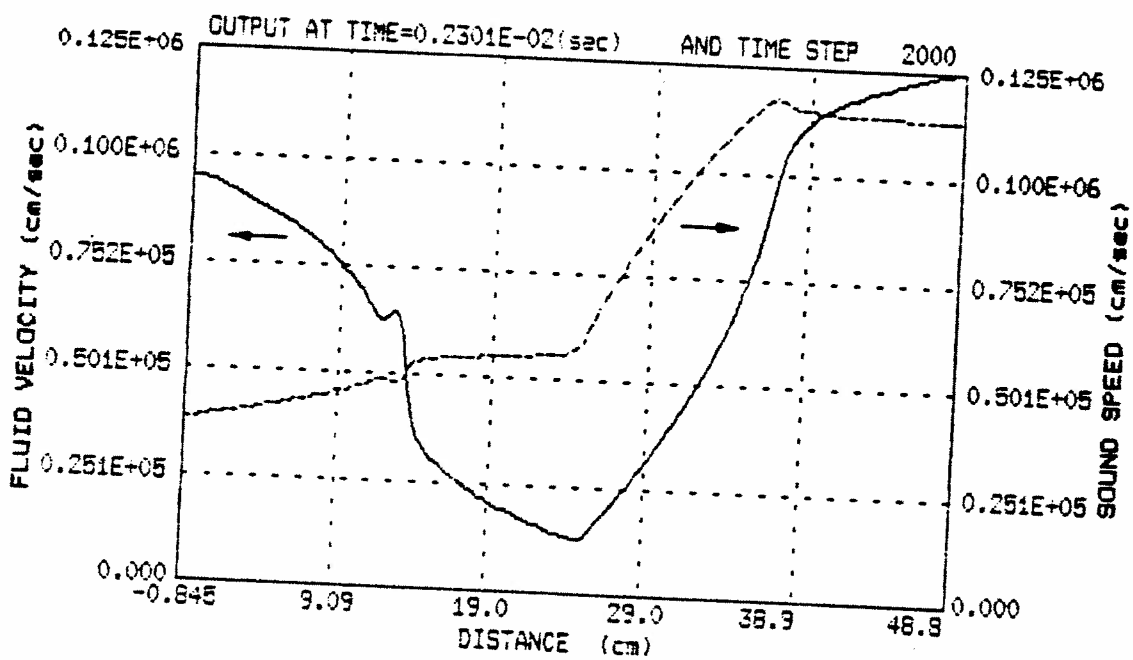


(b) Unstable Transient Response: Velocity and Sound Speed

<u>Figure 10. Typical Unstable Response (Frame One)</u>

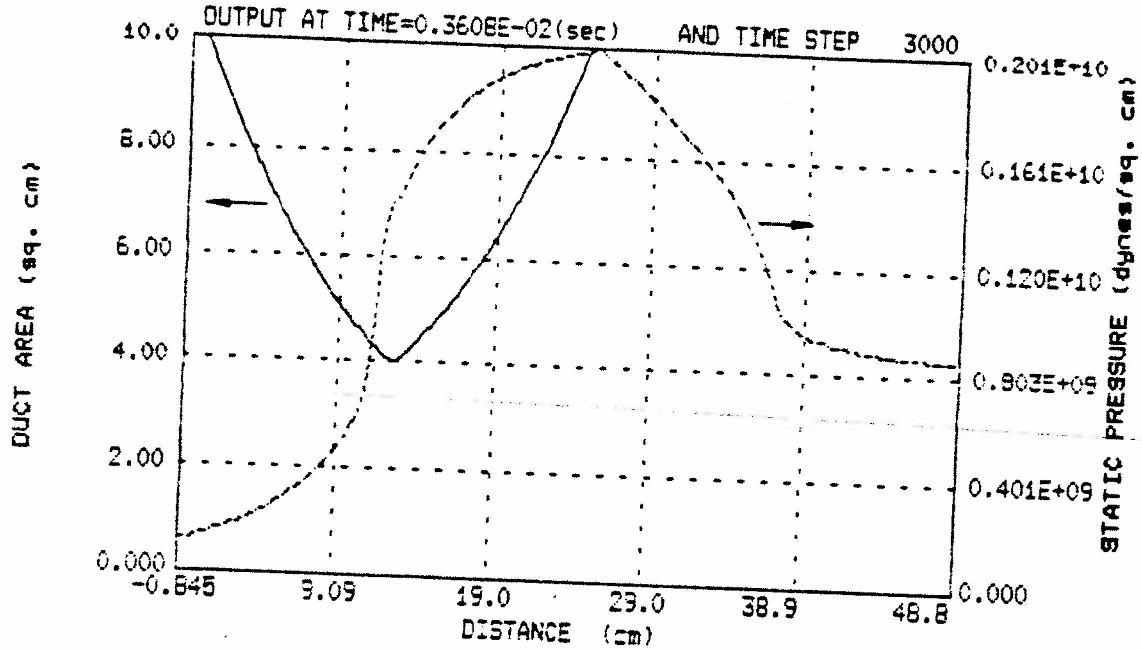


(a) Unstable Transient Response: Pressure and Area

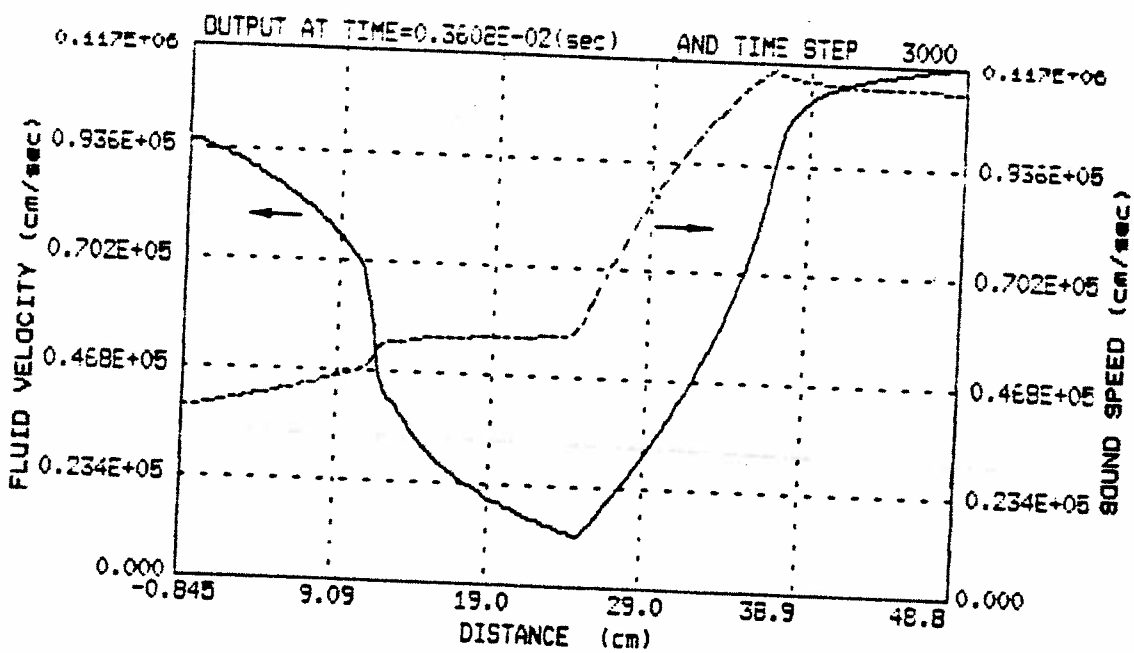


(b) Unstable Transient Response: Velocity and Sound Speed

Figure 11. Typical Unstable Response (Frame Two)



(a) Unstable Transient Response: Pressure and Area



(b) Unstable Transient Response: Velocity and Sound Speed

<u>**Figure 12. Typical Unstable Response (Frame Three)**</u>

a complete set of information about the flow at that particular time. The duct area is include to show the physical geometry of the convergent-divergent nozzle, the full scale, open duct, inlet area decreasing to the minimum throat value, and increasing back to the open duct area.

Fig. 7 shows the flow roughly a millisecond after the initiation of combustion. At this point the heat addition has attained the maximum prescribed value. Note that Fig. 7 shows the shock wave at the base of the nozzle. In Fig. 8 the shock is substantially into the nozzle, and Fig. 9 shows the shock just downstream of the nozzle in the steady-state location.

The typical unstable transient response shown in Figs. 10 - 12 is very similar to the stable response and the same graphical presentation is used. The significant difference between the two responses is seen in Fig. 12. In Fig. 12 the shock is upstream of the nozzle, causing unstart.

F. RESULTS FOR VARIOUS GAS MIXTURES

1) Definition of Gas Mixtures

The transient responses for three different stoichiometric gas mixtures, air-hydrogen, air-methane, and oxygen-methane, were investigated. As the fluid is assumed

TABLE 2

AVERAGED PHYSICAL AND CHEMICAL PROPERTIES FOR GAS MIXTURES

Gas Mixture	Molecular Weight	Gamma γ	Heat Release Per Unit Mass
Air-Hydrogen	22.90	1.35	3.23202×10^{11} erg/gm
Air-Methane	27.63	1.3196	2.64600×10^{10} erg/gm
Oxygen-Methane	26.66	1.2089	1.0055×10^{11} erg/gm
	Chemical Formula		
Air-Hydrogen	$O_2 + 4N_2 + 2H_2 \longrightarrow 2H_2O + 4N_2$		
Air-Methane	$2O_2 + 8N_2 + CH_4 \longrightarrow CO_2 + 2H_2O + 8N_2$		
Oxygen-Methane	$2O_2 + CH_4 \longrightarrow CO_2 + 2H_2O$		

both to combust and to have constant physical properties, values for both γ and molecular weight are the average of the initial and the combusted properites. The chemical formulas and physical properties of the three gas mixtures are given in Table 2.

2) Results for Various Configurations

The behavior of the transient responses for the three gas mixtures is given in Table 3 for several values of HR for each gas mixture. The HR values range from 0.80 to 1.10. This range is chosen to insure both stable and unstable transients. Inlet velocity is given for each case. The stability of a transient is determined by whether or not the solution settles to the steady-state result. Time for transient response is given to show the characteristic time required for the solution to unstart the diffuser or

TABLE 3

SUMMARY OF TRANSIENT RESPONSE BEHAVIOR
FOR VARIOUS CONFIGURATIONS

Gas Mixture	HR	Inlet Velocity (km/sec)	Stability	Time for Transient Response (sec)
Air-Hydrogen	0.80	1.01287	Stable	3.6×10^{-3}
	0.85	0.99476	Stable	4.9×10^{-3}
	0.90	0.97848	Stable	4.5×10^{-3}
	0.95	0.96427	Unstable	3.6×10^{-3}
	1.00	0.95229	Unstable	$< 3.6 \times 10^{-3}$
Air-Methane	0.80	0.89132	Stable	5.7×10^{-3}
	0.85	0.87591	Stable	5.7×10^{-3}
	0.90	0.86220	Stable	5.8×10^{-3}
	0.95	0.85020	Unstable	4.2×10^{-3}
	1.00	0.83967	Unstable	$< 4.2 \times 10^{-3}$
Oxygen-Methane	0.80	1.27848	Stable	3.5×10^{-3}
	0.85	1.24543	Stable	3.5×10^{-3}
	0.90	1.21595	Stable	3.5×10^{-3}
	0.95	1.18672	Stable	3.5×10^{-3}
	1.00	1.15779	Unstable	5.5×10^{-3}
	1.10	1.11657	Unstable	$< 3.5 \times 10^{-3}$

Conditions for all Configurations:

Inlet Pressure..... 1.295×10^{8} dynes/sq. cm
Inlet Temperature.......... 300 K
Inlet Area................ 10.0 sq. cm
Throat Area................ 4.0 sq. cm

settle into the steady-state result. A discussion of the implications of Table 3 is included in Chapter VI.

VI. CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

A. CONCLUSIONS FROM TRANSIENT RESPONSE ANALYSIS

The behavior of several combinations of working fluid and inlet velocity was presented in Chapter V. Each of the solutions gave results similar to those of the typical case which was presented. Graphical results similar to those in Figs. 7 - 12 were obtained for each of the configurations. Due to the high degree of similarity these graphical results are not presented here.

The summary of transient behavior indicates a simple working criterion for determining the stability of a configuration without complete numerical modeling. This stability criterion is based on the value of the heat addition ratio (HR). As shown in Table 3, for all cases but one, a value of HR = 0.95 yields an unstable transient response. For all cases, an HR value of 0.90 results in a stable transient. This result is of greater interest in light of the differences in working fluid and inlet velocity. This criterion is useful in the determination of stable operating velocites for a given working fluid, or may yield the best candidates for a working fluid at a given inlet velocity.

One further note on the stability of the transient response is that it is highly sensitive to the inlet velocity for a particular gas mixture. The velocity difference between the strongly stable transient of HR = 0.80 and the strongly unstable response of HR values of 1.00 or 1.10 is as small as 5.8%. This sensitivity to inlet velocity indicates that conservative values of HR are required to insure stable initital response. Also, further work to refine the accuracy of the numerical model is indicated by this result.

B. RECOMMENDATIONS FOR FURTHER WORK

The test problems of Chapter IV indicate that the numerical model is accurate to within a few percent. This meets the first order approximation criterion set in Chapter II for the accuracy of the model. As, such the model may be applied to various other configurations and other similar gasdynamic problems. Variation in upstream or downstream boundary conditions, acceleration of the device, alteration of the heat addition term, or the effects of variation in geometry, all may be investigated with this model.

With the sensitivity of the transient response to inlet velocity, and the desire to maximize the heat addition for certain gasdynamic devices, such as ramjets, the developement of a model of a higher order of accuracy for use in conjunction with the current technique is recommended. Various second order effects, such as the actual nature of the ignition process, the shock structure of the diffuser, and variation of working fluid properties due to combustion, should be included in this new model. Computational cost for the higher order of accuracy model may be at least twice that of the current model. Thus, initial analysis of any configuration should be performed by the numerical model presented in this work.

LIST OF REFERENCES

1. MacCormack, R. W. "Current Status of Numerical Solutions of the Navier - Stokes Equations." AIAA paper No. 85-0032, Reno, Nevada. Jan. 14, 1985 (Duplicated Copy).

2. Kuethe, A. M. Foundations of Aerodynamics: Bases of Aerodynamic Design 3rd ed. New York: John Wiley & Sons, Inc., 1976.

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

4. Oates, G. C. Aerothermodynamics of Gas Turbine and Rocket Propulsion. American Institute of Aeronautics and Astronautics Education Series, Vol. 2 (New York: American Institute of Aeronautics and Astronautics, 1984).

5. Conte, S. D. and deBoor, C. Elementary Numerical Analysis an Algorithmic Approach 3rd ed. New York: McGraw-Hill Book Co., 1980.

6. Steger, J. L. and Warming, R. F. "Flux Vector Splitting of the Inviscid Gasdynamics Equations with Application to Finite Difference Methods." NASA TM 78605, 1979.

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

APPENDIX A.

LISTING OF FORTRAN PROGRAM EMPLOYED FOR CURRENT RESEARCH

The following pages contain the listing of the computer program (GSTART) used for the actual numerical research. Included are both the main program and all subroutines. A sample input file is included at the end of the appendix. The program is written in PDP-11 FORTRAN-77 V5.0. The program was run on a Digital Equipment Corporation Profesional 380 microcomputer operating under the RSX-11M-PLUS V2.1 operating system. Several system dependant subroutines are called from various modules of the program especially the graphics routines. In this version of GSTART, all special initial conditions required for the two test problems of Chapter IV are included.

```
      PROGRAM GSMAIN
      INCLUDE 'GSCELDAT.INC'
      COMMON /RMN/PSX0,PSC1,PSC2,PSC3,PSC4,PSU2,PSW
      LOGICAL SHTU
      CHARACTER DIRECT*20,PREFIX*20,EXTEN*20
      CHARACTER*24 FILNM1,FILNM2
      CHARACTER*30 STARTM,FINITM
C
C  GET INITIAL TIME
C
      STARTM='                                  '
      FINITM='                                  '
      CALL TIME(STARTM)
      RUNTIM=SECNDS(0.0)
C
C  INITIALIZE INPUT AND OUTPUT FILES
C
C  OPEN INPUT FILE
      OPEN(UNIT=5,STATUS='OLD',NAME='[ZGSTART]GSDAT.DAT')
C  READ IN FILE NAME SPECIFICATION FOR OUTPUT
      READ(5,*)LD,LP,LE,IFILNM
      READ(5,4100)DIRECT
      READ(5,4100)PREFIX
      READ(5,4100)EXTEN
4100  FORMAT(A20)
      READ(5,*)SCREEN
C
C  INITIALIZE GRAPHICS  AND GRAPHIC OUTPUT FILES
C
      CALL FILINI(DIRECT,LD,PREFIX,LP,EXTEN,LE,IFILNM,IERRCD)
      CALL PLTINI(SCREEN)
C
C  INITIALIZE OUTPUT FILE
C
      CALL FILNEX(FILNM1,LEN,IERR)
      FILNM1(21:23)='OUT'
      OPEN(UNIT=6,STATUS='NEW',NAME=FILNM1)
C
C  CALL INPUT AND INITIALIZE VALUES SUBROUTINE
C
      CALL IAND1(U1,P1,TEMP1,NSK,NTO,TSET,NSET,PREFIX,QDOTVO,
     1           SHTU)
C INITIALIZE COUNTERS AND POINTERS
      JI=0
      KCC=1
      N=1
      M=1
      M1=2
      T=0.
      DNETA=1./KK
```

```
C USE ISENTROPIC EXPANSION FOR INITIAL CONDITIONS AND OUTPUT
      CALL ISNINI(U1,P1,TEMP1)
      CALL PRINT1(T,JI,PREFIX)
C
C SET SETTLING TIME
      IF (TSET.LT.0.0) THEN
         CALL TYME
         TSET=NSET*DT
      ENDIF
      IBRNED=NF+NP+NB
      XCZONE=XG(NF+NP+2-NPB)
      XBRNPT=XCZONE
      IBRNPT=NF+NP+1-NPB
C  SET TIMING VARIABLE
      TDIF1=SECNDS(0.0)
C *****  300 BLOCK
C  MAIN LOOP
C *****
  300 CONTINUE
        IF (T.GE.TSET) THEN
           IF ((XBRNPT.GE.XG(IBRNPT+1)).AND.(IBRNPT.LE.IBRNED)) THEN
              IBRNPT=IBRNPT+1
              QDOT(IBRNPT)=QDOTVO
           ENDIF
        ELSE
           XBRNPT=XCZONE
        ENDIF
        CALL TYME
        JI=JI+1
        CALL FLUXPR
        CALL FLUXCO
        XBRNPT=XBRNPT+RHOUG(IBRNPT,M)/RG(IBRNPT,M)*DT
        T=T+DT
        IF(JI.EQ.(NSK*KCC))  THEN
           CALL PRINT1(T,JI,PREFIX)
         TDIF2=SECNDS(0.0)
         TDIF3=TDIF2-TDIF1
         TDIF3=TDIF3/FLOAT(NSK)
         TDIF1=TDIF2
        WRITE(6,*) ' AVERAGE TIME FOR ONE ITERATION: ',TDIF3
        CALL FILNEX(FILNM1,IFL1,IERR)
        CALL FILNEX(FILNM2,IFL2,IERR)
        WRITE(6,*) ' GRAPHIC OUTPUT IN FILES:'
        WRITE(6,*) ' PRESSURE AND AREA       IN FILE:',
     1                    FILNM1(1:IFL1)
        WRITE(6,*) ' VELOCITY AND SOUNDSPEED IN FILE:',
     1                    FILNM2(1:IFL2)
         CALL PLOTIT(T,JI,PREFIX,FILNM1,IFL1,FILNM2,IFL2,SHTU)
         KCC=KCC+1
      ENDIF
   IF((JI.LT.NTO).AND.(((RHOUG(2,N)/RG(2,N)).GT.
```

```
     1     (.8*RHOUG(1,N)/RG(1,N))).OR.(SHTU))) GOTO 300
C IF MAX TIME STEPS EXCEEDED THEN QUIT
      CALL TIME(FINITM)
      RUNTIM=SECNDS(RUNTIM)
      WRITE(6,*)
      WRITE(6,*) ' ************************************************'
      WRITE(6,*) ' RUN-TIME STATISTICS:'
      WRITE(6,*)
      WRITE(6,*) ' STARTING   TIME: ',STARTM(1:29)
      WRITE(6,*) ' COMPLETION TIME: ',FINITM(1:29)
      WRITE(6,*)
      WRITE(6,*) ' RUN TIME (SECS): ',RUNTIM
      WRITE(6,*)
      WRITE(6,*) ' ************************************************'
      CLOSE(5)
      CLOSE(6)
      CALL EXIT
      END


      SUBROUTINE EOS(NT,A1,A2,R)
C   IDEAL GAS EQUATION OF STATE
      COMMON /GASINFO/ROFGAS,GAMMA
      RG=ROFGAS
      IF ((NT.EQ.1) .OR. (NT.EQ.2)) THEN
              R=A1/RG/A2
      ELSE IF (NT.EQ.3) THEN
              R=A1/(GAMMA-1)/A2
      ELSE IF (NT.EQ.4) THEN
              R=SQRT(GAMMA*A1/A2)
      ELSE IF (NT.EQ.5) THEN
              R=(GAMMA-1.)*A1*A2
      ELSE
              REASON=25.0
              CALL DIE (REASON)
      ENDIF
      RETURN
      END


      SUBROUTINE DIE(REASON)
C   EXIT ROUTINE
      WRITE(6,*) '  ********************************************'
      WRITE(6,*) '  ********************************************'
      WRITE(6,*) '  ********************************************'
      WRITE(6,*) '  ********************************************'
      WRITE(6,*) '  FATAL ERROR EXITING  FOR REASON'
      WRITE(6,*) '  REASON=',REASON
      WRITE(6,*) '  ********************************************'
      WRITE(6,*) '  ********************************************'
```

```
      WRITE(6,*) '  ******************************************'
      WRITE(6,*) '  ******************************************'
      CLOSE(5)
      CLOSE(6)
      CALL EXIT
      END



      SUBROUTINE IANDI(U1,P1,TEMP1,NSK,NTO,TSET,NSET,PREFIX,QDOTVO,
     1                SHTU)
C   INPUT AND INITIALIZATION SUBROUTINE
      INCLUDE 'GSCELDAT.INC'
      LOGICAL SHTU
      CHARACTER*20 PREFIX
C ****
C  SET MAX CELL NUMBER = MAX ARRARY DIMENSION - 3
C ****
      KKMAX=148
C *****  100 BLOCK
C  INPUT VALUES
C *****
C
C  READ IN UPSTREAM CONDITIONS
      READ(5,*)U1,P1,TEMP1
C  READ IN GAS CONSTANTS
      READ(5,*)RUGC,ZMW,GAMMA
C  READ IN MESH SIZE, SETTLING TIMES AND CFL
      READ(5,*)NF,NP,NPB,NB,NT,NTO,NSK,NSET
      READ(5,*)XN,XT,CFL  ,TSET
C  READ IN 'MAGIC HEAT ADDITION' Q-DOT
      READ(5,*)QRATE
C  READ IN AREA PROFILE
      DO 110 I=NF+2,NF+NP+2
        READ(5,*)A(I)
110   CONTINUE
C *****  200 BLOCK
C  INITALIZE VARIABLES ACCORDING TO INPUT
C *****
C  ECHO INPUT TO OUTPUT FILE
      WRITE(6,*)' NAME OF INPUT DAT IS:'
      WRITE(6,*)' ',PREFIX
      WRITE(6,*)' U1,P1,TEMP1, IF U1 < 0.0 --> SHOCK TUBE PROB'
      WRITE(6,*)' ',U1,P1,TEMP1
      WRITE(6,*)' RUGC,ZMW,GAMMA'
      WRITE(6,*)' ',RUGC,ZMW,GAMMA
      WRITE(6,*)' NF,NP,NPB,NB,NT,NTO,NSK,NSET'
      WRITE(6,*)' ',NF,NP,NPB,NB,NT,NTO,NSK,NSET
      WRITE(6,*)' XN,XT,CFL,TSET'
      WRITE(6,*)' ',XN,XT,CFL,TSET
```

```
      WRITE(6,*)' QRATE'
      WRITE(6,*)' ',QRATE
      WRITE(6,*)' AREA PROFILE'
      DO 200 I=NF+2,NF+NP+2
C          WRITE(7,*)' ',A(I)
         WRITE(6,*)' ',A(I)
200   CONTINUE
C
C  SET LOGICAL IF SHOCK TUBE TEST
C
      IF (U1*P1.LE.0.0) THEN
         SHTU=.TRUE.
      ELSE
         SHTU=.FALSE.
      ENDIF

C  SET GAS CONSTANTS
C
      ROFGAS=RUGC/ZMW
C      DO 210 I=1,75
C         GAMMA1(I)=GAMMA
C210   CONTINUE
C  TOTAL NUMBER OF FINITE VOLUMES
      KK=2+NF+NP+NB+NT
      IF (KK.GT.KKMAX) THEN
      WRITE(6,*) ' ******* MAX CELL NUMBER EXCEEDED *******'
      WRITE(6,*) ' KKMAX= ',KKMAX
      WRITE(6,*) ' '
      CALL DIE(9999.99)
      ENDIF
C
C   NOT ANY MORE!!!!!!
C
C  AREA MUST BE THE SAME BEFORE AND AFTER AREA CONTRACTION (A/C)
C      IF (A(NF+2).NE.A(NF+NP+2)) THEN
C         REASON=1.0
C         CALL DIE(REASON)
C      ENDIF
C INITIALIZE DUCT BEFORE AND AFTER A/C
      A0=A(NF+2)
      II=1
      III=NF+1
      DO 230 J=1,2
         DO 220  I=II,III
            A(I)=A0
            DADX(I)=0.0
220      CONTINUE
       A0=A(NF+NP+2)
         II=NF+NP+2
         III=KK
230   CONTINUE
```

```
      A(KK+1)=AO
C STEP SIZE SET BY LENGTH OF A/C AND NUMBER OF VOLUMES USED THERE
      DX=(XT-XN)/NP
C DA/DX FOR MOMENTUM SOURCE TERM
      DO 240  I=NF+2,NF+NP+1
         DADX(I)=(A(I+1)-A(I))/DX
240   CONTINUE
C CREATE BOUNDARY AND CENTER COORDINATES (X-WISE) FOR EACH VOLUME
      NF2=NF+2
      XG(1)=XN+DX*(1-NF2)
      DO  250 I=2,KK+1
         XG(I)=XN+DX*(I-NF2)
         XPG(I-1)=.5*(XG(I)+XG(I-1))
         V(I-1)=(A(I)+A(I-1))/2.0 * DX
250   CONTINUE
C
C SET 'MAGIC' HEAT ADDITION
      CALL EOS(1,P1,TEMP1,RHO1)
      TOTVOL=NB*V(NF2+NP)
      IF (NPB.GT.0) THEN
       DO 255 I=1+NF+NP-(NPB-1),1+NF+NP
          TOTVOL=TOTVOL+V(I)
255    CONTINUE
      ENDIF
      QDOTVO=QRATE*RHO1*U1*A(1)/TOTVOL
      DO 260  I=1,KK
         QDOT(I)=0.0
260   CONTINUE
      RETURN
      END


      SUBROUTINE ISNINI(U1,P1,TEMP1)
C INITITIAL CONDITIONS CALCULATIONS
      INCLUDE 'GSCELDAT.INC'
C DECLARE 'MACH' VARIABLES
      REAL MSQ,MNSQ,MIN,MU,ML,MG2
C DEFINE ISENTROPIC FUNCTIONS
      FUNC(X2)=1.0+(GAM-1)/2.0*X2
      PR(X2)=FUNC(X2)**(-GAM/(GAM-1))
      RHOR(X2)=FUNC(X2)**(-1.0/(GAM-1))
      AR2(X2)=1/X2*(2/(GAM+1)*FUNC(X2))**((GAM+1.0)/(GAM-1.0))
C CHECK FOR RIEMANN PROBLEM {Shock Tube Set Up (STSU)}
      IF (U1.LT.0.0) THEN
      CALL STSU(U1,P1,TEMP1)
      RETURN
      ENDIF
C SET INITIAL VALUES BEFORE A/C
      GAM=GAMMA
      CALL EOS(1,P1,TEMP1,RHO1)
      CALL EOS(3,P1,RHO1,EI1)
```

```
      CALL EOS(4,P1,RHO1,C1)
      E1=RHO1*(EI1+U1*U1/2.)
      RHOU1=RHO1*U1
      DO 100  I=1,NF+1
        RG(I,N)=RHO1
C         UG(I,N)=U1
C         TG(I,N)=TEMP1
          EG(I,N)=E1
C         EIG(I,N)=EI1
          PG(I)=P1
          CG(I)=C1
          RHOUG(I,N)=RHOU1
100   CONTINUE
C  SET 'TOTAL' QUANTITIES
      MIN=U1/C1
      MSQ=MIN*MIN
      P01=P1/PR(MSQ)
      RHO01=RHO1/RHOR(MSQ)
      ASTAR2=A(1)*A(1)/AR2(MSQ)
      MG2=MSQ
      EPS=1.0E-5
      DO 200 I=2+NF,2+NF+NP
       AAVE=(A(I)+A(I+1))/2.
       AV2=AAVE*AAVE
       ARN2=AV2/ASTAR2
       DELTAM=MG2-1.0
500    CONTINUE
       AR02=AR2(MG2)
       IF((ABS(ARN2-AR02).GT.2*EPS).AND.(ABS(DELTAM).GT.2*EPS))THEN
         IF ((MG2-1.0).GT.EPS) THEN
C  NEWTON'S METOHD
             AR2MP=AR2(MG2+EPS)
             AR2MM=AR2(MG2-EPS)
             AR2P=(AR2MP-AR2MM)/(2*EPS)
             DELTAM=(ARN2-AR02)/AR2P
             MG2=MG2+DELTAM
         ELSE
C  BOLZANO'S METHOD
             ML=1.0
             MU=10.0
             DELTAM=MU-ML
510          CONTINUE
             MG2=(MU+ML)/2.0
             AR02=AR2(MG2)
             IF ((ABS(ARN2-AR02).GT.EPS).AND.(DELTAM.GT.EPS)) THEN
                IF (AR02.GT.ARN2) THEN
                   MU=MG2
                ELSE
                   ML=MG2
                ENDIF
                DELTAM=MU-ML
```

```
                    GOTO 510
                 ENDIF
              ENDIF
                 GOTO 500
           ENDIF
           PG(I)=PR(MG2)*PO1
           RG(I,N)=RHOR(MG2)*RHOO1
C                CALL EOS (2,PG(I),RG(1,N),TG(1,N))
              CALL EOS (3,PG(I),RG(I,N),EITEMP)
              CALL EOS (4,PG(I),RG(I,N),CG(I))
           UTEMP=SQRT(MG2)*CG(I)
              RHOUG(I,N)=RG(I,N)*UTEMP
              EG(I,N)=RG(I,N)*(EITEMP+UTEMP**2/2.)
 200    CONTINUE
        ISUM=2+NF+NP
        DO 300 I=ISUM+1 ,ISUM+NB+NT
           RG(I,N)=RG(1,N)
C            UG(1,N)=U1
C            TG(I,N)=TG(1,N)
           EG(I,N)=EG(1,N)
C            EIG(I,N)=EIG(1,N)
           PG(I)=PG(1)
           CG(I)=CG(1)
           RHOUG(I,N)=RHOUG(1,N)
 300    CONTINUE
        RETURN
        END

        SUBROUTINE STSU(POP,PRESS3,T)
        INCLUDE 'GSCELDAT.INC'
C COMMON FOR 'EXACT' SOLUTION
        COMMON/RMN/X0,C1,C2,C3,C4,U2,W
        IF (PRESS3.GT.0.0) THEN
C   THEN RIEMANN PROBLEM
C   SOLVE FOR P2/P1 ITERATIVELY
C BOLZANO'S METHOD
           EPS=1E-5
           GAM=GAMMA
           PRL=1E-5
           PRU=44.1
           DPR=PRU-PRL
 510       CONTINUE
           PRG=(PRU+PRL)/2.0
           POPG=1/PRG*(1-(GAM-1.0)/(2*GAM)*(PRG-1.0)/
     1           SQRT(1+(GAM+1.0)/(2*GAM)*(PRG-1.0)))
     2                 **(2*GAM/(GAM-1))
           IF ((ABS(POPG+POP).GT.EPS).AND.(DPR.GT.EPS)) THEN
              IF (POPG.GT.ABS(POP)) THEN
                 PRL=PRG
              ELSE
                 PRU=PRG
```

```
              ENDIF
              DPR=PRU-PRL
              GOTO 510
            ENDIF
            P1=-POP*PRESS3
            P3=PRESS3
              CALL EOS(1,P1,T,RHO1)
              CALL EOS(3,P1,RHO1,EI1)
              CALL EOS(4,P1,RHO1,C1)
              E1=RHO1*EI1
            P2=P1*PRG
              CALL EOS(1,P3,T,RHO3)
              CALL EOS(3,P3,RHO3,EI3)
              CALL EOS(4,P3,RHO3,C3)
              E3=RHO3*EI3
            U2=C3*2.0/(GAM-1.0)*(1.0-(P2/P3)**((GAM-1)/(2.0*GAM)))
            W=C1*SQRT((PRG-1.0)*(GAM+1.0)/(2*GAM)+1.0)
            C4=C3-(GAM-1)/2.0*U2
            RHOR21=((GAM+1.0)*P2+(GAM-1.0)*P1)/((GAM+1.0)*P1+(GAM-1)*P2)
            RHO2=RHOR21*RHO1
            CALL EOS(4,P2,RHO2,C2)
            MIDPT=KK/2
            X0=(XPG(MIDPT)+XPG(MIDPT+1))/2.0
            DO 100 I=1,MIDPT
                RG(I,N)=RHO3
C               UG(I,N)=0.0
C               TG(I,N)=T
                EG(I,N)=E3
C               EIG(I,N)=EI3
                PG(I)=P3
                CG(I)=C3
                RHOUG(I,N)=0.0
100         CONTINUE
            DO 200 I=MIDPT+1,KK
                RG(I,N)=RHO1
C               UG(I,N)=0.0
C               TG(I,N)=T
                EG(I,N)=E1
C               EIG(I,N)=EI1
                PG(I)=P1
                CG(I)=C1
                RHOUG(I,N)=0.0
200         CONTINUE
        ELSE
C AREA CHANGE PROBLEM
            U2C1=-POP
            P1=-PRESS3
              RHO1=T
              CALL EOS(3,P1,RHO1,EI1)
              CALL EOS(4,P1,RHO1,C1)
            E1=RHO1*EI1
```

```
      U1=0.0
      ZMACH=0.0
      DUM=U2C1*C1
      CALL SHOCK(P1,RHO1,C1,U1,ZMACH,DUM,P2,RHO2,C2,U2,ZM2,US
     1                              ,GAMMA)
      CALL EOS(3,P2,RHO2,EI2)
      E2=RHO2*(EI2+U2*U2/2.)
      RU2=RHO2*U2
      IBOUND=1+IFIX(.75*NF)
      DO 7100 I=1,IBOUND
           RG(I,N)=RHO2
           EG(I,N)=E2
           PG(I)=P2
           CG(I)=C2
           RHOUG(I,N)=RU2
 7100 CONTINUE
      DO 7200 I=1+IBOUND,KK
           RG(I,N)=RHO1
           EG(I,N)=E1
           PG(I)=P1
           CG(I)=C1
           RHOUG(I,N)=0.0
 7200 CONTINUE
      ENDIF
      RETURN
      END


      SUBROUTINE SHOCK(P1X,R1X,C1X,U1X,ZM1X,DUCX,P2X,R2X,C2X,
     1                        U2X,ZM2X,US2X,G)
C  NORMAL SHOCK RELATIONS
      Z1=(G+1.)/2.
      Z2=G/(G+1.)
      Z3=(G+1.)/(G-1.)
      B=Z1*ABS(DUCX)/C1X
      P2X=P1X*(1.+Z2*B*(B+(B*B+4.)**.5))
      RPX=P2X/P1X
      R2X=R1X*(Z3*RPX+1.)/(RPX+Z3)
      C2X=C1X*(RPX*R1X/R2X)**.5
      U2X=U1X+DUCX
      ZM2X=U2X/C2X
      WX=R2X*DUCX/(R2X-R1X)
      US2X=U1X+WX
      RETURN
      END


      SUBROUTINE TYME
C  CALCULATE TIME STEP
```

```
      INCLUDE 'GSCELDAT.INC'
      DT=DX/(ABS(RHOUG(1,N)/RG(1,N))+CG(1))
      DO 350 I=2,KK
      CJ=DX/(ABS(RHOUG(I,N)/RG(I,N))+CG(I))
      IF(CJ.LT.DT) THEN
         DT=CJ
      ENDIF
  350 CONTINUE
      DT=CFL*DT
      RETURN
      END



      SUBROUTINE FLUXPR
C PREDICTOR STEP
      INCLUDE 'GSCELDAT.INC'
      COMMON /FLXVAR/FW(151),PM(151),OP(151),FN(151),PN(151),ON(151)
      B1=RHOUG(1,N)/RG(1,N)
      DO 200 I=1,KK-1
         F9=GAMMA-1.
         B=B1
      B1=RHOUG(I+1,N)/RG(I+1,N)
C POSITIVE EIGENVALUES
         P1=(B+ABS(B))/2
         P2=(B+CG(I)+ABS(B+CG(I)))/2
         P3=(B-CG(I)+ABS(B-CG(I)))/2
C NEGATIVE EIGENVALUES
         AP1=(B-ABS(B))/2
         AP2=(B+CG(I)-ABS(B+CG(I)))/2
         AP3=(B-CG(I)-ABS(B-CG(I)))/2
         D=B+CG(I)
         E=B-CG(I)
         G1=RG(I,N)/(2*GAMMA) *A(I+1)
C POSITIVE FLUXES ACROSS THE I+1ST INTERFACE
         FW(I)=(2*F9*P1+P2+P3)*G1
         PM(I)=(2*F9*P1*B+P2*D+P3*E)*G1
         W1=(3-GAMMA)*(P2+P3)*CG(I)*CG(I)/(2*F9)
         OP(I)=(F9*P1*B**2+P2*D*D/2+P3*E*E/2+W1)*G1
         F91=GAMMA-1.
         D2=B1+CG(I+1)
         E2=B1-CG(I+1)
         G2=RG(I+1,N)/(2*GAMMA) *A(I+1)
C NEGATIVE FLUXES ACROSS THE I+1ST INTERFACE
         FN(I)=(2*F91*AP1+AP2+AP3)*G2
         PN(I)=(2*F91*AP1*B1+AP2*D2+AP3*E2)*G2
         W2=(3-GAMMA)*(AP2+AP3)*CG(I+1)**2/(2*F91)
         ON(I)=(F91*AP1*B1**2+AP2*D2*D2/2+AP3*E2*E2/2+W2)*G2
  200 CONTINUE
C CREATE FLUXES FOR KK+1ST INTERFACE  (DNSTREAM BC.)
      IF (ABS(B1).LE.ABS(CG(KK))) THEN
```

```
            KLOOP =KK-1
            CALL DSBC(N,M1)
         ELSE
            KLOOP=KK
            I=KK
            F9=GAMMA-1.
            B=RHOUG(I,N)/RG(I,N)
C POSITIVE EIGENVALUES
            P1=(B+ABS(B))/2
            P2=(B+CG(I)+ABS(B+CG(I)))/2
            P3=(B-CG(I)+ABS(B-CG(I)))/2
            D=B+CG(I)
            E=B-CG(I)
            G1=RG(I,N)/(2*GAMMA) *A(I+1)
C POSITIVE FLUXES ACROSS THE I+1ST INTERFACE
            FW(I)=(2*F9*P1+P2+P3)*G1
            PM(I)=(2*F9*P1*B+P2*D+P3*E)*G1
            W1=(3-GAMMA)*(P2+P3)*CG(I)*CG(I)/(2*F9)
            OP(I)=(F9*P1*B**2+P2*D*D/2+P3*E*E/2+W1)*G1
C SUPERSONIC FLOW... EG. NO NEGATIVE FLUXES
            FN(I)=0.0
            PN(I)=0.0
            ON(I)=0.0
         ENDIF
C CREATE 'PREDICTED' VALUES
         I=1
         RG(I,M1)=RG(I,N)
         RHOUG(I,M1)=RHOUG(I,N)
         EG(1,M1)=EG(I,N)
         DO 201 I=2,KLOOP
C           ST=DT/V(I)       WHAT IS THE DIFFERENCE?????
            ST=DT/(DX*A(I))
            RG(I,M1)=RG(I,N)-ST*(FW(I)-FW(I-1)+FN(I)-FN(I-1))
            RHOUG(I,M1)=RHOUG(I,N)-ST*(PM(I)-PM(I-1)+PN(I)-PN(I-1))+
     1                DT/A(I)*PG(I)*DADX(I)
            EG(I,M1)=EG(I,N)-ST*(OP(I)-OP(I-1)+ON(I)-ON(I-1))+QDOT(I)*DT
  201 CONTINUE
      DO 202 I=1,KK
         UTEMP=RHOUG(I,M1)/RG(I,M1)
         EITEMP=(EG(I,M1)-RHOUG(I,M1)*UTEMP/2)/RG(I,M1)
         CALL EOS(5,RG(I,M1),EITEMP,PG(I))
         CALL EOS(4,PG(I),RG(I,M1),CG(I))
  202 CONTINUE
      RETURN
      END


      SUBROUTINE FLUXCO
C CORRECTOR STEP
      INCLUDE 'GSCELDAT.INC'
```

```
      COMMON /FLXVAR/FW(151),PM(151),OP(151),FN(151),PN(151),ON(151)
      B=RHOUG(1,M1)/RG(1,M1)
      DO 203 I=1,KK-1
        F9=GAMMA-1.
       UI=B
        B=RHOUG(I+1,M1)/RG(I+1,M1)
C  POSITIVE EIGENVALUES
        P1=(B+ABS(B))/2
        P2=(B+CG(I+1)+ABS(B+CG(I+1)))/2
        P3=(B-CG(I+1)+ABS(B-CG(I+1)))/2
C  NEGATIVE EIGENVALUES
        AP1=(B-ABS(B))/2
        AP2=(B+CG(I+1)-ABS(B+CG(I+1)))/2
        AP3=(B-CG(I+1)-ABS(B-CG(I+1)))/2
        D=UI+CG(I)
        E=UI-CG(I)
        G1=RG(I,M1)/(2*GAMMA)   *A(I+1)
C  POSITIVE FLUXES ACROSS THE I+1ST INTERFACE
        FW(I)=(2*F9*P1+P2+P3)*G1
        PM(I)=(2*F9*P1*UI+P2*D+P3*E)*G1
        W1=(3-GAMMA)*(P2+P3)*CG(I)*CG(I)/(2*F9)
        OP(I)=(F9*P1*UI**2+P2*D*D/2+P3*E*E/2+W1)*G1
        F91=GAMMA-1.
        D2=B+CG(I+1)
        E2=B-CG(I+1)
        G2=RG(1+1,M1)/(2*GAMMA)  *A(I+1)
C  NEGATIVE FLUXES ACROSS THE I+1ST INTERFACE
        FN(I)=(2*F91*AP1+AP2+AP3)*G2
        PN(I)=(2*F91*AP1*B+AP2*D2+AP3*E2)*G2
        W2=(3-GAMMA)*(AP2+AP3)*CG(I+1)**2/(2*F91)
        ON(I)=(F91*AP1*B**2+AP2*D2*D2/2+AP3*E2*E2/2+W2)*G2
  203 CONTINUE
C  POSITIVE FLUXES ACROSS THE KK+1ST INTERFACE (DNSTREAM B.C.)
      IF (ABS(B).LE.ABS(CG(KK))) THEN
        KLOOP=KK-1
        CALL DSBC(M1,M)
C  AVERAGE RESULTS OF PREDICTOR AND CORRECTOR STEP (AS USUAL)
C  ALTERNATE METHOD (USES CORRECTOR VALUE ONLY)
C      RG(KK,M)=.5*(RG(KK,M1)+RG(KK,M))
C        RHOUG(KK,M)=.5*(RHOUG(KK,M1)+RHOUG(KK,M))
C        EG(KK,M)=.5*(EG(KK,M1)+EG(KK,M))
      ELSE
        KLOOP=KK
        I=KK
      UI=RHOUG(I,M1)/RG(I,M1)
        F9=GAMMA-1.
        D=U1+CG(I)
        E=UI-CG(I)
        G1=RG(I,M1)/(2*GAMMA)   *A(I+1)
C  POSITIVE FLUXES ACROSS THE KK+1ST INTERFACE
        FW(1)=(2*F9*P1+P2+P3)*G1
```

```
      PM(I)=(2*F9*P1*UI+P2*D+P3*E)*G1
      W1=(3-GAMMA)*(P2+P3)*CG(I)*CG(I)/(2*F9)
      OP(I)=(F9*P1*UI**2+P2*D*D/2+P3*E*E/2+W1)*G1
C  SUPERSONIC FLOW...  EG. NO NEGATIVE FLUXES
      FN(I)=0.0
      PN(I)=0.0
      ON(I)=0.0
      ENDIF
C  GENERATE 'CORRECTED' VALUES
      I=1
      RG(1,M)=RG(I,N)
      RHOUG(I,M)=RHOUG(I,N)
      EG(1,M)=EG(1,N)
      DO 207 I=2,KLOOP
      ST=DT/(DX*A(I+1))
      RG(I,M)=.5*(RG(I,N)+RG(I,M1)-ST*(FW(I)-FW(I-1)+FN(I)-
     1FN(I-1)))
      RHOUG(I,M)=.5*(RHOUG(I,N)+RHOUG(I,M1)-ST*(PM(I)-PM(I-1)
     1+PN(I)-PN(I-1)) + DT/A(I+1)*PG(I)*DADX(I) )
      EG(I,M)=.5*(EG(I,M1)+EG(I,N)-ST*(OP(I)-OP(I-1)+ON(I)-
     1ON(I-1)) +QDOT(I)*DT)
  207 CONTINUE
C  GENERATE UPDATED OTHER VARIABLES OF INTEREST
      DO 208 I=1,KK
         UTEMP=RHOUG(I,M)/RG(I,M)
         EITEMP=(EG(I,M)-RHOUG(I,M)*UTEMP/2)/RG(I,M)
         CALL EOS(5,RG(I,M),EITEMP,PG(I))
         CALL EOS(4,PG(I),RG(I,M),CG(I))
  208 CONTINUE
      RETURN
      END


      SUBROUTINE PRINT1(TT1,JJ,PREFIX)
C PRINTED OUTPUT FOR GSTART
      INCLUDE 'GSCELDAT.INC'
      CHARACTER*20 PREFIX
      L=N
      WRITE (6,901)
901   FORMAT(1H1)
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*)' DATA NAME: ',PREFIX,' OUTPUT AT TIME= ',TT1,
     1          '     AND TIME STEP ',JJ
      WRITE(6,*)' DX= ',DX,'     DT= ',DT
  136 FORMAT(1X,/)
  135 FORMAT(1X,E16.9,21X,E16.9)
  130 FORMAT(7X,'X',13X,'A',13X,'P',13X,'R',13X,'U',13X,'C',13X,'M',
     113X,'E',13X,'T')
      WRITE(6,130)
      GMASS3=0
```

```
      RMASS3=0.
      EMASS3=0.
      DO 134 I=1,KK
      J=I
      UTEMP=RHOUG(J,L)/RG(J,L)
      CALL EOS(2,PG(J),RG(J,L),TTEMP)
      AM=UTEMP/CG(J)
      WRITE(6,132) XPG(J),A(J),PG(J),RG(J,L),UTEMP,CG(J),AM
     1,EG(J,L),TTEMP
      GMASS3=GMASS3+RG(J,L)*V(J)
      RMASS3=RMASS3+RHOUG(J,L)*V(J)
      EMASS3=EMASS3+EG(J,L)*V(J)
  134 CONTINUE
      WRITE(6,*)' CONSERVATION TOTALS'
      WRITE(6,*)GMASS3,RMASS3,EMASS3
      WRITE(6,136)
  132 FORMAT(1X,9(E13.6,1X))
C 'THRUST' CALCULATION
      THRUST=0.0
      DO 500 I=NF+2,NF+NP+1
      THRUST=THRUST+DADX(I)*PG(I)*DX
500   CONTINUE
      WRITE(6,*)' 1st ORDER EST. OF THRUST... SUM OF DADX*DX*P: '
      WRITE(6,*)' THRUST (in consistant units) = ',THRUST
      WRITE(6,*)' '
      WRITE(6,*)' GAMMA=' ,GAMMA
      RETURN
      END



      SUBROUTINE PLOTIT(TT,JJ,PREFIX,FILNM1,IFL1,FILNM2,IFL2,SHTU)
C GRAPHIC OUTPUT FOR GSTART
      INCLUDE 'GSCELDAT.INC'
      COMMON/RMN/X0,C1,C2,C3,C4,U2,W
      LOGICAL SHTU
      REAL UA(150),CA(150),UG(150)
        CHARACTER*24 FILNM1,FILNM2
        CHARACTER*56 TITLE
      CHARACTER*6 PREFIX
      CHARACTER*56 XKEY
      XKEY(1:14)='DISTANCE  (cm)'
      LX=14
      ENCODE(54,4299,TITLE)TT,JJ
4299  FORMAT(15HOUTPUT AT TIME=,G10.4,8H(sec)   ,15H AND TIME STEP ,I6)
      WRITE(6,*)' ',TITLE(1:54)
      DO 100 I=1,KK
        UG(I)=RHOUG(I,N)/RG(I,N)
100   CONTINUE
      IF (SHTU) GOTO 8888
      CALL PLTGO(FILNM1,IFL1)
```

```
      CALL PLTCLR
      CALL PLTDAT(2,KK,XPG,XKEY,LX,A,'    DUCT AREA (sq. cm)',22,PG,
     1              'STATIC PRESSURE (dynes/sq. cm)',30,0)
      CALL PLTTL(TITLE,54)
      CALL PLTSTP(FILNM1,IFL1)
      CALL PLTGO(FILNM2,IFL2)
      CALL PLTCLR
      CALL PLTDAT(2,KK,XPG,XKEY,LX,UG,'  FLUID VELOCITY (cm/sec)',25,
     1        CG,'   SOUND SPEED (cm/sec)',23,3)
      CALL PLTTL(TITLE,54)
      CALL PLTSTP(FILNM2,IFL2)
      RETURN
 8888 CONTINUE
C     WRITE(6,*)' AT PLOTIT: TT,JJ',TT,JJ
C     SHOCK TUBE OUTPUT
      XC=XO-C1*TT
      IC=IFIX((XC-XG(1))/DX+1.0)
      DO 9000 I=1,IC
         CA(I)=C3
         UA(I)=0.0
 9000 CONTINUE
      XC=XO+(U2-C4)*TT
      ICE=IFIX((XC-XG(1))/DX+1.0)
      GAM=GAMMA
      DO 9100 I=IC+1,ICE-1
         ETA=(XPG(I)-XO)/TT
         CA(I)=2/(GAM+1.0)*C3-(GAM-1.0)/(GAM+1.0)*ETA
         UA(I)=CA(I)+ETA
 9100 CONTINUE
      XC=XO+U2*TT
      ICC=IFIX((XC-XG(1))/DX+1.0)
      DO 9200 I=ICE,ICC-1
         UA(I)=U2
         CA(I)=C4
 9200 CONTINUE
      UA(ICC)=U2
      CA(ICC)=C4
      XC=XO+W*TT
      ISH=IFIX((XC-XG(1))/DX+1.0)
      DO 9300 I=ICC+1,ISH
         UA(I)=U2
         CA(I)=C2
 9300 CONTINUE
      DO 9400 I=ISH+1,KK
         UA(I)=0.0
         CA(I)=C1
 9400 CONTINUE
      WRITE(6,9450)
 9450 FORMAT(1H1,6X,'UA',12X,'UERR',10X,'CA',12X,'CERR')
 9475 FORMAT(1X,4(G12.4,2X))
      DO 9500 I=1,KK
```

```
            UNUM=SQRT(UA(I)**2+UG(I)**2)
            IF (UNUM.NE.0.0) THEN
               UERR=(UA(I)-UG(I))/UNUM
            ELSE
               UERR=0.0
            ENDIF
            CNUM=SQRT(CA(I)**2+CG(I)**2)
            IF (CNUM.NE.0.0) THEN
               CERR=(CA(I)-CG(I))/CNUM
            ELSE
               CERR=0.0
            ENDIF
            WRITE(6,9475) UA(I),UERR,CA(I),CERR
 9500    CONTINUE
         CALL PLTGO(FILNM1,IFL1)
         CALL PLTCLR
         CALL PLTDAT(2,KK,XPG,XKEY,LX,UA,'ANALYTIC VEL. RESULT (cm/sec)'
     1            ,29,UG,'CFD VELOCITY RESULT (cm/sec)',28,3)
         CALL PLTTL(TITLE,54)
         CALL PLTSTP(FILNM1,IFL1)
         CALL PLTGO(FILNM2,IFL2)
         CALL PLTCLR
         CALL PLTDAT(2,KK,XPG,XKEY,LX,CA,'SOUND SPEED: ANALYTIC (cm/sec)'
     1        ,30,CG,'SOUND SPEED: CFD (cm/sec)',25,3)
         CALL PLTTL(TITLE,54)
         CALL PLTSTP(FILNM2,IFL2)
         RETURN
         END

         SUBROUTINE FILINI(DIR,LDP,PREFIX,LPP,FILTYP,LTP,NSTART,IERR)
C THIS SUBROUTINE AND THE NEXT FORM A AUTOMATIC
C FILE NAME GENERATOR FOR GRAPHIC OUTPUT FILES (OR OTHER
C SEQUENTIALLY NUMBERED FILES)
          COMMON /FILNAM/FILNAM,/FILNUM/NFILES
          CHARACTER*20 PREFIX,NLSTR,DIR,FILTYP
          CHARACTER*24 FILNAM
C CHECK INCOMING DATA
C
       LD=LDP
       LT=LTP
       LP=LPP
       IF ((LP.GT.20).OR.(LP.LE.0).OR.(LT.GT.20).OR.(LT.LE.0).OR.
     1      (LD.GT.20).OR.(LD.LE.0)) THEN
          IERR=-100
       ELSE
          NLSTR='                    '
          IF (PREFIX(1:LP).EQ.NLSTR(1:LP)) THEN
             IERR=100
          ELSE
C INPUT IS SAFE TO WORK ON....
C SET INITIAL FILE NUMBER
```

```
      NFILES=NSTART
C   INPUT CONDITIONING FOR PREFIX
      IFP=0
      ILP=LP
C     REPEAT
10    CONTINUE
      IFP=IFP+1
      IF ((PREFIX(IFP:IFP).EQ.' ').AND.(IFP.LT.ILP)) GOTO 10
C     UNTIL IFP POINTS TO A NON-SPACE CHARACTER
C     SET ILP TO MAX ALLOWED LENGTH .... IF GREATER
      IF ((ILP-IFP).GT.5) ILP=IFP+5
      DO 20 I=IFP,ILP
        IF (PREFIX(I:I).EQ.' ') THEN
          ILP=I-1
          GOTO 30
        ENDIF
20    CONTINUE
30    CONTINUE
C    PACK FILTYP
      IF (FILTYP(1:LT).EQ.NLSTR(1:LT)) THEN
        IFPT=1
        ILPT=1
      ELSE
        IFPT=0
        ILPT=LT
C       REPEAT
110     CONTINUE
        IFPT=IFPT+1
        IF ((FILTYP(IFPT:IFPT).EQ.' ')
     1          .AND.(IFPT.LT.ILPT)) GOTO 110
C       UNTIL IFPT POINTS TO NON SPACE CHARACTER
        IF ((ILPT-IFPT).GT.2) ILPT=IFPT+2
        DO 120 I=IFPT,ILPT
          IF (FILTYP(I:I).EQ.' ') THEN
            ILPT=I-1
            GOTO 130
          ENDIF
120     CONTINUE
130             CONTINUE
      ENDIF
C    PACK DIR
      IF (DIR(1:LD).EQ.NLSTR(1:LD)) THEN
        IFPD=-1
        ILPD=-1
      ELSE
        IFPD=0
        ILPD=LD
C       REPEAT
210     CONTINUE
        IFPD=IFPD+1
        IF ((DIR(IFPD:IFPD).EQ.' ')
```

```
      1                   .AND.(IFPD.LT.ILPD)) GOTO 210
C           UNTIL IFPT POINTS TO NON SPACE CHARACTER
               IF ((ILPD-IFPD).GT.8) ILPD=IFPD+8
               DO 220 I=IFPD,ILPD
                  IF (DIR(I:I).EQ.' ') THEN
                     ILPD=I-1
                     GOTO 230
                  ENDIF
220            CONTINUE
230                     CONTINUE
            ENDIF
C     CONSTRUCT FILE NAME
            FILNAM(1:23)='                    '
            IPOINT=6-ILP+IFP+11
            IF (IFPD.NE.-1) THEN
               IPOINT=IPOINT-(3+ILPD-IFPD)
               FILNAM(IPOINT:IPOINT)='['
               IPOINT=IPOINT+1
               IEND=IPOINT+ILPD-IFPD
               FILNAM(IPOINT:IEND)=DIR(IFPD:ILPD)
               IPOINT=IEND+1
               IEND=IEND+1
               FILNAM(IPOINT:IEND)=']'
               IPOINT=IEND+1
            ENDIF
            IEND=IPOINT+ILP-IFP
            FILNAM(IPOINT:IEND)=PREFIX(IFP:ILP)
            IPOINT=IEND+3
            IEND=IPOINT
            FILNAM(IPOINT:IEND)='.'
            IPOINT=IEND+1
            IEND=IPOINT+ILPT-IFPT
            FILNAM(IPOINT:IEND)=FILTYP(IFPT:ILPT)
            IERR=000
         ENDIF
      ENDIF
      RETURN
      END

      SUBROUTINE FILNEX(NAME,LENGTH,IERR)
C GENERATES NEXT SEQUENTIAL FILE NAME
        CHARACTER*2 FILNUM
        CHARACTER*24 FILNAM,NAME
        COMMON /FILNAM/FILNAM,/FILNUM/NFILES
        IF (NFILES.GE.10) THEN
         IF (NFILES.GT.99) THEN
            IERR=100
            RETURN
         ELSE
              ASSIGN 4199 TO IFORM
        ENDIF
```

```
      ELSE
         IF (NFILES.LT.0) NFILES=0
         ASSIGN 4099 TO IFORM
      ENDIF
      ENCODE(2,IFORM,FILNUM) NFILES
4099  FORMAT('0',I1)
4199    FORMAT(I2)
      NAME(1:23)=FILNAM(1:23)
      NAME(18:19)=FILNUM(1:2)
      NFILES=NFILES+1
      LENGTH=23
      IERR=000
      RETURN
      END


      SUBROUTINE PLTINI(SCREEN)
C THE FOLLOWING SUBROUTINES FORM A SET AUTOPLOTTER ROUTINES
C FOR THE PRO 300 SERIES WITH CGL OR OTHER CORE GRAPHICS
C COMPATIBLE COMPUTERS
      CALL CGL(90)
      IF (SCREEN.LT.0.0) THEN
         CALL CGL(106,'TI:',3)
         CALL CGL(104,'TI:',3)
      ENDIF
      RETURN
      END

      SUBROUTINE PLTTRM
      CALL CGL(91)
      RETURN
      END

      SUBROUTINE PLTGO(NAME,LENGTH)
      CHARACTER*15 NAME
      CALL CGL(103,NAME,LENGTH)
      CALL CGL(105,NAME,LENGTH)
      RETURN
      END

      SUBROUTINE PLTSTP(NAME,LENGTH)
      CHARACTER*15 NAME
      CALL CGL(106,NAME,LENGTH)
      CALL CGL(104,NAME,LENGTH)
      RETURN
      END

      SUBROUTINE PLTCLR
      CALL CGL(92)
      RETURN
      END
```

```
      SUBROUTINE PLTDAT(NCURVE,NPOINT,X,XKEY,LX,Y1,KEY1,L1,
     1                 Y2,KEY2,L2,NSCLON)
C   PLDAT ...    (ASSUMES YOU HAVE CALLED PLGO ETC.)
C       PLDAT IS A CONVENIENCE AUTOPLOTTER FOR UPPER-HALF-PLANE ONLY
C     FUNCTIONS.  IT PROVIDES SIMPLE INDIRECT ACCESS TO CGL.
C       A MAXIMUM OF TWO CURVES MAY BE PLOTTED ON SIMULTANEOUSLY AS
C     OF THIS REVISION.  THE SECOND CURVE DATA HOWEVER NEED NOT BE
C     PASSED IF ONLY ONE CURVE IS PLOTTED.  SEE BODY OF PROGRAM FOR
C     DESCRIPTION OF THE FUNCTION OF NSCLON.
C     FEATURES...  LEAVES TOP LINE BLANK FOR USE AS TITLE (SEE PLTTL)
C                  GENERATES AXES, GRID LINES AND REFERENCE VALUES
C
      CHARACTER*35 KEY1,KEY2
      REAL X(NPOINT),Y1(NPOINT),Y2(NPOINT)
      CHARACTER*10 Y1VAL(5),Y2VAL(5),XVAL(6)
      INTEGER LENY1(5),LENY2(5),LENX(6)
C INITIALIZE LINE TYPE (JUST IN CASE)
      IONE=1
      CALL CGL(12,IONE)
C FIND EXTREMUM FOR SCALING
      BIGNUM=1E38
      XMAX=X(1)
      XMIN=X(1)
      Y1MAX=-BIGNUM
      Y2MAX=-BIGNUM
      Y1MIN=0.0
      Y2MIN=0.0
      DO 100 I=1,NPOINT
         IF(X(I).GT.XMAX) THEN
              XMAX=X(I)
         ELSE IF (X(I).LT.XMIN) THEN
              XMIN=X(I)
         ENDIF
         IF(Y1(I).GT.Y1MAX) THEN
              Y1MAX=Y1(I)
         ENDIF
100   CONTINUE
      IF (NCURVE.GT.1) THEN
C NSCLON   DENOTES THE SCALING DEPENDANCE BETWEEN THE TWO DATA
C          SETS.  THE VALUES ARE DEFINED AS FOLLOWS
C NSCLON = 0  INDEPENDENT SCALING
C   "    = 1  SCALE ON MAXIMUM OF Y1
C   "    = 2  SCALE ON MAXIMUM OF Y2
C   "    = 3  SCALE ON THE GREATEST OF THE TWO MAXIMA
C
         IF (NSCLON.EQ.1) THEN
            Y2MAX=Y1MAX
         ELSE
            DO 200 I=1,NPOINT
               IF(Y2(I).GT.Y2MAX) THEN
```

```
                Y2MAX=Y2(I)
             ENDIF
200       CONTINUE
          IF (NSCLON.EQ.2) THEN
             Y1MAX=Y2MAX
          ELSE IF (NSCLON.EQ.3) THEN
             IF (Y1MAX.GE.Y2MAX)THEN
                Y2MAX=Y1MAX
             ELSE
                Y1MAX=Y2MAX
             ENDIF
          ENDIF
       ENDIF
    ENDIF
C   GENERATE VALUES FOR SCALING TRANSFORMATIONS
C
    XM=1.0/(XMAX-XMIN)
    XB=-XMIN*XM
    IF (Y1MAX.LE.0.0) THEN
       WRITE(6,*) ' ERROR IN PLTDAT  ALL Y1<0.0'
       RETURN
    ELSE
       Y1M=1/Y1MAX
       Y1B=0.0
    ENDIF
    IF (NCURVE.GT.1) THEN
       IF (Y2MAX.LE.0.0) THEN
          WRITE(6,*) ' ERROR IN PLTDAT  ALL Y2<0.0'
          RETURN
       ELSE
          Y2M=1/Y2MAX
          Y2B=0.0
       ENDIF
    ENDIF
C
C     GENERATE BOUNDARY AND KEY VALUES
C        USING FORTRAN ENCODE STATEMENT
C   LABEL FORMAT IS :
4100  FORMAT(G10.3)
C
      DO 400 I=1,5
         XSC=(XMAX-XMIN)*.2*I+XMIN
         Y1SC=Y1MAX*.2*I
         ENCODE (10,4100,XVAL(I+1)) XSC
         CALL PLT001(XVAL(I+1),LENX(I+1))
         ENCODE (10,4100,Y1VAL(I)) Y1SC
         CALL PLT001(Y1VAL(I),LENY1(I))
400   CONTINUE
      ENCODE(10,4100,XVAL(1)) XMIN
      CALL PLT001(XVAL(1),LENX(1))
      IF (NCURVE.GT.1) THEN
```

```
          DO 450 I=1,5
             Y2SC=Y2MAX*0.2*I
             ENCODE (10,4100,Y2VAL(I)) Y2SC
          CALL PLT001(Y2VAL(I),LENY2(I))
 450      CONTINUE
       ENDIF
C
C SET UP GRID FOR PLOT
C
C BATCH MODE GRAPHICS
      CALL CGL(96)                                         ! START BATCH MODE
      CALL CGL(92)                                         ! CLEAR SCREEN
      IF (NCURVE.EQ.1) THEN
         CALL CGL(80,-0.22,1.0,-0.13334,1.06667)                 ! SET WINDOW
      ELSE
         CALL CGL(80,-0.22,1.22,-0.13334,1.06667)                ! SET WINDOW
      ENDIF
      CALL CGL(86,0)                                       ! SET ORIGIN
      CALL CGL(1,0.0,0.0)                                  ! MOVE TO ORIGIN
      CALL CGL(10,1.0,1.0)                                 ! DRAW PLOT BORDER
      ISIX=6
      CALL CGL(12,ISIX,,1)
      DO 500 I=1,4                                         ! DOTTED LINES FORM
         YCUR=.2*I                                         ! THE BACKGROUND GRID
         CALL CGL(1,0.0,YCUR)
         CALL CGL(4,1.0,YCUR)
         CALL CGL(1,YCUR,0.0)
         CALL CGL(4,YCUR,1.0)
 500  CONTINUE
      CALL CGL(12,IONE,,1)
C                                                                 ! RESET TO SOLID LINES
C     DRAW Y AXIS (X AXIS IS BOTTOM OF PLOT)
C
C     IF (XB.GE.0.0) THEN
C        CALL CGL(1,XB,0.0)                               !XB CORRESPONDS TO X=0
C        CALL CGL(4,XB,1.0)                               !Y-AXIS
C     ENDIF
C
C     GENERATE GRID LABELS
C
      CALL CGL(26,2,1)                                    !CHARJUST, CENTER-TOP
      DO 600 I=1,4                                        !
         XSC=(I-1)*.2                                     ! MOVE TO GRID POSTIONS
         CALL CGL(1,XSC,-0.01)                            ! PRINT LABELS
         CALL CGL(16,XVAL(I),LENX(I))
 600  CONTINUE
      CALL CGL(1,0.77,-0.01)                              !ADJUST 4th LABEL
      CALL CGL(16,XVAL(5),LENX(5))                        !GRID LABEL AT .8 FULL SCALE
      CALL CGL(26,3,1)                                    !CHARJUST, RIGHT-TOP
      CALL CGL(1,1.0,-0.01)                               !PRINT FULL SCALE
      CALL CGL(16,XVAL(6),LENX(6))                        !GRID LABEL
```

```
      CALL CGL(17,10,STRLEN,STRHT)                          !DETERIME VERT. LABEL SIZE
      XWIDTH=STRLEN/10.0
      XOFSET=STRLEN+.2*XWIDTH
      CALL CGL(26,3,2)                                      !CHARJUST, RIGHT-CENTER
      DO 700 I=1,5
         YSC=I*0.2
C+0.01
         CALL CGL(1,-XWIDTH,YSC)                            !MOVE TO GRID POSITIONS
         CALL CGL(16,Y1VAL(I),LENY1(I))                     !PRINT LABEL
         CALL CGL(17,LENY1(I),STRLEN,STRHT)                 !DETERIME VERT. LABEL SIZE
C        XOFSET=STRLEN+.2*XWIDTH
C        CALL CGL(2,-XOFSET,0.025)
C        CALL CGL(5,-.2,0.0)                                !DRAW LINE TYPE
700   CONTINUE
      CALL CGL(1,-0.01,0.0075)
      CALL CGL(16,'0.000',5)                                !LABEL X AXIS
C   GENERATE LEGEND
      CALL CGL(22,2,1)                                      !SET CHARPATH TO 2
      CALL CGL(26,1,1)                                      !CHARJUST LEFT-TOP
      CALL CGL(1,-0.219,0.1)
      CALL CGL(16,KEY1,L1)                                  !PRINT LEGEND FOR DATA SET 1
      CALL CGL(22,0,1)
C     CALL CGL(26,1,3)                                      !CHARJUST LEFT-BOTTOM
C     CALL CGL(1,-0.09,-0.12)
C     CALL CGL(16,KEY1,L1)                                  !PRINT LEGEND FOR DATA SET 1
C     CALL CGL(2,-.01,0.037)
C     CALL CGL(5,-.1,0.0)                                   !DRAW LINE TYPE
C   LABEL X-AXIS
      CALL CGL(1,.5,-.12)
      CALL CGL(26,2,3)
      CALL CGL(16,XKEY,LX)
C   PLOT DATA SET 1
C
      I=1
      XSC=XM*X(I)+XB                                        !SCALE DATA
      YSC=Y1M*Y1(I)
C     IF (YSC.LT.0.0) THEN
C        YSC=0.0
C     ENDIF
      CALL CGL(1,XSC,YSC)                                   !MOVE TO FIRST POINT
      DO 710 I=2,NPOINT
         XSC=XM*X(I)+XB                                     !SCALE DATA
         YSC=Y1M*Y1(I)
C        IF (YSC.LT.0.0) THEN
C           CALL CGL(1,XSC,0.0)
         CALL CGL(4,XSC,YSC)                                !DRAW NEXT DATA POINT
710   CONTINUE
C
C   REPEAT LABEL KEY AND PLOT PROCESS FOR SECOND DATA SET (IF EXTANT)
      IF (NCURVE.GT.1) THEN
         IDTWO=4
```

```
          CALL CGL(12,IDTWO,,1)                    !LINE TYPE 4
          CALL CGL(26,1,2)                         !CHARJUST LEFT-CENTER
          XLOC=1.0+XWIDTH
          DO 800 I=1,5
             YSC=I*.2
C-.01                  !MOVE TO GRID POINT
             CALL CGL(1,XLOC,YSC)
             CALL CGL(16,Y2VAL(I),LENY2(I))
C            CALL CGL(17,LENY2(I),STRLEN,STRHT)!DETERIME VERT. LABEL SIZE
C            XOFSET=STRLEN+.2*XWIDTH
C            CALL CGL(2,-XOFSET,-0.025)
C            CALL CGL(5,-.2,0.0)                   !DRAW LINE TYPE
800       CONTINUE
C   LABEL ZERO
          CALL CGL(1,1.01,0.0075)
          CALL CGL(16,'0.000',5)
C   GENERATE LEGEND                                      !LABEL X AXIS
          CALL CGL(22,2,1)                         !SET CHARPATH 2
          CALL CGL(26,1,1)                         !CHARJUST LEFT-TOP
          XLAB2=1.19
          CALL CGL(1,XLAB2,0.1)
          CALL CGL(16,KEY2,L2)                     !PRINT LEGEND FOR DATA SET 2
          CALL CGL(22,0,1)                         !SET CHARPATH 0
C         CALL CGL(26,1,3)                         !CHARJUST LEFT BOTTOM
C         CALL CGL(1,0.50,-0.12)
C         CALL CGL(16,KEY2,L2)                     !PRINT LEGEND FOR DATA SET 2
C         CALL CGL(2,-.01,0.037)
C         CALL CGL(5,-.1,0.0)                      !DRAW LINE TYPE
C
C   PLOT DATA SET 2
C
          I=1
          XSC=XM*X(I)+XB                           !SCALE DATA
          YSC=Y2M*Y2(I)
C         IF (YSC.LT.0.0) YSC=0.0
          CALL CGL(1,XSC,YSC)                      !MOVE TO 1st DATA POINT
          DO 810 I=1,NPOINT
             XSC=XM*X(I)+XB                        !SCALE DATA
             YSC=Y2M*Y2(I)
C            IF (YSC.LT.0.0) YSC=0.0
             CALL CGL(4,XSC,YSC)                   !DRAW TO NEXT DATA POINT
810       CONTINUE
       CALL CGL(12,IONE,,1)                        !RESET LINE TYPE TO DEFAULT
       ENDIF
       CALL CGL(97)                                !END BATCH
       RETURN
       END

       SUBROUTINE PLTTL(TITLE,LENGTH)
       CHARACTER*66 TITLE
       CALL CGL(26,2,1)
```

```
      CALL CGL(1,.5,1.06)
      CALL CGL(16,TITLE,LENGTH)
      RETURN
      END

      SUBROUTINE PLT001(STR,LEN)
C  STRIPS LEADING AND TRAILING BLANKS FROM ENCODED STRINGS
      CHARACTER STR*10
      LEN=10
      I=0
10    I=I+1
      IF (STR(I:I).EQ.' ') GOTO 10
      J=LEN+1
20    J=J-1
      IF (STR(J:J).EQ.' ') GOTO 20
      LEN=J-I+1
      IF (I.NE.1) THEN
         DO 30 K=I,J
            KK=K-I+1
            STR(KK:KK)=STR(K:K)
30       CONTINUE
      ENDIF
      RETURN
      END
```

COMMON BLOCK INCLUDED BY PROGRAM LINE

" INCLUDE 'GSCELDAT.INC' "

```
      COMMON /CELLDAT/RG(150,2),EG(150,2),
     1 PG(150),CG(150),RHOUG(150,2),
     2 QDOT(150),XG(151),A(151),DADX(150),XPG(150),V(150)
      COMMON /STEPSIZ/DNETA,KK,M,M1,DT,N,DX,NF,NP,NPB,NB,NT,CFL
      COMMON /GASINFO/ROFGAS,GAMMA
```

EXAMPLE INPUT FILE: FOR GSTART

```
9,5,3,0
USERFILES
EXMPL
GID
-1.0
101287.0,1.295E8,300.0
8.3144E7,22.9,1.35
2,21,0,20,20,3000,500,0
0.0,25.0,0.9,0.0
3.2302E10
10.00
 9.50
 9.025
 8.574
 8.145
 7.738
 7.351
 6.983
 6.634
 6.302
 5.987
 5.987
 6.302
 6.634
 6.983
 7.351
 7.738
 8.145
 8.574
 9.025
 9.500
10.000
```