BRL CR 252

BRL

CONTRACT REPORT NO. 252

A TWO-DIMENSIONAL UNSTEADY METHOD OF CHARACTERISTICS ANALYSIS OF FLUID FLOW PROBLEMS - MCDU 43

Prepared by

Drexel University
32nd & Chestnuts Streets
Philadelphia, PA 19104

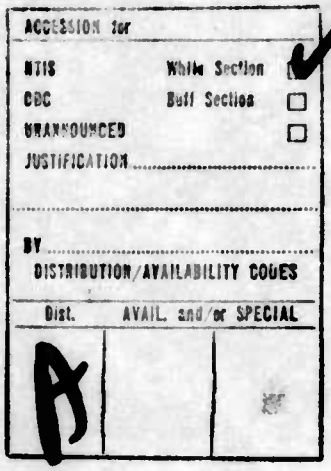
July 1975

USA BALLISTIC RESEARCH LABORATORIES
ABERDEEN PROVING GROUND, MARYLAND

Destroy this report when it is no longer needed.
Do not return it to the originator.

Secondary distribution of this report by originating or sponsoring activity is prohibited.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, Springfield, Virginia 22151.



The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: BRL Contract Report No. 252	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): A Two-Dimensional Unsteady Method of Characteristics Analysis of Fluid Flow Problems - MCDU 43	5. TYPE OF REPORT & PERIOD COVERED: Final Contract Report, 1973 – 1975
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Pei Chi Chou, Arthur Warnock, William Flis	8. CONTRACT OR GRANT NUMBER(s): DAAD05-73-C-0484
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Drexel University, 32nd & Chestnuts Streets, Philadelphia, PA 19104	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Project Number 1T061102A14B
11. CONTROLLING OFFICE NAME AND ADDRESS: Director, USA Ballistic Research Laboratories, Aberdeen Proving Ground, Maryland 21005	12. REPORT DATE: Jul 75
	13. NUMBER OF PAGES: 155
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): BRL CR-252	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

DA-1-T-061102-A-14-B

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Unsteady flow
Inviscid flow
Two-dimensional flow
Method of characteristics
Numerical Analysis
Blast wave

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A computer code for solving two-dimensional, unsteady, inviscid, fluid-flow problems has been developed. Presented are details of the method-of-characteristics formulation, the treatment of the boundary conditions, and the numerical solution technique. In a sample problem of a cylindrical blast wave, the error to the exact solution is less than 0.05% at all points after 50 times planes of calculation.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 405 723

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS

Nomenclature

p - pressure, N/m^2

ρ - density, kg/m^3

$\bar{v}$ - velocity vector, m/sec

u_r - radial velocity component, m/sec

u_z - axial velocity component, m/sec

E - internal energy, joules/kg

a - sound speed, m/sec

δ - bicharacteristic angular coordinate, degrees

ϕ - bicharacteristic angle at the mach cone apex, degrees

θ - bicharacteristic angle at the mach cone base, degrees

r - radial coordinate, m

z - axial coordinate, m

t - time coordinate, seconds

I. Introduction

This report presents a numerical solution to the problem of two-dimensional, unsteady, compressible flow of an inviscid, non-conducting fluid by the method of characteristics.

The numerical method developed here can be applied to many unsteady fluid mechanics problems, such as the interaction of a blast wave with solid surfaces, one-phase gas flow in interior ballistics, and flow around turbine blades. It may also be used to obtain solutions of two-dimensional steady flow problems by running the unsteady problem to an asymptotic time limit.

An alternate numerical solution is the finite-difference method, in which the differential equations are written directly in difference form and integrated in a step-by-step manner. This technique is comparatively simple and well suited to the digital computer. The major disadvantage of the finite-difference method is its lack of accuracy in regions where shock waves or large property gradients exist. It also encounters accuracy problems near flow boundaries, where loss of accuracy results from extrapolation and one-sided difference equations [1]*.

The approach presented here is based on the method of characteristics, in which linear combinations of the equations are formed in such a way that, at each point in the space of the three independent variables, differentiation occurs parallel to certain planes, called characteristic planes. Thus, the combined equations, known as compatibility equations, become partial differential equations, and a relatively complex numerical scheme is required for their integration.

Several techniques of solution have been advanced using the method of characteristics. Thornhill [2] proposed a finite-difference scheme applied along three bicharacteristics (lines of intersection between characteristic surfaces and the Mach cone, the envelope of characteristic surfaces). A mesh is constructed of points which are the intersections of three families of characteristic surfaces. Each new point (P_4 in Fig. 1) is found by integrating the compatibility equations along the three bicharacteristics originating at the known base points (P_1, P_2, and P_3). The major advantage of this method is that no interpolation is required to determine the properties at the base points; however, since the domain of dependence (the Mach cone) is larger than the domain of the finite-difference scheme, the calculations become unstable after a short time.**

---

*Numbers in brackets designate references at end of report.

**According to the Courant-Friedrichs-Lewy stability criterion [3], a necessary condition for the numerical stability of the solution is that the domain of dependence of the differential equations fall within the domain of dependence of the difference equations. See also [4].

Sauerwein [4] has proposed a version of Thornhill's scheme called the modified tetrahedral network, shown in Fig. 2. To ensure numerical stability, the base of the Mach cone is inscribed in the triangle forming the domain of the difference scheme; a single interpolation is then required for the variables at each of the base points P_{12}, P_{23}, P_{31}. However, in formulating his finite-difference equations, Sauerwein makes the assumption that the partial derivatives of the dependent variables remain constant on the characteristic surfaces, thus abandoning second-order accuracy. The method is discussed in greater detail in Appendix A.

In another technique proposed by Coburn and Dolph [5] and later refined by Holt [6], the bicharacteristic lines are arranged in what has been called the prismatic network, shown in Fig. 3. The new point P_6 is found by writing finite-difference relations along the two bicharacteristics P_2-P_6 and P_3-P_6 and introducing a third relation along the space curve P_5-P_6. The introduction of this ordinary curve implies that the flow must be known on some surface other than the initial surface, such as a plane of symmetry. In addition, the ordinary curve introduces a disturbance to P_6 from outside its domain of dependence. Because of this, difficulties are encountered in the handling of boundary conditions at shock waves and contact discontinuities.

Finally, the technique proposed by Butler [7], which is used in this report, is a combination of the standard finite-difference approach and the method of characteristics scheme originally conceived by Hartree [8]. As shown in Fig. 4, the new point P_o is chosen and the Mach cone projected down to the initial surface. Four bicharacteristics are used in determining the flow properties at the new point, one more than necessary. The addition of this fourth equation and the proper orientation of the bicharacteristics permit the elimination of the partial derivatives at the new point, which in other schemes are finite-differenced. The flow properties at the base points P_1, P_2, P_3, and P_4 must be obtained by two-dimensional interpolation. To complete the solution, an additional equation is introduced along a curve other than a bicharacteristic. This curve must lie inside the envelope of the characteristic surfaces through the point under consideration. The solution at each new point can thus be computed without using conditions outside its domain of dependence, while at the same time maintaining second-order accuracy in the solution.

In solving for points along a flow boundary, those equations written along curves falling outside the flow are replaced by the boundary conditions, the required number of which will be determined by the number of lost equations.

Although Butler's technique involves considerably more interpolation, and hence complexity in the computer coding, the straight-forward logic of the scheme, the facility in the application of the boundary conditions, and the ability to maintain second-order accuracy appear to outweigh the disadvantage.

It has come to our attention during the preparation of the manuscript of this report that Delaney and Kavanagh [9] have applied the method of characteristics to transonic flow in turbomachinery cascades. They used essentially the same approach as ours, but with least-squares interpolation instead of the second-degree polynomial method. They did not show detailed error analysis, and a direct comparison of their results with ours can not be made readily. But, in general, as shown in Appendix F of this report, the least-squares interpolation is not as accurate as the second-degree polynomial method used in this report.

In this report, the characteristic cones, bicharacteristic equations, and compatibility equations are first derived. The numerical scheme adopted is then explained in detail. An example problem, the blast wave problem, which has a known exact solution, is solved by the present method and the error analyzed. It is shown that after calculation of 50 times planes, the maximum error in all properties at any point is less then 0.05%. The accuracy of the method seems satisfactory.

II. Analysis by Method of Characteristics

Governing Equations

The governing equations for unsteady flow in an inviscid, compressible, non-conducting fluid are the continuity, momentum, and energy equations, from which the following equations can be derived:

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \bar{v} = 0 \tag{1}$$

$$\frac{D\bar{v}}{Dt} + \frac{1}{\rho} \nabla p = 0 \tag{2}$$

$$\frac{DE}{Dt} - \frac{p}{\rho^2} \frac{D\rho}{Dt} = 0 \tag{3}$$

where the total derivative $\frac{D}{Dt} = \frac{\partial}{\partial t} + \bar{v} \cdot \nabla$. Let the equation of state be

$$E = E(p,\rho) \tag{4}$$

For the two-dimensional case, in cylindrical coordinates, eqs. (1) through (4) represent five equations in five unknowns: p, pressure; ρ, density; u_r and u_z, the particle velocities in the radial and axial directions, respectively; and $E(p,\rho)$, the specific internal energy. Before proceeding further, the internal energy E will be eliminated from the energy equation by means of the equation of state. Substitution of Eq. (4) into Eq. (3) yields

$$\frac{D\rho}{Dt} - \frac{1}{a^2} \frac{Dp}{Dt} = 0 \tag{5}$$

where the sound speed a is defined as

$$a^2 = \frac{\frac{p}{\rho^2} - \left(\frac{\partial E}{\partial \rho}\right)}{\left(\frac{\partial E}{\partial p}\right)} \tag{6}$$

Equations (1), (2), (5), and (6) now represent five equations in the unknowns p, ρ, u_r, u_z, and a. This set of equations comprises a system of four first-order, quasi-linear, hyperbolic partial differential equations in the three independent variables r, z, and t, and an equation of state.

Method of Characteristics

The application of the method of characteristics to this system follows essentially the procedure given in Refs. [7] & [9]. It involves forming a linear combination of the governing equations in such a way that it results in a system of equations where the partial derivatives occur in two new coordinates only (that is, within the characteristic plane of the system), rather than in the three physical coordinates r, z, and t.

To find the proper combination, the governing Eqs. (1), (2), and (5) are transformed into a new orthogonal coordinate system (β_1, β_2, β_3); each is multiplied by an arbitrary factor, then all are added together. By requiring that those terms involving derivatives along one coordinate (say β_1) vanish, the appropriate values of the multiplying factors can be found, and the form of the transformation obtained. The resulting equation will then contain only derivatives in β_2 and β_3, which will describe the characteristic planes. The mathematical details of this procedure are given below.

Transforming Eqs. (1), (2), and (5) into a coordinate system (β_1, β_2, β_3) we obtain the following four (scalar) equations:

$$\sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial t} + u_r \sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + u_z \sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} + \rho \left\{ \sum_{i=1}^{3} \frac{\partial u_r}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + \frac{u_r}{r} + \sum_{i=1}^{3} \frac{\partial u_z}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} \right\} = 0 \qquad (7)$$

$$\sum_{i=1}^{3} \frac{\partial u_r}{\partial \beta_i} \frac{\partial \beta_i}{\partial t} + u_r \sum_{i=1}^{3} \frac{\partial u_r}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + u_z \sum_{i=1}^{3} \frac{\partial u_r}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} + \frac{1}{\rho} \sum_{i=1}^{3} \frac{\partial p}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} = 0 \qquad (8)$$

$$\sum_{i=1}^{3} \frac{\partial u_z}{\partial \beta_i} \frac{\partial \beta_i}{\partial t} + u_r \sum_{i=1}^{3} \frac{\partial u_z}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + u_z \sum_{i=1}^{3} \frac{\partial u_z}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} + \frac{1}{\rho} \sum_{i=1}^{3} \frac{\partial p}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} = 0 \qquad (9)$$

$$\sum_{i=1}^{3} \frac{\partial p}{\partial \beta_i} \frac{\partial \beta_i}{\partial t} + u_r \sum_{i=1}^{3} \frac{\partial p}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + u_z \sum_{i=1}^{3} \frac{\partial p}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} - a^2 \left\{ \sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial t} + u_r \sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial r} + u_z \sum_{i=1}^{3} \frac{\partial \rho}{\partial \beta_i} \frac{\partial \beta_i}{\partial z} \right\} = 0 \qquad (10)$$

If we multiply Eq. (7) by λ_1, Eq. (8) by λ_2, Eq. (9) by λ_3, and Eq. (10) by λ_4, then add all these equations together and finally group the dependent variables in terms of β_1, β_2, and β_3, we obtain

$$\begin{aligned}
&\frac{\partial \rho}{\partial \beta_1}\left[(\lambda_1 - a^2\lambda_4)\left\{\frac{\partial \beta_1}{\partial t} + u_r \frac{\partial \beta_1}{\partial r} + u_z \frac{\partial \beta_1}{\partial z}\right\}\right] + \\
&\frac{\partial u_r}{\partial \beta_1}\left[\lambda_2 \frac{\partial \beta_1}{\partial t} + \left\{\lambda_2 u_r + \rho\lambda_1\right\} \frac{\partial \beta_1}{\partial r} + \lambda_2 u_z \frac{\partial \beta_1}{\partial z}\right] + \\
&\frac{\partial u_z}{\partial \beta_1}\left[\lambda_3 \frac{\partial \beta_1}{\partial t} + \lambda_3 u_r \frac{\partial \beta_1}{\partial r} + \left\{\rho\lambda_1 + \lambda_3 u_z\right\} \frac{\partial \beta_1}{\partial z}\right] + \\
&\frac{\partial p}{\partial \beta_1}\left[\lambda_4 \frac{\partial \beta_1}{\partial t} + \left\{\frac{\lambda_2}{\rho} + \lambda_4 u_r\right\} \frac{\partial \beta_1}{\partial r} + \left\{\frac{\lambda_3}{\rho} + \lambda_4 u_z\right\} \frac{\partial \beta_1}{\partial z}\right] + \\
&\frac{\partial \rho}{\partial \beta_2}\left[(\lambda_1 - a^2\lambda_4)\left\{\frac{\partial \beta_2}{\partial t} + u_r \frac{\partial \beta_2}{\partial r} + u_z \frac{\partial \beta_2}{\partial z}\right\}\right] + \qquad (11) \\
&\frac{\partial u_r}{\partial \beta_2}\left[\lambda_2 \frac{\partial \beta_2}{\partial t} + \left\{\lambda_2 u_r + \rho\lambda_1\right\} \frac{\partial \beta_2}{\partial r} + \lambda_2 u_z \frac{\partial \beta_2}{\partial z}\right] + \\
&\frac{\partial u_z}{\partial \beta_2}\left[\lambda_3 \frac{\partial \beta_2}{\partial t} + \lambda_3 u_r \frac{\partial \beta_2}{\partial r} + \left\{\rho\lambda_1 + \lambda_3 u_z\right\} \frac{\partial \beta_2}{\partial z}\right] + \\
&\frac{\partial p}{\partial \beta_2}\left[\lambda_4 \frac{\partial \beta_2}{\partial t} + \left\{\frac{\lambda_3}{\rho} + \lambda_4 u_r\right\} \frac{\partial \beta_2}{\partial r} + \left\{\frac{\lambda_3}{\rho} + \lambda_4 u_z\right\} \frac{\partial \beta_2}{\partial z}\right] + \\
&\frac{\partial \rho}{\partial \beta_3}\left[(\lambda_1 - a^2\lambda_4)\left\{\frac{\partial \beta_3}{\partial t} + u_r \frac{\partial \beta_3}{\partial r} + u_z \frac{\partial \beta_3}{\partial z}\right\}\right] +
\end{aligned}$$

$$\frac{\partial u_r}{\partial \beta_3}\left[\lambda_2 \frac{\partial \beta_3}{\partial t} + \{\lambda_2 u_r + \rho\lambda_1\}\frac{\partial \beta_3}{\partial r} + \lambda_2 u_z \frac{\partial \beta_3}{\partial z}\right] +$$

$$\frac{\partial u_z}{\partial \beta_3}\left[\lambda_3 \frac{\partial \beta_3}{\partial t} + \lambda_3 u_r \frac{\partial \beta_3}{\partial r} + \{\rho\lambda_1 + \lambda_3 u_z\}\frac{\partial \beta_3}{\partial z}\right] + \tag{11}$$

$$\frac{\partial p}{\partial \beta_3}\left[\lambda_4 \frac{\partial \beta_3}{\partial t} + \left\{\frac{\lambda_2}{\rho} + \lambda_4 u_r\right\}\frac{\partial \beta_3}{\partial r} + \left\{\frac{\lambda_3}{\rho} + \lambda_4 u_z\right\}\frac{\partial \beta_3}{\partial z}\right] =$$

$$-K\lambda_1 \rho \frac{u_r}{r}$$

where a parameter K has been added to accomodate plane flows; K equals zero for two-dimensional plane flow, and unity for axisymmetric flow in cylindrical coordinates.

If the equation is to contain derivatives with respect to β_2 and β_3 only, then the coefficients of $\partial\rho/\partial\beta_1$, $\partial p/\partial\beta_1$, $\partial u_r/\partial\beta_1$ and $\partial u_z/\partial\beta_1$ must be equal to zero; thus,

$$\{\lambda_1 - a^2\lambda_4\}\left\{\frac{\partial \beta_1}{\partial t} + u_r\frac{\partial \beta_1}{\partial r} + u_z\frac{\partial \beta_1}{\partial z}\right\} = 0$$

$$\lambda_2\frac{\partial \beta_1}{\partial t} + \{\lambda_2 u_r + \rho\lambda_1\}\frac{\partial \beta_1}{\partial r} + \lambda_2 u_z\frac{\partial \beta_1}{\partial z} = 0 \tag{12}$$

$$\lambda_3\frac{\partial \beta_1}{\partial t} + \lambda_3 u_r\frac{\partial \beta_1}{\partial r} + \{\rho\lambda_1 + \lambda_3 u_z\}\frac{\partial \beta_1}{\partial z} = 0$$

$$\lambda_4\frac{\partial \beta_1}{\partial t} + \left\{\frac{\lambda_2}{\rho} + \lambda_4 u_r\right\}\frac{\partial \beta_1}{\partial r} + \left\{\frac{\lambda_3}{\rho} + \lambda_4 u_z\right\}\frac{\partial \beta_1}{\partial z} = 0$$

Rewriting these equations in vector form gives

$$\bar{A}_1 \cdot \nabla\beta_1 = \bar{A}_2 \cdot \nabla\beta_1 = \bar{A}_3 \cdot \nabla\beta_1 = \bar{A}_4 \cdot \nabla\beta_1 = 0 \tag{13}$$

where

$$\bar{A}_1 = (\lambda_1 - a^2\lambda_4)(u_r\bar{e}_r + u_z\bar{e}_z + \bar{e}_t)$$

$$\bar{A}_2 = (\lambda_2 u_r + \rho\lambda_1)\bar{e}_r + \lambda_2 u_z\bar{e}_z + \lambda_2\bar{e}_t$$

$$\bar{A}_3 = \lambda_3 u_r \bar{e}_r + (\rho\lambda_1 + \lambda_3 u_z)\bar{e}_z + \lambda_3 \bar{e}_t$$

$$\bar{A}_4 = \left(\frac{\lambda_2}{\rho} + \lambda_4 u_r\right)\bar{e}_r + \left(\frac{\lambda_3}{\rho} + \lambda_4 u_z\right)\bar{e}_z + \lambda_4 \bar{e}_t$$

in which $\bar{e}_r$, $\bar{e}_z$, $\bar{e}_t$ are unit vectors in the r, z, t directions respectively, and the gradient is also taken in these three dimensions. The expression $\bar{A}_i \cdot \nabla\beta_1$ represent the directional derivatives of β_1, in the directions of A_1, A_2, A_3, and A_4. Since they are all zero, the vectors A_i must lie in a common plane normal to the gradient of β_1, i.e., in a β_1= constant plane.

Equation (12) can also be expressed in matrix form:

$$\begin{bmatrix} \Phi & 0 & 0 & -a^2\Phi \\ \rho\frac{\partial\beta_1}{\partial r} & \Phi & 0 & 0 \\ \rho\frac{\partial\beta_1}{\partial z} & 0 & \Phi & 0 \\ 0 & \frac{1}{\rho}\frac{\partial\beta_1}{\partial r} & \frac{1}{\rho}\frac{\partial\beta_1}{\partial z} & \Phi \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix} = 0 \qquad (14)$$

where

$$\Phi = u_r\frac{\partial\beta_1}{\partial r} + u_z\frac{\partial\beta_1}{\partial z} + \frac{\partial\beta_1}{\partial t}$$

This is a system of homogeneous algebraic equations in the λ's; for a solution to exist, the determinant of the coefficient matrix must vanish. Therefore,

$$\Phi^2\left[\Phi^2 - a^2\left\{\left(\frac{\partial\beta_1}{\partial r}\right)^2 + \left(\frac{\partial\beta_1}{\partial z}\right)^2\right\}\right] = 0 \qquad (15)$$

Thus, the solution for the characteristic surface consists of a repeated linear factor and a quadratic cone.

Consider first the quadratic part of the solution:

$$(u_r \frac{\partial \beta_1}{\partial r} + u_z \frac{\partial \beta_1}{\partial z} + \frac{\partial \beta_1}{\partial t})^2 - a^2 \left\{ (\frac{\partial \beta_1}{\partial r})^2 + (\frac{\partial \beta_1}{\partial z})^2 \right\} = 0 \tag{16}$$

We wish to determine the derivatives of β_1 (i.e., the orientation of the β_1 = constant surface) in order to obtain the λ's from Eq. (14), which will in turn yield the compatibility relations on the characteristic surface upon substitution into Eq. (11). We assume a direction for $\nabla\beta_1$ as given in Fig. 5 and rewrite the derivatives of β_1 in terms of the new coordinates Ψ and δ to obtain

$$\frac{\partial \beta_1}{\partial r} = \psi \cos\delta \quad ; \quad \frac{\partial \beta_1}{\partial z} = \psi \sin\delta \tag{17}$$

In addition, we require that $\nabla\beta_1$ be a unit vector

$$\psi^2 + (\frac{\partial \beta_1}{\partial t})^2 = 1 \tag{18}$$

Solving Eqs. (16), (17) and (18), we obtain

$$\begin{aligned} \frac{\partial \beta_1}{\partial r} &= \frac{\cos\delta}{\{1+(q+a)^2\}^{1/2}} \\ \frac{\partial \beta_1}{\partial z} &= \frac{\sin\delta}{\{1+(q+a)^2\}^{1/2}} \\ \frac{\partial \beta_1}{\partial t} &= -\frac{q+a}{\{1+(q+a)^2\}^{1/2}} \end{aligned} \tag{19}$$

where $q = u_r \cos\delta + u_z \sin\delta$, and the signs have been chosen to agree with a direction of the β_1 coordinate as given in Fig. 6.

By substituting eqs. (19) into eq. (14), we obtain the values of the multipliers:

$$\frac{\lambda_4}{\lambda_1} = \frac{1}{a^2}$$
$$\frac{\lambda_2}{\lambda_1} = \frac{\rho}{a}\cos\delta \qquad (20)$$
$$\frac{\lambda_3}{\lambda_1} = \frac{\rho}{a}\sin\delta$$

With these results, Eq. (11) yields the compatibility equations for the dependent variables on the characteristic cone,

$$dp + \rho a\cos\delta\, du_r + \rho a\sin\delta\, du_z = -\rho a^2 s\, dt \qquad (21)$$

where

$$s = K\frac{u_r}{r} + \sin^2\delta\,\frac{\partial u_r}{\partial r} - \sin\delta\cos\delta\left\{\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}\right\} + \cos^2\delta\,\frac{\partial u_z}{\partial z}$$

The intersections of the characteristic surfaces with the characteristic cone are the bicharacteristic curves, defined by the following equations.

$$\frac{dr}{dt} = u_r + a\cos\delta$$
$$\frac{dz}{dt} = u_z + a\sin\delta \qquad (22)$$

Returning to eq. (15), the repeated linear solution

$$\left\{u_r\frac{\partial\beta_1}{\partial r} + u_z\frac{\partial\beta_1}{\partial z} + \frac{\partial\beta_1}{\partial t}\right\}^2 = 0 \qquad (23)$$

is the equation of the particle path line. By performing an

analysis similar to the one on the cone, it is found that the energy is conserved along the particle path; i.e., the compatibility equation is

$$\frac{Dp}{Dt} = a^2 \frac{D\rho}{Dt} \tag{5}$$

From Eq. (23) can be found the equation of the particle path line,

$$\frac{dr}{dt} = u_r$$
$$\frac{dz}{dt} = u_z \tag{24}$$

In summary, Eqs. (5), (6), and (21) comprise the governing partial differential equations for the five dependent variables p, ρ, u_r, u_z, and a, on the characteristic surface described by Eq. (19). It may be seen that, although an infinite number of equations can be written in the form of Eq. (21), only three will be independent.

III. Finite-Difference Scheme

Orientation of Bicharacteristics

In order to solve an initial-value problem numerically, Eqs. (5) and (21) must be written in finite-difference form. Since we will solve the final equations algebraically, we must consider the partial derivatives $\partial u_r/\partial r$ and $\partial u_z/\partial z$ as unknowns. In other method-of-characteristic schemes, these terms are often finite-differenced; however, Butler's technique, the one used here, presents a different approach. While only three equations of the form of (21) are independent, four are written for the angles $\phi = 0, \pi/2, \pi, 3\pi/2$, then appropriately combined to eliminate the partial derivatives.

Flow properties at the new point will be denoted by the subscript 0; those at the point on the $t = t_0 - \Delta t$ plane (initial surface for the current time step) at the base of the $\phi = 0$- characteristic by the subscript 1; those at the base point of the $\phi = \pi/2$-characteristic by the subscript 2; etc.

Writing Eq. (21) in finite-difference form along each of the bicharacteristics we obtain

$$p_0 - p_i + \tfrac{1}{2}[\rho_0 a_0 \cos\phi_i + \rho_i a_i \cos\theta_i][u_{r_0} - u_{r_i}] +$$

$$\tfrac{1}{2}[\rho_0 a_0 \sin\phi_i + \rho_i a_i \sin\theta_i][u_{z_0} - u_{z_i}] =$$

$$-\tfrac{1}{2}[\rho_0 a_0^2\{K\frac{u_{r_0}}{r_0} + \sin^2\phi_i\left(\frac{\partial u_r}{\partial r}\right)_0 - \tag{25}$$

$$\sin\phi_i\cos\phi_i\left(\frac{\partial u_r}{\partial z} + \frac{\partial u_z}{\partial r}\right)_0 + \cos^2\phi_i\left(\frac{\partial u_z}{\partial z}\right)_0\} + \rho_i a_i^2 S_i]\Delta t, \qquad i = 1, 2, 3, 4$$

where $$S_i = K\frac{u_{r_i}}{r_i} + \sin^2\theta_i\left(\frac{\partial u_r}{\partial r}\right)_i + \cos^2\theta_i\left(\frac{\partial u_z}{\partial z}\right)_i$$

$$- \sin\theta_i\cos\theta_i\left\{\left(\frac{\partial u_r}{\partial z}\right)_i + \left(\frac{\partial u_z}{\partial r}\right)_i\right\}$$

where the θ_i's are evaluated in Appendix B and defined in Fig. B1. These angles arise from the assumption that the bicharacteristic curves are not straight lines, thus preserving second-order accuracy.

We may also write the energy equation (5) in finite-difference form along the particle path, the properties at the base point of which are denoted by the subscript 5.

$$p_0 - p_5 = -\frac{1}{2}\rho_0 a_0^2 \left\{K\frac{u_{r_0}}{r_0} + \left(\frac{\partial u_r}{\partial r}\right)_0 + \left(\frac{\partial u_z}{\partial z}\right)_0\right\}\Delta t -$$

$$\frac{1}{2}\rho_5 a_5^2 \left\{K\frac{u_{r_5}}{r_5} + \left(\frac{\partial u_r}{\partial r}\right)_5 + \left(\frac{\partial u_z}{\partial z}\right)_5\right\}\Delta t \qquad (26)$$

If the partial derivatives at the new point are treated as dependent variables, then we have so far six equations, (6), (25), (26), in seven unknowns, p_0, ρ_0, a_0, u_{r_0}, u_{z_0}, $(\partial u_r/\partial r)_0$, $(\partial u_z/\partial z)_0$, clearly one more equation is required to solve the problem. We choose to integrate the continuity equation (1) along the particle path.

$$\rho_0 - \rho_5 = -\frac{1}{2}\rho_0\left\{K\frac{u_{r_0}}{r_0} + \left(\frac{\partial u_r}{\partial r}\right)_0 + \left(\frac{\partial u_z}{\partial z}\right)_0\right\}\Delta t$$

$$-\frac{1}{2}\rho_5\left\{K\frac{u_{r_5}}{r_5} + \left(\frac{\partial u_r}{\partial r}\right)_5 + \left(\frac{\partial u_z}{\partial z}\right)_5\right\}\Delta t$$

Application of Finite-Difference Method

Having completed the mathematical system, it remains to reduce it to a form amenable to an iterative solution by the computer. We define the quantities

$$\Delta p = p_0 - p_5$$

$$\Delta\rho = \rho_0 - \rho_5$$

$$\Delta u_r = u_{r_0} - u_{r_5}$$

$$\Delta u_z = u_{z_0} - u_{z_5}$$

Substituting the values for the ϕ_1's, we can rewrite the equations in the following form:

$$\Delta p + \tfrac{1}{2}[\rho_0 a_0 + \rho_1 a_1 \cos\theta_1]\Delta u_r + \tfrac{1}{2}\rho_0 a_0 [u_{r5} - u_{r1}] + \tfrac{1}{2}\rho_1 a_1 \sin\theta_1 \Delta u_z = -\tfrac{1}{2}[\rho_0 a_0^2 \{K \tfrac{u_{r0}}{r_0} + (\tfrac{\partial u_z}{\partial z})_0\}]\Delta t - R_1 \quad (27)$$

$$\Delta p + \tfrac{1}{2}\rho_2 a_2 \cos\theta_2 \Delta u_r + \tfrac{1}{2}\rho_0 a_0 [u_{z5} - u_{z2}] + \tfrac{1}{2}[\rho_0 a_0 + \rho_2 a_2 \sin\theta_2]\Delta u_z = -\tfrac{1}{2}[\rho_0 a_0^2 \{K \tfrac{u_{r0}}{r_0} + (\tfrac{\partial u_r}{\partial r})_0\}]\Delta t - R_2 \quad (28)$$

$$\Delta p + \tfrac{1}{2}[-\rho_0 a_0 + \rho_3 a_3 \cos\theta_3]\Delta u_r - \tfrac{1}{2}\rho_0 a_0 [u_{r5} - u_{r3}] + \tfrac{1}{2}\rho_3 a_3 \sin\theta_3 \Delta u_z = -\tfrac{1}{2}[\rho_0 a_0^2 \{K \tfrac{u_{r0}}{r_0} + (\tfrac{\partial u_z}{\partial z})_0\}]\Delta t - R_3 \quad (29)$$

$$\Delta p + \tfrac{1}{2}\rho_4 a_4 \cos\theta_4 \Delta u_r - \tfrac{1}{2}\rho_0 a_0 [u_{z8} - u_{z4}] + \tfrac{1}{2}[-\rho_0 a_0 + \rho_4 a_4 \sin\theta_4]\Delta u_z = -\tfrac{1}{2}[\rho_0 a_0^2 \{K \tfrac{u_{r0}}{r_0} + (\tfrac{\partial u_r}{\partial r})_0\}]\Delta t - R_4 \quad (30)$$

$$\Delta p = -\tfrac{1}{2}\rho_0 a_0^2 \left\{ K\,\frac{u_{r_0}}{r_0} + \left(\frac{\partial u_r}{\partial r}\right)_0 + \left(\frac{\partial u_z}{\partial z}\right)_0 \right\} \Delta t - R_5 \qquad (31)$$

$$\Delta \rho = -\tfrac{1}{2}\rho_0 \left\{ K\,\frac{u_{r_0}}{r_0} + \left(\frac{\partial u_r}{\partial r}\right)_0 + \left(\frac{\partial u_z}{\partial z}\right)_0 \right\} \Delta t - \frac{R_5}{a_5^2} \qquad (32)$$

where

$$R_i = p_5 - p_i + \tfrac{1}{2}\rho_i a_i \cos\theta_i [u_{r_5} - u_{r_i}] + \tfrac{1}{2}\rho_i a_i \sin\theta_i [u_{z_5} - u_{z_i}] + \tfrac{1}{2}\rho_i a_i^2 s_i \Delta t \,, \quad i = 1,2,3,4$$

$$R_5 = \tfrac{1}{2}\rho_5 a_5^2 \left\{ K\,\frac{u_{r_5}}{r_5} + \left(\frac{\partial u_r}{\partial r}\right)_5 + \left(\frac{\partial u_z}{\partial z}\right)_5 \right\} \Delta t$$

Combination of the Compatibility Equations

As noted previously, these equations may now be appropriately combined to eliminate the partial derivatives $(\partial u_r/\partial r)_0$, $(\partial u_z/\partial z)_0$.

Subtraction of eq. (29) from eq. (27) yields

$$\tfrac{1}{2}[2\rho_0 a_0 + \rho_1 a_1 \cos\theta_1 - \rho_3 a_3 \cos\theta_3]\Delta u_r + \tfrac{1}{2}\rho_0 a_0 [2u_{r_5} - u_{r_1} - u_{r_3}] + \tfrac{1}{2}[\rho_1 a_1 \sin\theta_1 - \rho_3 a_3 \sin\theta_3]\Delta u_z = R_3 - R_1 \qquad (33)$$

Similarly, subtraction of eq. (30) from eq. (28) yields:

$$\tfrac{1}{2}[\rho_2 a_2 \cos\theta_2 - \rho_4 a_4 \cos\theta_4]\Delta u_r + \tfrac{1}{2}\rho_0 a_0 [2u_{z_5} - u_{z_2} - u_{z_4}] + \tfrac{1}{2}[2\rho_0 a_0 + \rho_2 a_2 \sin\theta_2 - \rho_4 a_4 \sin\theta_4]\Delta u_z = R_4 - R_2 \qquad (34)$$

Next, equations (27), (28), (29), and (30) are summed and this result is subtracted from twice eq. (31).

$$2\Delta p + \tfrac{1}{2}[\rho_1 a_1 \cos\theta_1 + \rho_2 a_2 \cos\theta_2 + \rho_3 a_3 \cos\theta_3 + \rho_4 a_4 \cos\theta_4]\Delta u_r + \tfrac{1}{2}[\rho_1 a_1 \sin\theta_1 + \rho_2 a_2 \sin\theta_2 + \rho_3 a_3 \sin\theta_3 + \rho_4 a_4 \sin\theta_4]\Delta u_z + \tfrac{1}{2}\rho_0 a_0 [u_{r_3} - u_{r_1} + u_{z_4} - u_{z_2}] = -\rho_0 a_0^2 K\,\frac{u_{r_0}}{r_0}\Delta t + 2R_5 - [R_1 + R_2 + R_3 + R_4] \qquad (35)$$

Finally, the sum of Eqs. (27), (28), (29), and (30) is subtracted from twice Eq. (29).

$$4\Delta p - 2a_o^2\Delta\rho + \tfrac{1}{2}[\rho_1 a_1\cos\theta_1 + \rho_2 a_2\cos\theta_2 + \rho_3 a_3\cos\theta_3 + \rho_4 a_4\cos\theta_4]\Delta u_r + \tfrac{1}{2}[\rho_1 a_1\sin\theta_1 + \rho_2 a_2\sin\theta_2 + \rho_3 a_3\sin\theta_3 + \rho_4 a_4\sin\theta_4]\Delta u_z + \tfrac{1}{2}\rho_o a_o[u_{r_3} - u_{r_1} + u_{z_4} - u_{z_2}] = -\rho_o a_o^2 K\frac{u_{r_o}}{r_o}\Delta t + 2\frac{a_o^2}{a_5^2}R_5 - [R_1 + R_2 + R_3 + R_4] \quad (36)$$

Thus, we have five equations, (6), (33), (34), (35), and (36), to solve for the unknowns a_o, Δp, $\Delta\rho$, Δu_r, and Δu_z at the new point (r_o, z_o, t_o). For the first iteration, crude estimates are made for a_o, ρ_o, p_o, u_{r_o}, and u_{z_o} based on corresponding values in the initial time plane. Then using Eqs. (22) and (24), the four bicharacteristics and the particle path are projected down to the initial time plane. The flow properties at the base points are obtained by interpolation, and Eqs. (6), (33), (34), (35), and (36) are solved for improved values of a_o, Δp, $\Delta\rho$, Δu_r, and Δu_z. The process is repeated until the improved values are close within a certain tolerance to the previous values, i.e., convergence is obtained. (For details, see Appendix C)

Boundary Conditions

Many types of boundaries may be encountered in formulating a fluid flow problem. We shall discuss three main types, namely, the particle-path boundary, the exit boundary, and the entrance boundary.

The particle-path boundary is the simplest to handle. It may be a rigid wall with prescribed velocity, a free surface, or an interface with another fluid. We have formulated the numerical treatment of the rigid wall only. In this case, among the four bicharacteristics traced backwards from a boundary point, only one falls outside the flow domain, Fig. 7. Only one property needs to be specified, since only one compatibility equation is missing. For a rigid wall, the vanishing of the normal velocity is the proper boundary condition. In the example problem to be presented later, we have prescribed the pressure along a plane of symmetry, which may also be considered as a rigid wall.

On an exit boundary, among the four bicharacteristics traced back from a boundary point, only one falls outside the flow domain, and the particle path falls within the domain, Fig. 8. Again, only one compatibility equation is missing, and only one property has to be specified. In the example problem, we have specified either the pressure or one velocity component as the boundary condition for an exit boundary.

For an entrance boundary, three of the four bicharacteristics, as well as the particle path line, all fall outside the flow domain, Fig. 9. Compared to an interior point, there are five equations missing: three compatibility equations along the three bicharacteristics, and the energy- and mass-conservation equations along the particle path. But among the seven unknowns, p, ρ, a, u_r, u_z, $\partial u_r/\partial r$, and $\partial u_z/\partial z$, the last one does not appear in our calculation scheme. If the tangential velocity u_r is specified along the z = constant boundary, then $\partial u_r/\partial r$ may be considered as known. Therefore, specifying three boundary conditions is sufficient for the solution of the entrance boundary point. In the example problem, we have specified the two velocity components and density.

Boundary Finite-Difference Equations

In developing the finite-difference equations for use on the boundaries, an approach similar to the general-point equations has been used. First, consider the finite-difference formulation for the plane-wall (particle-path) and exit boundaries, as shown in Figs. 7 and 8. To solve for the seven unknowns, p_o, ρ_o, a_o, u_{r_o}, u_{z_o}, $(\partial u_r/\partial r)_o$, and $(\partial u_z/\partial z)_o$, at the boundary point, we write three compatibility equations in the form of Eq. (21) for the angles $\phi = 0$, π, and $3\pi/2$; these equations are the same as Eq. (25) for $i = 1$, 3, and 4. See Figs. 7 and 8. Also, we write energy- and mass-conservation equations along the particle path. These, coupled with the equation of state (6) and the appropriate boundary conditions, complete our system of equations.

In a manner similar to that used for the general-point finite-difference equations, the derivatives $(\partial u_r/\partial r)_o$ and $(\partial u_z/\partial z)_o$ are eliminated from the system, yielding the following difference equations for the exit-plane and plane-wall boundaries.

$$p_o - p_5 + \tfrac{1}{4}\{\rho_1 a_1 \cos\theta_1 + \rho_3 a_3 \cos\theta_3 + 2\rho_4 a_4 \cos\theta_4\}\Delta u_r +$$
$$\tfrac{1}{4}\{-2\rho_o a_o + \rho_1 a_1 \sin\theta_1 + \rho_3 a_3 \sin\theta_3 + 2\rho_4 a_4 \sin\theta_4\}\{u_{z_o} - u_{z_5}\}$$
$$+ \tfrac{1}{4}\rho_o a_o\{u_{r_3} - u_{r_1} + 2u_{z_4} - 2u_{z_3}\} = R_5$$
$$- \tfrac{1}{2}\rho_o a_o^2 K \cdot \frac{u_{r_o}}{r_o}\Delta t - \tfrac{1}{2}\{R_1 + R_3 + 2R_4\}$$

$$a_o^2 \Delta\rho = p_o - p_5 - R_5\left\{\frac{a_o^2}{a_5^2} - 1\right\}$$

$$\tfrac{1}{2}\{2\rho_o a_o + \rho_1 a_1 \cos\theta_1 - \rho_3 a_3 \cos\theta_3\}\Delta u_r +$$
$$\tfrac{1}{2}\{\rho_1 a_1 \sin\theta_1 - \rho_3 a_3 \sin\theta_3\}\{u_{z_o} - u_{z_5}\} + \tfrac{1}{2}\rho_o a_o\{2u_{r_5}$$
$$- u_{r_1} - u_{r_3}\} = R_3 - R_1$$

where p_o is specified for the exit-plane boundary and u_{z_o} for the plane wall.

For a point on the entrance boundary, we may write only one equation in the form of (21), for the angle $\phi = 3\pi/2$, and the equation of state (6). The variables to be solved for are p_o, ρ_o, a_o, u_{r_o}, u_{z_o}, and $(\partial u_r/\partial r)_o$. To complete the system, we specify three boundary conditions; the derivative $(\partial u_r/\partial r)_0$ can be obtained by differentiating the boundary condition u_r. The single difference equation for the entrance boundary thus becomes

$$p_o - p_4 + \tfrac{1}{2}\rho_4 a_4 \cos\theta_4\{u_{r_o} - u_{r_4}\}$$
$$+ \tfrac{1}{2}\{\rho_o a_o - \rho_4 a_4 \sin\theta_4\}\{u_{z_o} - u_{z_4}\}$$
$$= -\tfrac{1}{2}\left[\rho_o a_o^2\left\{K\frac{u_{r_o}}{r_o} + \left(\frac{\partial u_r}{\partial r}\right)_o\right\} + \rho_4 a_4^2 S_4\right]\Delta t$$

where u_{r_o} and any two of the set $\{p_o, \rho_o, a_o, u_{z_o}\}$ may be specified.

Numerical Stability

The stability criteria for first-order non-linear, hyperbolic, partial differential equations has been discussed in Refs. [4], [9], [11], and [12]. The Courant-Friedrichs-Lewy stability criterion is a necessary condition for both linear and non-linear systems of equations. Briefly stated, the CFL criterion requires that the domain of dependence of the difference scheme must contain the domain of the differential equations. For simple difference schemes the CFL criterion is both necessary and sufficient, while for a network such as Butler's (Fig. 4) the requirements for stability are unclear. However, using the CFL criterion to regulate the integration time increment does seem to insure stability for our system. This is most likely due to the interpolation scheme for the cone base points, which effectively increases the domain of dependence of the difference scheme to such an extent that it contains the domain of dependence of the differential equations.

IV. Example Problem

Application to Blast Wave

An example problem with a known analytical solution is solved numerically by the present computer code. The error between our numerical solution and the analytical solution is used to judge the degree of accuracy of the present code.

The problem chosen is the cylindrical blast wave problem solved by Sedov [10]. This problem has axial symmetry; one space variable, radius r, and one time variable are sufficient. In applying our numerical method, we purposely used two space variables, r and z, and one time variable. For this two-dimensional, unsteady problem the data for the initial conditions and boundary conditions are all taken from the exact solution. After numerical calculation of a number of time planes, the result is then compared with the exact solution.

The initial time chosen is t = 159.1 μsec in the blast wave problem. At this time, the shock front is at r = 24 units, and a rectangular area in the r,z plane was chosen as our domain of calculation. This domain was divided into a grid of 18 axial increments of 0.5 unit ,and 24 radial increments of 0.5 unit, as shown in Fig. 10. Exact values of all properties are taken from Sedov's solution and assumed as our initial values at the t = 159.1 plane at these mesh points. Of the four boundaries z = 12 (boundary 1) and r = 12 (boundary 4) are exit boundaries, z = 3 (boundary 3) is an entrance boundary, and r = 0 (boundary 2) is a fixed boundary. The boundary conditions specified are: u_z along boundary 1, p along boundary 2, ρ, u_r and u_z along boundary 3, and p along boundary 4. Values of these quantities are taken again from Sedov's solution and prescribed along these boundaries for all time.

With these specified initial and boundary conditions, we calculated the properties of the interior grid-points on succeeding time planes until the 50th time plane. Figure 11 shows the points and lines at which the percent error has been plotted in Figs. 12, 13 and 14. The exact values from Sedov are also indicated. It may be seen that the error in the density, the quantity for which it has been most difficult to obtain convergence, at the 50th time plane is less than 0.05% at all points. The data also show no sign of instability at any point.

References

1. Porter, R.W., and Coakley, J.F., "Use of Characteristics for Boundaries in Time Dependent Finite Difference Analysis of Multidimensional Gas Dynamics," International Journal for Numerical Methods in Engineering, Vol. 5, 1972, pp. 91-101.

2. Thornhill, C.K., The Numerical Method of Characteristics for Hyperbolic Problems in Three Independent Variables," Aeronautical Research Council R&M No. 2615, September, 1948.

3. Courant, R.; Friedrichs, K.O.; and Lewy, H., "Über die partiellen Differenzengleichungen der mathematischen Physik," Math. Ann., vol. 100, 1928, pp. 32-74.

4. Sauerwein, Harry, The Calculation of Two- and Three-Dimensional Inviscid Unsteady Flows by the Method of characteristics," AFOSR 64-1055, MIT Fluid Dynamics Research Lab. Report No. 6-4, AD 605324, June, 1964.

5. Coburn, N., and Dolph, C.L., Proceedings of the First Symposium on Applied Mathematics, American Mathematical Society, 1947, p. 55.

6. Holt, M., "The Method of Characteristics for Steady Supersonic Rotational Flow in Three Dimensions," Journal of Fluid Mechanics, Vol. 1, October 1956, Part 4, p. 409.

7. Butler, D.S., "The Numerical Solution of Hyperbolic Systems of Partial Differential Equations in Three Independent Variables," Proceedings of the Royal Society, Series A, Vol. 225, 1960, p. 232-252.

8. Hartree, D.R., "Some Practical Methods of Using Characteristics in the Calculation of Non-Steady Compressible Flow," United States Atomic Energy Commission Report AECU-2713, September 1953.

9. Delaney, R.A., and Kavanagh, P., "Transonic Flow Analysis in Axial Flow Turbomachinery Cascades by a Time-Dependent Method of Characteristics," Iowa State Univ., Engineering Research Inst., 74034

10. Karpp, R. and Chou, P.C., "The Method of Characteristics", Chapter 6 of Dynamic Response of Materials to Intense

Impulsive Loading, edited by P.C. Chou and A.K. Hopkins, Air Force Materials Lab., Wright Patterson AFB, Ohio 45433, 1972.

11. Cline, M.C. and Hoffman, J.D., "The Analysis of Nonequilibrium, Chemically Reacting, Supersonic Flow in Three Dimensions Using a Bicharacteristics Method," Journal of Computational Physics, Vol. 12, pp. 1-23 (1973).

12. Ransom, V.H.; Hoffman, J.D.; and Thompson, H.D., "Second-Order Bicharacteristics Methods for Three-Dimensional, Steady, Supersonic Flow," AIAA Journal, Vol. 10, No. 12, pp. 1573-1581, 1972.

Acknowledgements

The authors would like to express thanks to Dr. Aivars Celmins of Ballistic Research Laboratories for his general guidance, and to Dr. D. Tuckmantel, of Dyna East Corporation and Mr. Robert Croman, of Drexel University for their comments and suggestions.

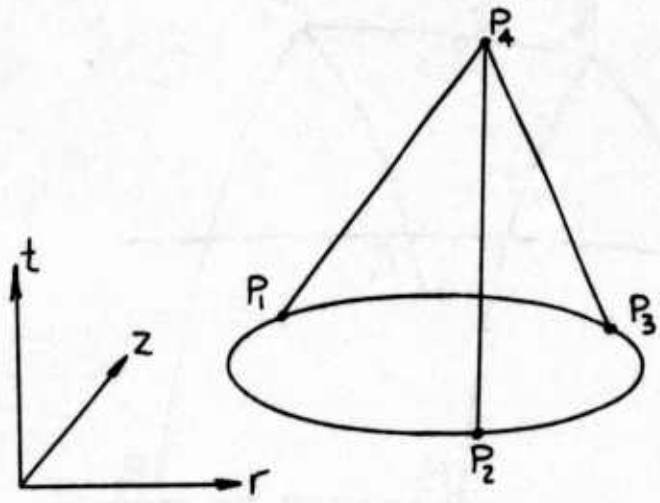


Figure 1. Diagram of Thornhill's Characteristic Scheme.

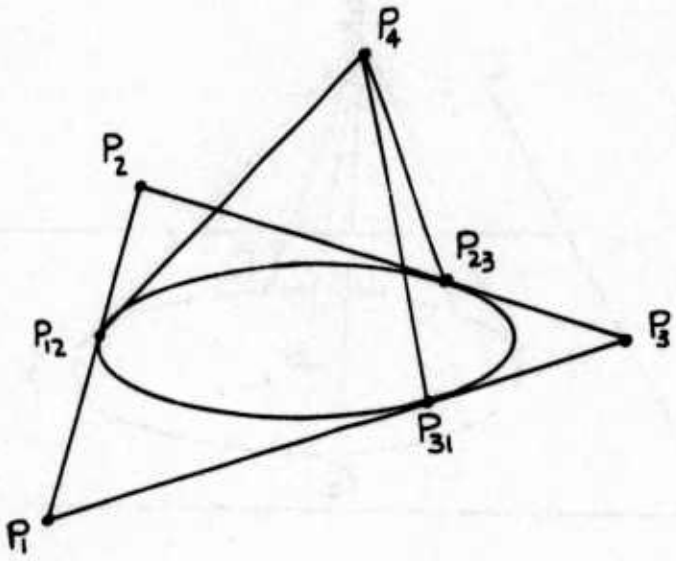


Figure 2. Diagram of Sauerwein's Modified Tetrahedral Characteristic Scheme.

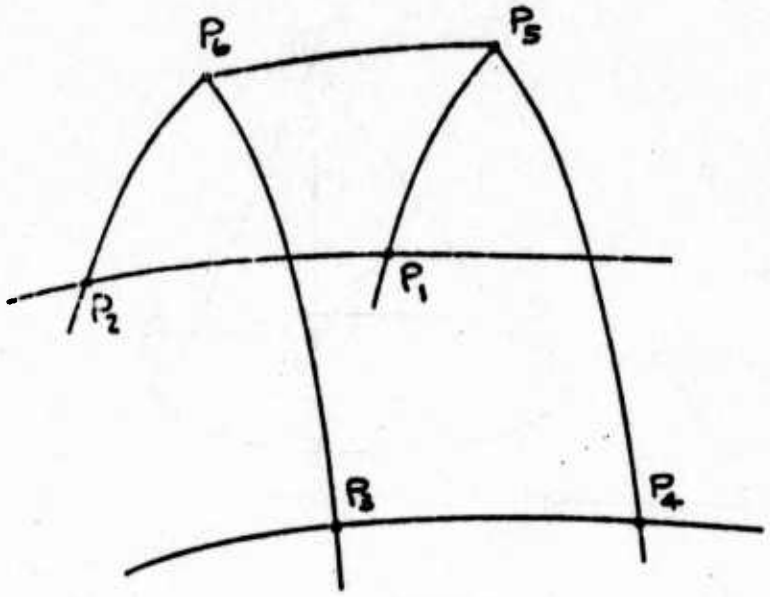


Figure 3. Diagram of prismatic network developed by Coburn & Dolph and Holt.

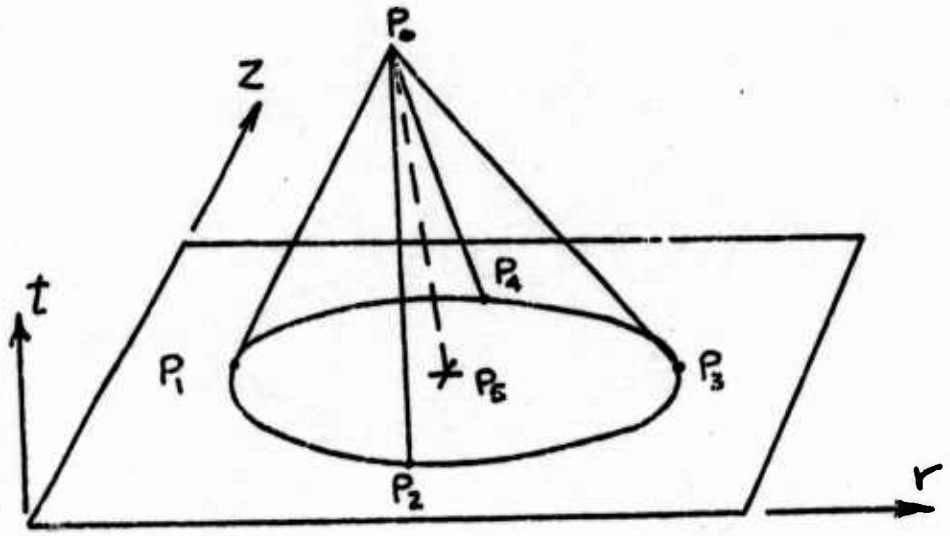


Figure 4. Diagram of Butler's characteristic scheme.

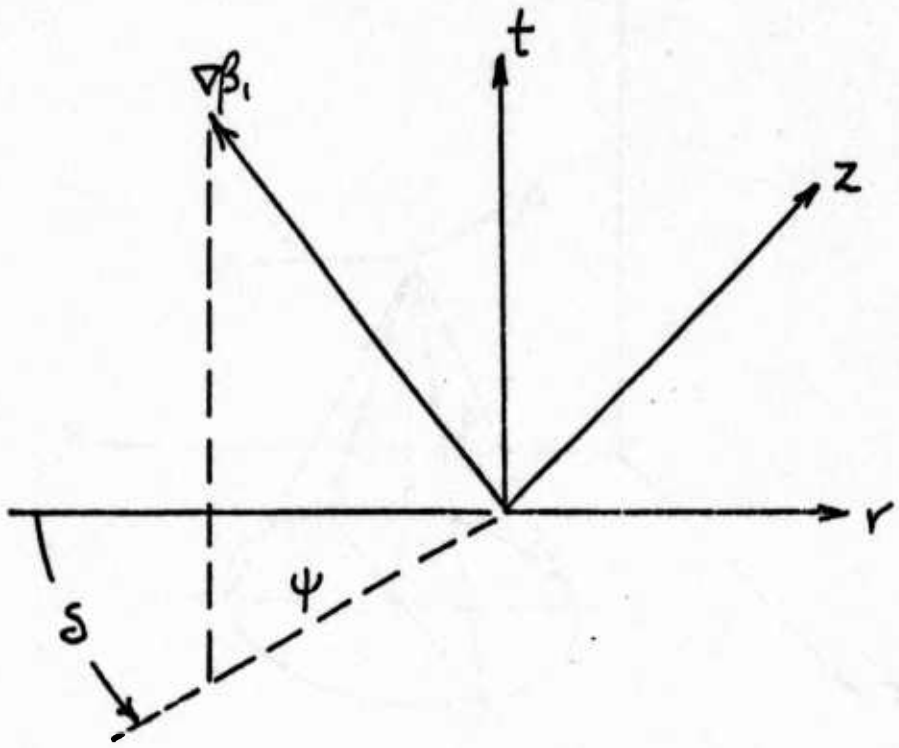


Figure 5. Diagram of unit vector describing characteristic plane.

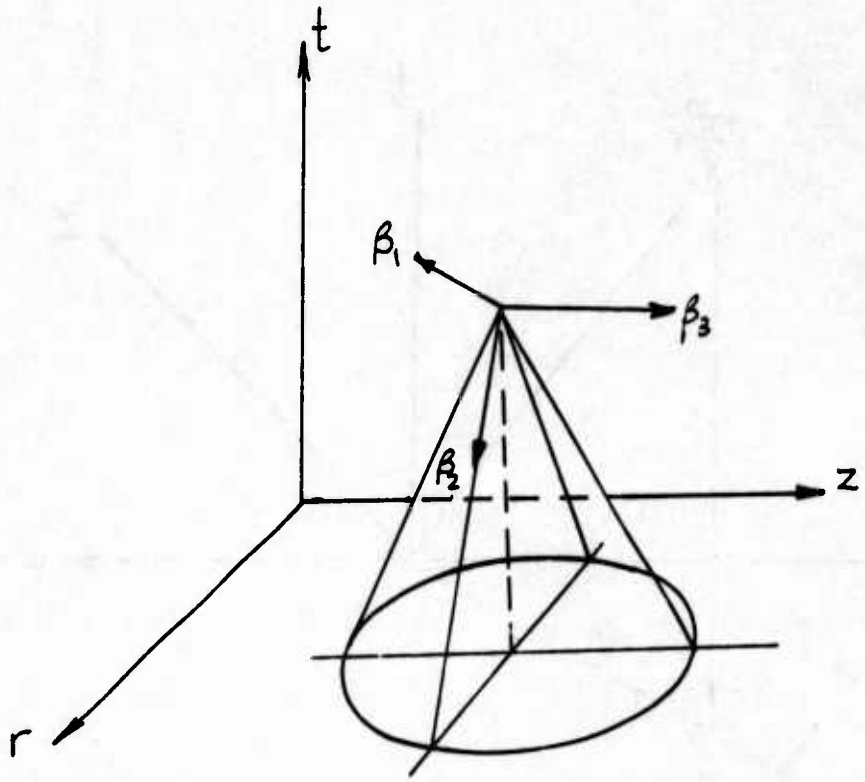


Figure 6. Diagram of characteristic cone and coordinate system.

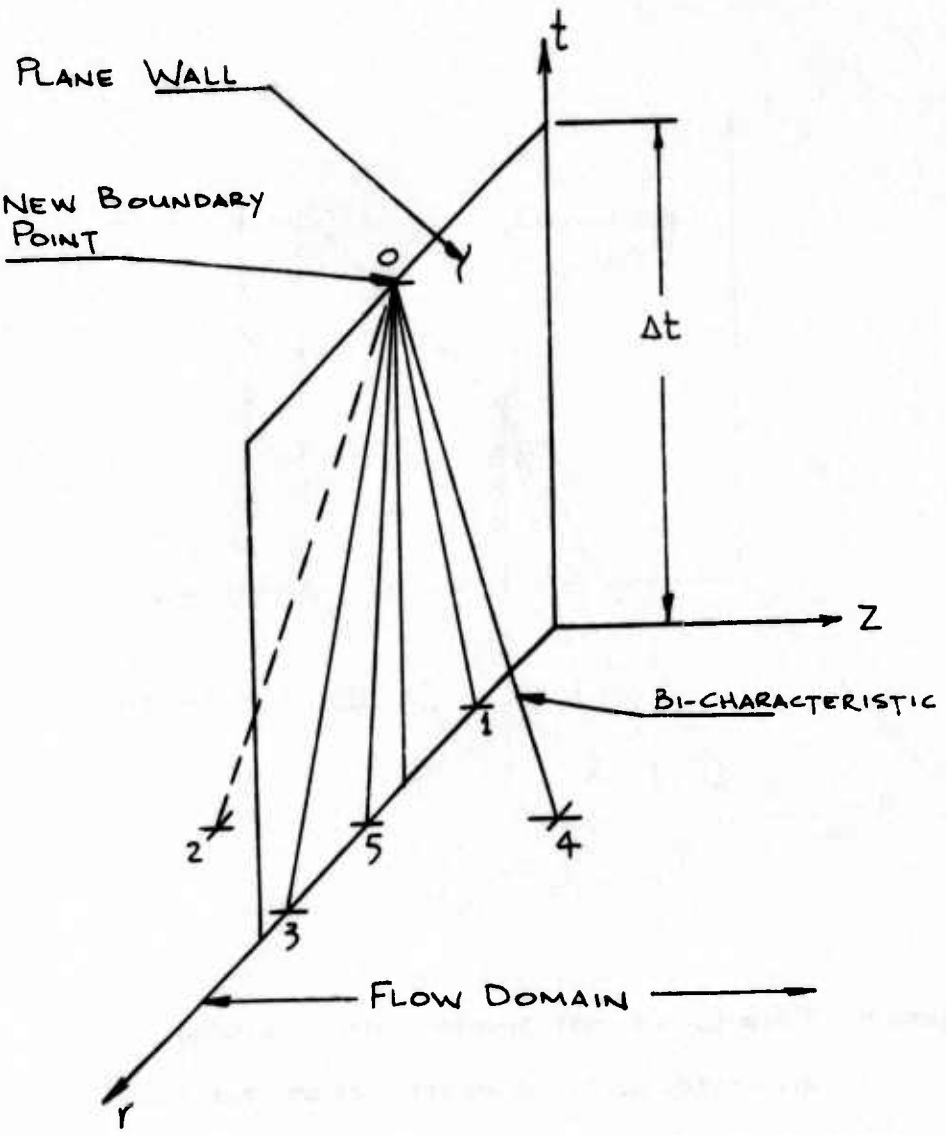


Figure 7. Schematic of plane-wall boundary, showing projection of bicharacteristics and particle path to the initial time plane.

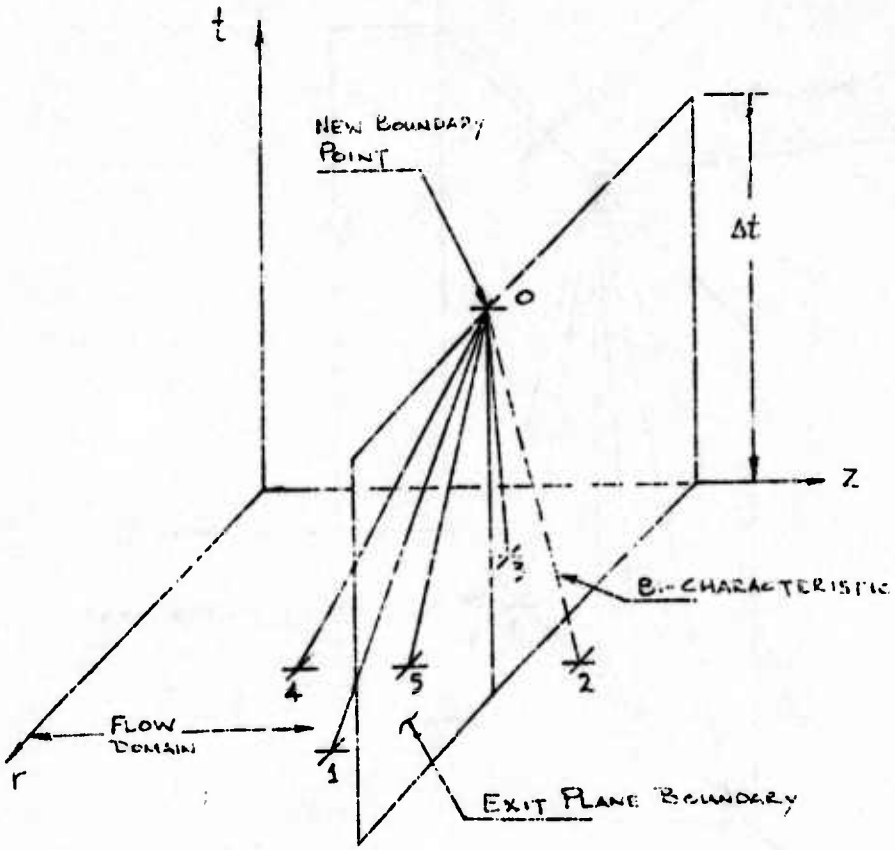


Figure 8. Schematic of exit boundary plane, showing projection of bicharacteristics and particle path to the initial time plane.

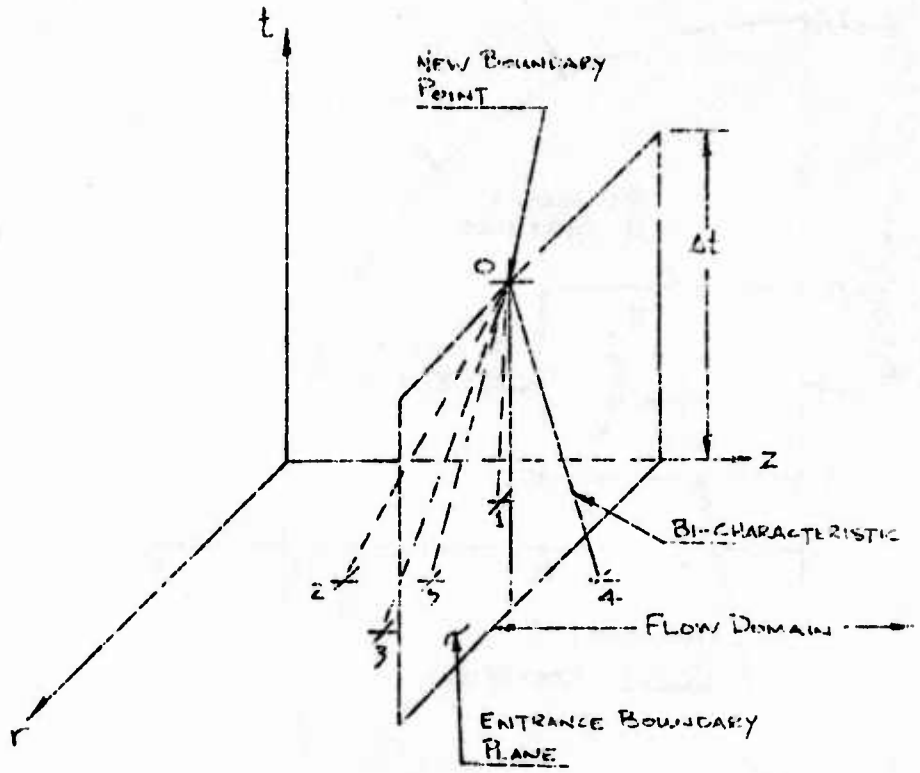


Figure 9. Schematic of entrance boundary plane, showing projection of bicharacteristics and particle path to the initial time plane.

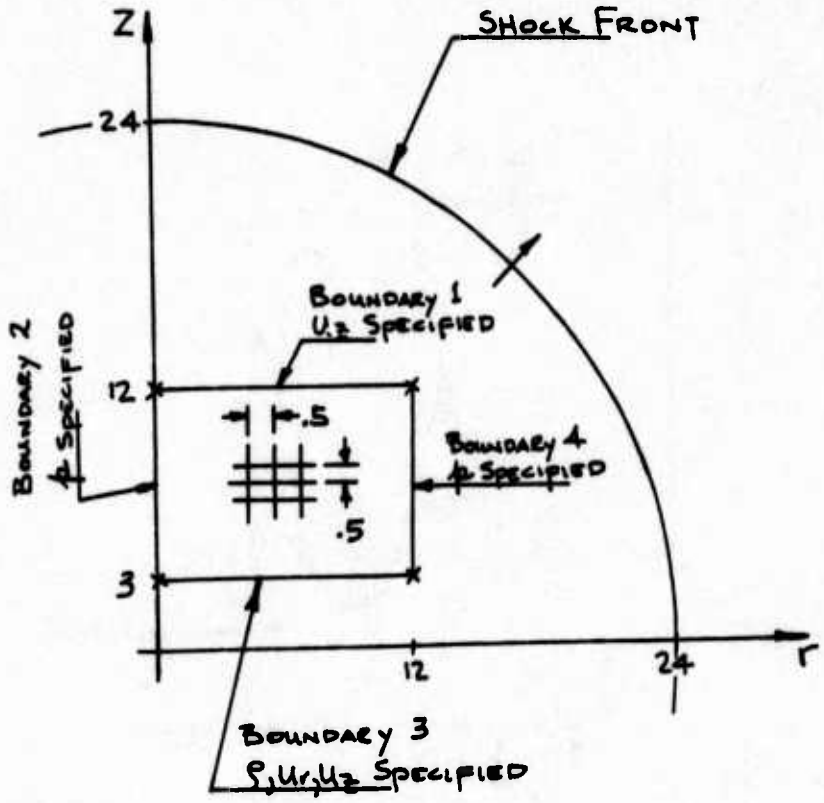


Figure 10. Schematic of the rectangular domain within the blast wave and shock front at the initial time t = 159.1 μsec. Initial energy = 6.887025×10^6 joules/m (1.5482649×10^6 ft-lb/ft).

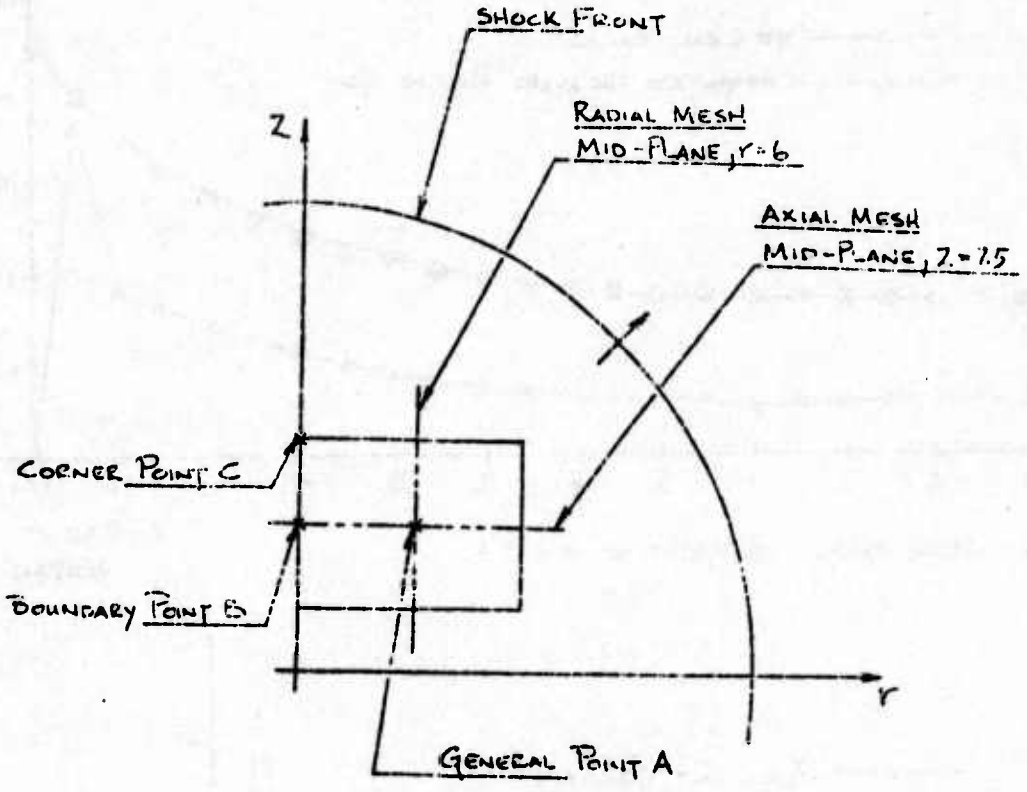
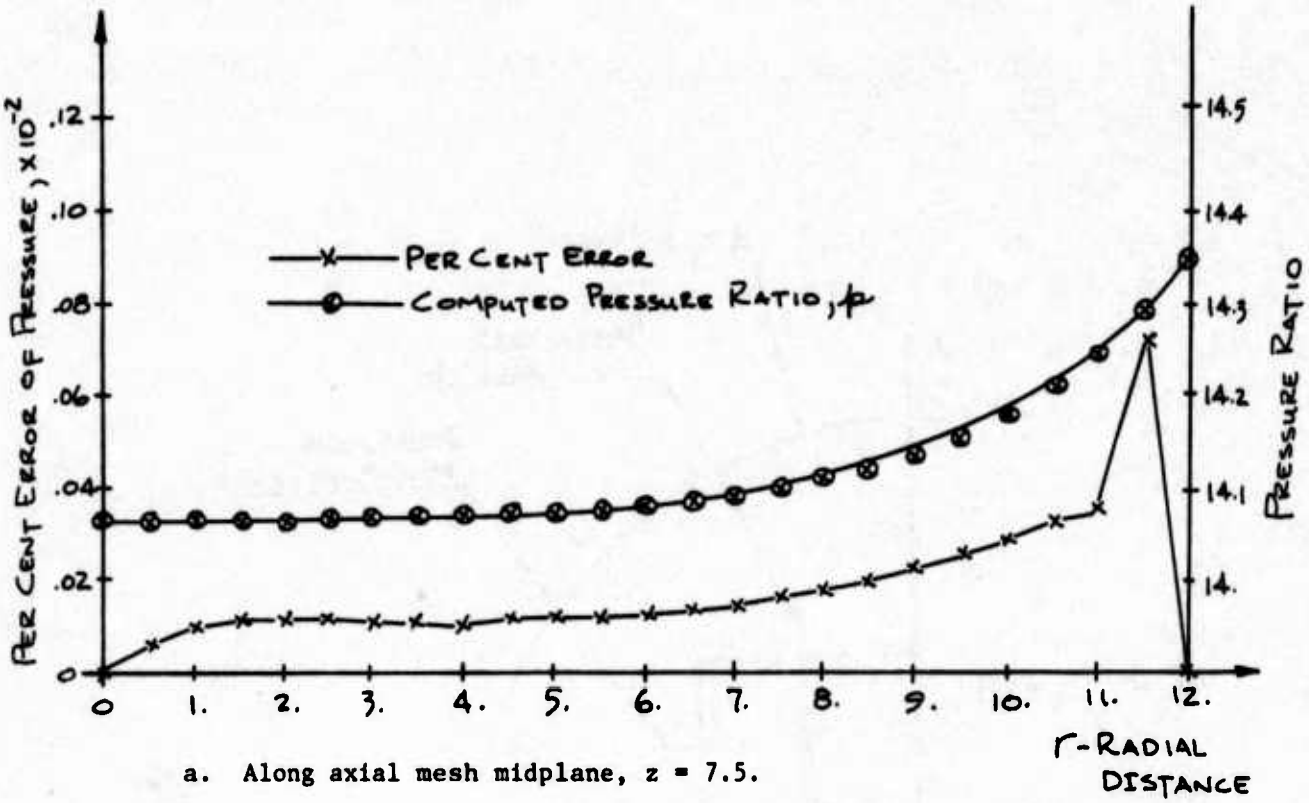
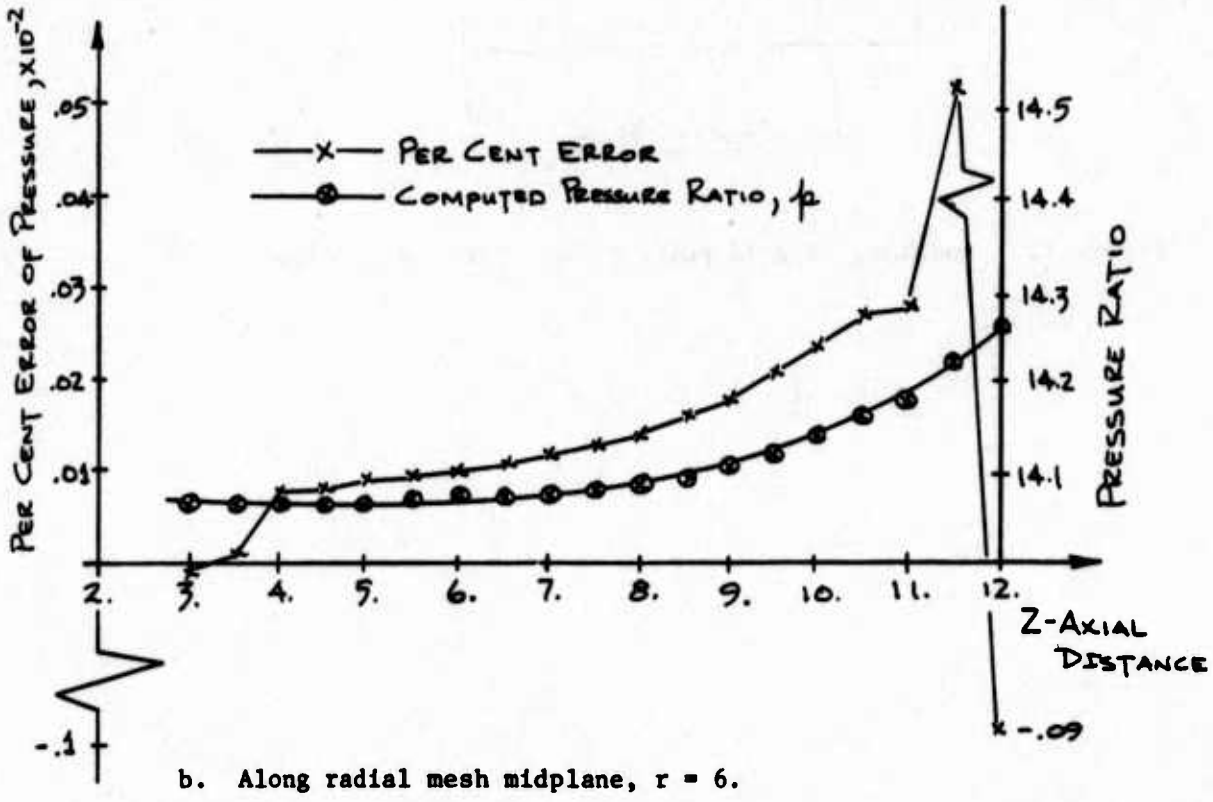


Figure 11. Location of grid points where error is plotted.



a. Along axial mesh midplane, z = 7.5.



b. Along radial mesh midplane, r = 6.

Figure 12. Data at 50th time plane

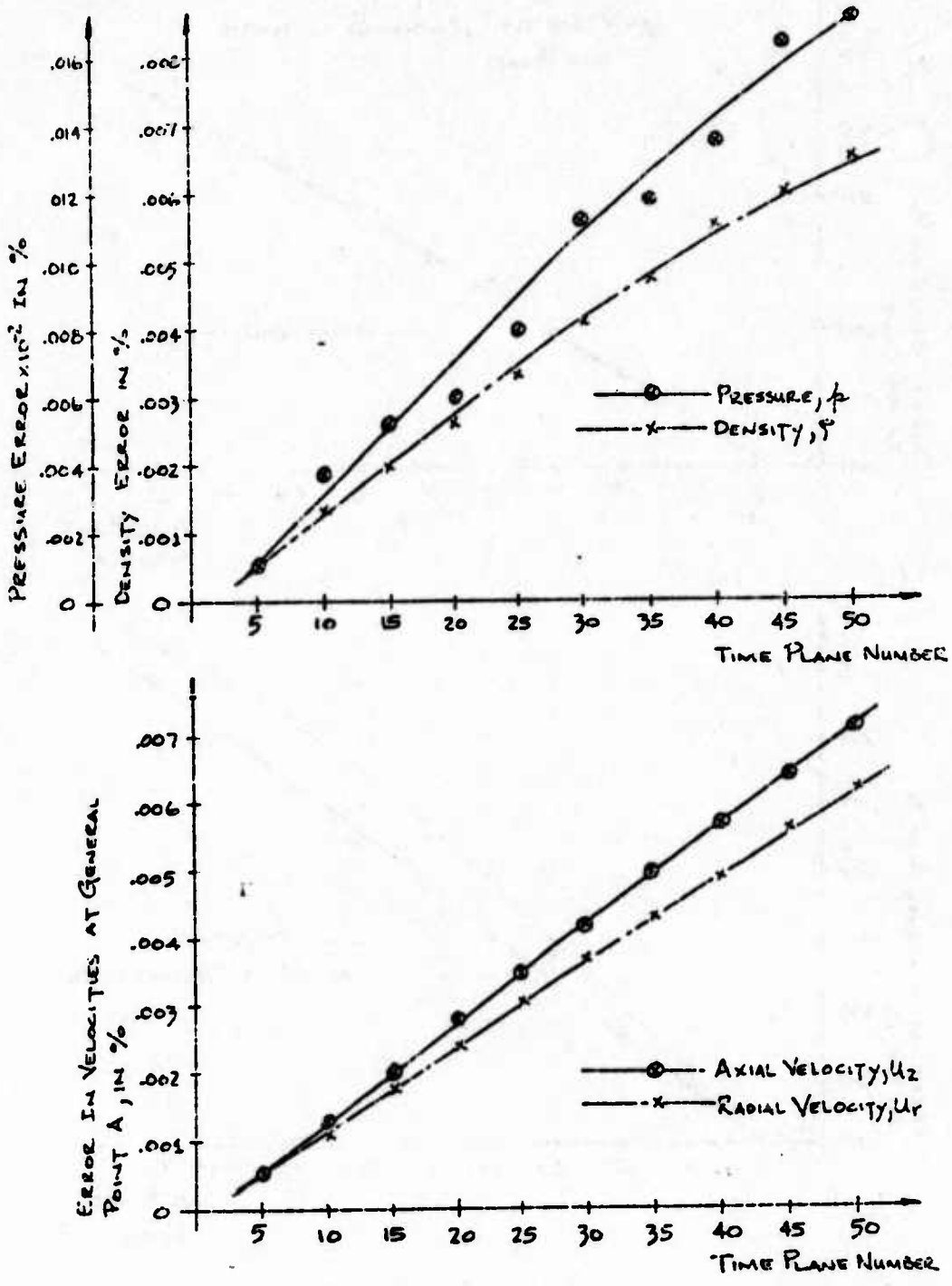


Figure 13. Data at general point A.

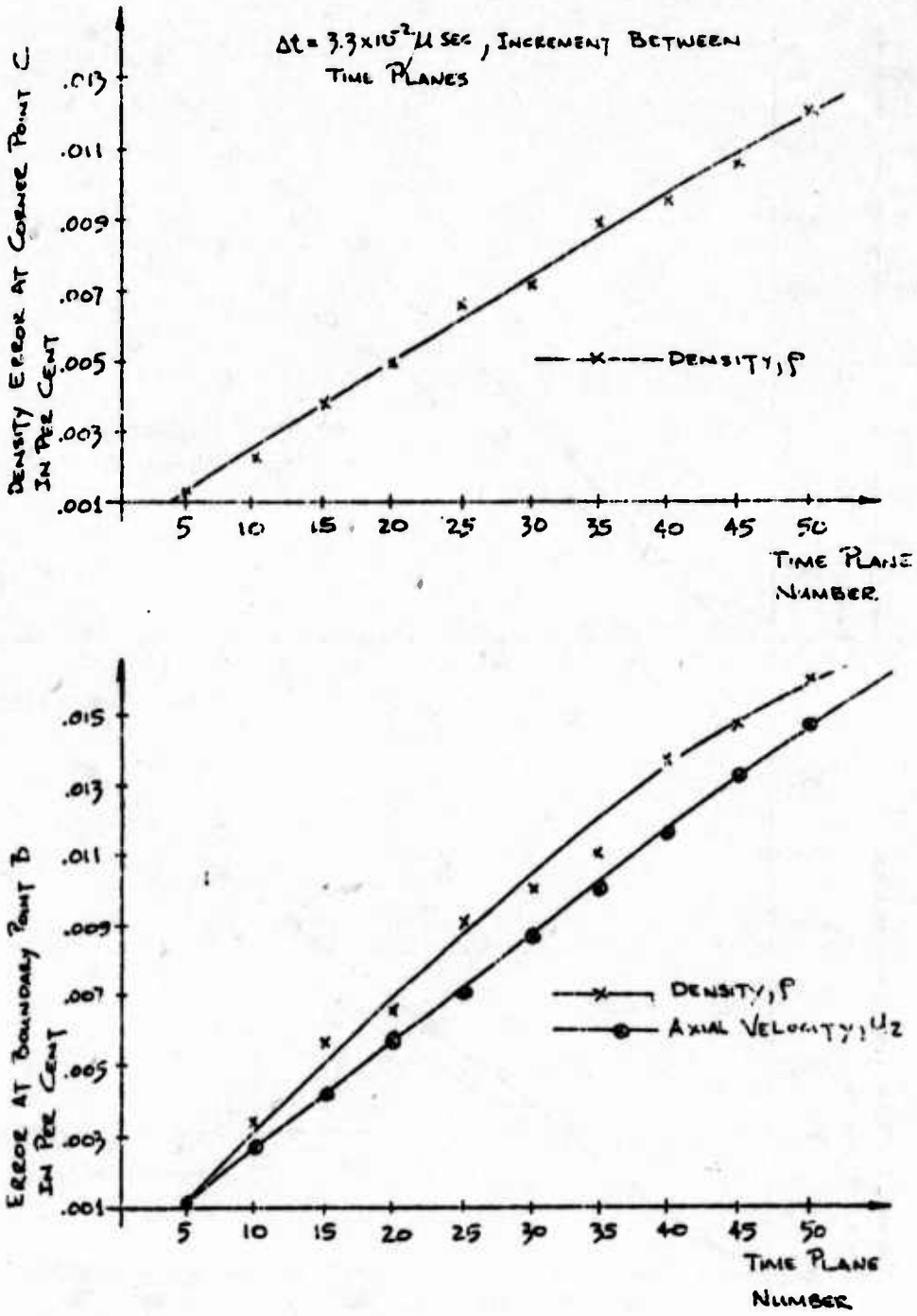


Figure 14. Error vs. time at boundary point B and corner point C.

Appendix A. Sauerwein's Finite-Difference Scheme

As presented in the main text, the initial-value problem may be solved using Eqs. (5), (6), and (21) for the five unknowns p, ρ, a, u_r, and u_z. According to Sauerwein's method, three finite-difference equations are generated from Eq. (21), for the angles $\phi = 0,\ 2\pi/3,\ 4\pi/3$. The partial derivatives are eliminated by the requirement that the derivatives are continuous along the three bicharacteristics.

By writing Eq. (21) in finite-difference form we obtain the following:

$$p_0 - p_i + \tfrac{1}{2}\{\rho_0 a_0 \cos\phi_i + \rho_i a_i \cos\theta_i\}[u_{r_0} - u_{r_i}] + \tfrac{1}{2}\{\rho_0 a_0 \sin\phi_i + \rho_i a_i \sin\theta_i\}[u_{z_0} - u_{z_i}] = \tfrac{1}{2}\rho_0 a_0^2 s_{0_i}\Delta t + \tfrac{1}{2}\rho_i a_i^2 s_i \Delta t\ ,\quad i = 1,2,3 \tag{A.1}$$

$$s_{0_i} = K\frac{u_{r_0}}{r_0} + \sin^2\phi_i\left(\frac{\partial u_r}{\partial r}\right)_0 + \sin\phi_i\cos\phi_i\left\{\left(\frac{\partial u_r}{\partial z}\right)_0 + \left(\frac{\partial u_z}{\partial r}\right)_0\right\} + \cos^2\phi_i\left(\frac{\partial u_z}{\partial z}\right)_0$$

$$s_i = K\frac{u_{r_i}}{r_i} + \sin^2\theta_i\left(\frac{\partial u_r}{\partial r}\right)_i + \sin\theta_i\cos\theta_i\left\{\left(\frac{\partial u_r}{\partial z}\right)_i + \left(\frac{\partial u_z}{\partial r}\right)_i\right\} + \cos^2\theta_i\left(\frac{\partial u_z}{\partial z}\right)_0$$

and θ_1 is defined in Appendix B. The subscripts 1, 2, and 3 refer to properties at the base points of the bicharacteristics, the subscript 4 to the base point of the particle path line, and the subscript 0 to the new point (t_0, r_0, z_0).

The compatibility equation along the particle path, Eq. (5), is also written in finite-difference form.

$$p_o - p_4 = -\frac{1}{2}\rho_o a_o^2 \left\{K\frac{u_{r_o}}{r_o} + \left(\frac{\partial u_r}{\partial r}\right)_o + \left(\frac{\partial u_z}{\partial z}\right)_o\right\}\Delta t - \frac{1}{2}\rho_4 a_4^2 \left\{K\frac{u_{r_4}}{r_4} + \left(\frac{\partial u_r}{\partial r}\right)_4 + \left(\frac{\partial u_z}{\partial z}\right)_4\right\}\Delta t \tag{A.2}$$

By requiring the derivatives of the particle velocities to be continuous along the bicharacteristics we obtain the following six equations.

$$u_{r_o} - u_{r_i} = \frac{1}{2}\left\{\left(\frac{\partial u_r}{\partial r}\right)_o + \left(\frac{\partial u_r}{\partial r}\right)_i\right\}[r_o - r_i] + \frac{1}{2}\left\{\left(\frac{\partial u_r}{\partial z}\right)_o + \left(\frac{\partial u_r}{\partial z}\right)_i\right\}[z_o - z_i] + \frac{1}{2}\left\{\left(\frac{\partial u_r}{\partial t}\right)_o + \left(\frac{\partial u_r}{\partial t}\right)_i\right\}\Delta t$$

$$u_{z_o} - u_{z_i} = \frac{1}{2}\left\{\left(\frac{\partial u_z}{\partial r}\right)_o + \left(\frac{\partial u_z}{\partial r}\right)_i\right\}[r_o - r_i] + \frac{1}{2}\left\{\left(\frac{\partial u_z}{\partial z}\right)_o + \left(\frac{\partial u_z}{\partial z}\right)_i\right\}[z_o - z_i] + \frac{1}{2}\left\{\left(\frac{\partial u_z}{\partial t}\right)_o + \left(\frac{\partial u_z}{\partial t}\right)_i\right\}\Delta t \quad , i = 1,2,3 \tag{A.3}$$

Equations (A.1), (A.2), (A.3) and (6) constitute a system of eleven equations which may be solved for the eleven dependent variables:

$$p_o,\ a_o,\ \rho_o,\ u_{r_o},\ u_{z_o},\ (\partial u_r/\partial r)_o,\ (\partial u_r/\partial z)_o,\ (\partial u_r/\partial t)_o,$$

$$(\partial u_z/\partial r)_o,\ (\partial u_z/\partial z)_o,\ (\partial u_z/\partial t)_o.$$

Appendix B

Derivation of the Bicharacteristic Angle, θ_i

When the characteristic compatibility equations (21) were derived, it was determined that the dependent variables varied along the bicharacteristic curves. Since these are curves, and not straight lines, the value of the coordinate angle δ with respect to the negative r-axis (Fig. 5 and Fig. B1) may also vary along the curve between the apex of the mach cone and the base point.

We can express the relationship between r and z for any point on the characteristic conoid parametrically as

$$\begin{aligned} r &= r(\phi,\tau) \\ z &= z(\phi,\tau) \end{aligned} \tag{B.1}$$

where $\tau = t-t_0$ and ϕ is defined as the value of the coordinate δ at the cone's apex.

By using the relation

$$\frac{\partial \beta_1}{\partial s} = \frac{\partial \beta_1}{\partial r}\frac{\partial r}{\partial s} + \frac{\partial \beta_1}{\partial z}\frac{\partial z}{\partial s} + \frac{\partial \beta_1}{\partial t}\frac{\partial t}{\partial s}$$

where s is taken in the direction ϕ along the characteristic cone, and the previous result

$$\nabla\beta_1 = \left\{ \frac{\cos\delta}{[1+(q+a)^2]^{1/2}}, \frac{\sin\delta}{[1+(q+a)^2]^{1/2}}, -\frac{q+a}{[1+(q+a)^2]^{1/2}} \right\}$$

we can obtain a relationship governing δ along the bicharacteristic curves

$$\frac{\partial r/\partial\phi}{\partial z/\partial\phi} = -\frac{\partial\beta_1/\partial z}{\partial\beta_1/\partial r} = -\tan\delta \tag{B.2}$$

Referring to Fig. B1, we wish to obtain an expression for the angle θ, which is the value of the angular coordinate δ at the base of the cone. In order to accomplish this, we will write a Taylor's expansion for the bicharacteristic curve in the direction of constant ϕ. When the expression for θ is obtained, a particular bicharacteristic curve will be designated by a subscript i.

The dependent variables u_r, u_z, a, and ϕ may be expanded in a Taylor series about the point r_0, z_0, t_0, in the direction of the bicharacteristic.

$$u_r = u_{r_0} - \frac{\partial u_r(\phi, t_0)}{\partial t}\tau + \cdots$$

$$u_z = u_{z_0} - \frac{\partial u_z(\phi, t_0)}{\partial t}\tau + \cdots$$

$$a = a_0 - \frac{\partial a(\phi, t_0)}{\partial t}\tau + \cdots \qquad \text{(B.3)}$$

$$\delta = \phi - \frac{\partial \delta(\phi, t_0)}{\partial t}\tau + \cdots$$

Substituting $\bar{u}_r = \frac{\partial u_r}{\partial t}$, $\bar{u}_r = \frac{\partial u_z}{\partial t}$, $\bar{a} = \frac{\partial a}{\partial t}$, $\bar{\delta} = \frac{\partial \delta}{\partial t}$

and neglecting second-order terms we obtain

$$u_r = u_{r_0} - \bar{u}_r\tau$$

$$u_z = u_{z_0} - \bar{u}_z\tau \qquad \text{(B.4)}$$

$$a = a_0 - \bar{a}\tau$$

$$\delta = \phi - \bar{\delta}\tau$$

Equations (B.4) are now substituted into the bicharacteristic equations (22),

$$\frac{dr}{dt} = u_{r_0} - \bar{u}_r\tau + (a_0 - \bar{a}\tau)\cos(\phi - \bar{\delta}\tau) \qquad \text{(B.5)}$$

$$\frac{dz}{dt} = u_{z_0} - \bar{u}_z\tau + (a_0 - \bar{a}\tau)\sin(\phi - \bar{\delta}\tau)$$

If this result is expanded and the small-angle approximation is used, we obtain

$$\frac{dr}{dt} = u_{r_o} + a_o \cos\phi - \{\bar{u}_r + \bar{a}\cos\phi - a_o \bar{\delta} \sin\phi\}\tau + \text{second order} \tag{B.6}$$

$$\frac{dz}{dt} = u_{z_o} + a_o \sin\phi - \{\bar{u}_z + \bar{a}\sin\phi + a_o \bar{\delta} \cos\phi\}\tau + \text{second order}$$

which becomes, upon integration,

$$r = r_o - \{u_{r_o} + a_o \cos\phi\}\tau + \tfrac{1}{2}\{\bar{u}_r + \bar{a}\cos\phi - a_o \bar{\delta} \sin\phi\}\tau^2 + \cdots \tag{B.7}$$

$$z = z_o - \{u_{z_o} + a_o \sin\phi\}\tau + \tfrac{1}{2}\{\bar{u}_z + \bar{a}\sin\phi + a_o \bar{\delta} \cos\phi\}\tau^2 + \cdots$$

We now rewrite eq. (B.2) as

$$\sin\delta \frac{\partial z}{\partial \phi} = -\cos\delta \frac{\partial r}{\partial \phi} \tag{B.8}$$

and substitute eqs. (B.4) and (B.7) to get

$$a_o \bar{\delta} = -\sin\phi \frac{\partial \bar{u}_z}{\partial \phi} - \cos\phi \frac{\partial \bar{u}_r}{\partial \phi} - \frac{\partial \bar{a}}{\partial \phi} \tag{B.9}$$

which may in turn be integrated with respect to time to yield

$$a_o \bar{\delta}\tau = -\sin\phi \frac{\partial u_z}{\partial \phi} - \cos\phi \frac{\partial u_r}{\partial \phi} - \frac{\partial q}{\partial \phi} \tag{B.10}$$

Using eq. (B.7) to expand this result and utilizing the Taylor expansion for the angle $\theta_i = \delta$, and $\tau = -\Delta t$:

$$\theta_i = \phi_i + \Delta t \left\{-\sin\phi_i \left(\frac{\partial q}{\partial r}\right)_i + \cos\phi_i \left(\frac{\partial q}{\partial z}\right)_i + \cos^2\phi_i \left(\frac{\partial u_r}{\partial z}\right)_i \right. \tag{B.11}$$
$$\left. - \sin\phi_i \cos\phi_i \left\{\left(\frac{\partial u_r}{\partial r}\right)_i - \left(\frac{\partial u_z}{\partial z}\right)_i\right\} - \sin^2\phi_i \left(\frac{\partial u_z}{\partial r}\right)_i \right.$$

where the subscript i denotes the particular bicharacteristic and the partial derivatives are evaluated at time τ. Equation (B.11) will allow the variation of the bicharacteristic angle θ_i to be computed in terms of the derivatives of the flow properties at the base of the characteristic cone.

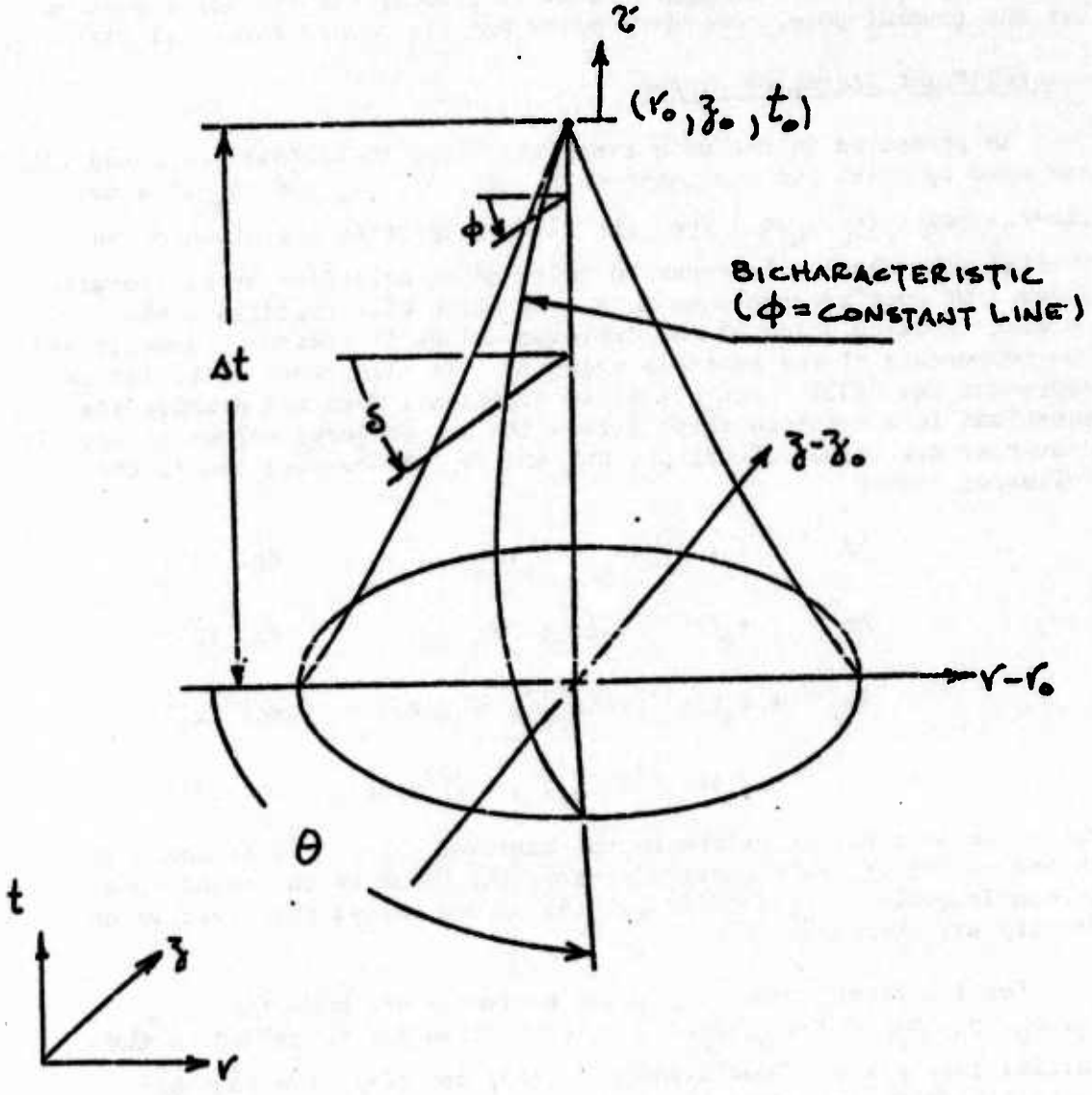


Figure B1. Diagram of Bicharacteristic Angles

Appendix C

Iteration Equations

The purpose of this appendix is to present the iteration schemes for the general point, boundary point and the corner point calculations.

General-Point Iteration Scheme

As presented in the main text, Eqs. (6), (33), (34), (35) and (36) are used to solve for the unknowns a_0, $\Delta\rho$, Δp, Δu_r and Δu_z at a new general point (r_0, z_0, t_0) when the fluid properties are given on an initial time plane. In order to solve these equations by an iteration scheme, we must arrange them in a form which will facilitate the successive calculation of the improved values of pressure, density and the components of the particle velocity. To illustrate this, let us represent Eqs. (33) through (36) in functional form and arrange the equations in a sequence which allows the new property values to rapidly converge; new values of Δp, $\Delta\rho$, Δu_r and Δu_z can be obtained in the following manner

$$\Delta p^{(1)} = f_1(\Delta p, \Delta u_r, \Delta u_z, \Delta\rho) \qquad \text{Eq. (35)}$$

$$\Delta u_r^{(1)} = f_2(\Delta p^{(1)}, \Delta u_r, \Delta u_z, \Delta\rho) \qquad \text{Eq. (33)}$$

$$\Delta u_z^{(1)} = f_3(\Delta p^{(1)}, \Delta u_r^{(1)}, \Delta u_z, \Delta\rho) \qquad \text{Eq. (34)}$$

$$\Delta\rho^{(1)} = f_4(\Delta p^{(1)}, \Delta u_r^{(1)}, \Delta u_z^{(1)}, \Delta\rho) \qquad \text{Eq. (36)}$$

where the superscript refers to the improved value. It is noted that, in the course of the iteration scheme, the value of the sound speed is continuously refined using Eq. (6) as new values for pressure and density are obtained.

For the first iteration, crude estimates are made for a_0, ρ_0, p_0, u_{r0} and u_{z0} based on their corresponding values in the initial time plane. Thus using Eqs. (22) and (24), the four bi-characteristics and the particle path are projected down to the initial time plane. The flow properties at the base points are obtained by interpolation and Eqs. (6), (33), (34), (35) and (36) are solved for improved values of a_0, Δp, $\Delta\rho$, Δu_r and Δu_z. This process is repeated until convergence is obtained within .0001 for each flow property. The reader is referred to subroutine GENCOR in the computer program for the general point iteration process.

Boundary-Point Iteration Equations

As presented in the main text Eqs. (6), (37), (38), and (39) are used to solve for the unknown flow properties p_0, ρ_0, a_0, u_{r0}, u_{z0} at a new point on the boundary of a plane wall or exit plane boundary. Again, in order to solve these equations by an iteration scheme, we arrange them in a form which will facilitate the calculation for the improved flow property values. To show this, let us represent Eqs. (37), (38), and (39) in functional form and arrange the equations in a sequence which allows a rapid convergence: new values of p_0, ρ_0, u_{r0} and u_{z0} can be obtained in the following manner

Prescribed Pressure Boundary

$$\Delta u_r^{(1)} = g_1(y, \Delta u_r, \Delta u_z, \Delta\rho) \qquad \text{Eq. (39)}$$

$$\Delta u_z^{(1)} = g_2(y, \Delta u_r^{(1)}, \Delta u_z, \Delta\rho) \qquad \text{Eq. (37)}$$

$$\Delta\rho^{(1)} = g_3(y, \Delta u_r^{(1)}, \Delta u_z^{(1)}, \Delta\rho) \qquad \text{Eq. (38)}$$

Prescribed Normal Velocity Boundary

$$\Delta p^{(1)} = h_1(\Delta p, \Delta u_r, x, \Delta\rho) \qquad \text{Eq. (37)}$$

$$\Delta u_r^{(1)} = h_2(\Delta p^{(1)}, \Delta u_r, x, \Delta\rho) \qquad \text{Eq. (39)}$$

$$\Delta\rho^{(1)} = h_3(\Delta p^{(1)}, \Delta u_r^{(1)}, x, \Delta\rho) \qquad \text{Eq. (38)}$$

where the superscript refers to the improved values of the flow properties and y is the prescribed pressure and x the prescribed normal velocity.

The iteration scheme for determining the flow properties at a boundary point is similar to that of the general point. The reader is referred to subroutine BONPT in the computer program for the boundary point iteration process. It is noted that the difference equation shown for computing $\Delta\rho^{(1)}$ for a prescribed normal velocity in subroutine BONPT is equivalent to Eq. (38).

Corner-Point Iteration Equations

For the example blast-wave problem, two corner-point iteration schemes were developed. These handled corner points A (at the intersection of boundaries 1 and 4) and B (boundaries 1 and 2) as shown in Fig. C1. The corner points C and D on the entrance boundary needed no iteration scheme, since all of the dependent variables must be specified in the boundary conditions. Points A and B differ in

the prescribed component of particle velocity with respect to the local coordinate system, and in the choice of compatibility equations.

For corner point A, the equations used to obtain a solution are the equation of state (6), two bicharacteristic equations (25) for i = 3 and 4 and energy - mass - conservation equations (31) and (32). Two boundary conditions are also specified to complete the system. After combining these equations to eliminate the derivatives, we may arrange them in a form suitable for iterative solution.

$$\Delta u_v^{(1)} = \{-2(Y-p_5) - \rho_0 a_0 [u_{r_3} - u_{r_5} + u_{z_4} - u_{z_5}] - \rho_0 a_0^2 K \frac{u_{r_0}}{r_0} \Delta t - [-\rho_0 a_0 + \rho_3 a_3 \sin\theta_3 + \rho_4 a_4 \sin\theta_4][Z - u_{z_5}] + 2R_5 - 2[R_3 + R_4]\} / \{-\rho_0 a_0 + \rho_3 a_3 \cos\theta_3 + \rho_4 a_4 \cos\theta_4\}$$

$$\Delta p^{(1)} = \{Y - p_5 - R_5 [\frac{a_0^2}{a_5^2} - 1]\} / a_0^2$$

where Y is the prescribed pressure and Z is the prescribed normal velocity along boundary 1.

For corner point B, a similar procedure is used to develop the iteration equations. The only difference from point A is that bicharacteristic equations (25) are written for i = 2 and 3. The difference equations are presented below.

$$\Delta u_v^{(1)} = \{-2[Y-p_5] - \rho_0 a_0 [u_{r_5} - u_{r_1} + u_{z_4} - u_{z_5}] + 2R_5 - \rho_0 a_0^2 K \frac{u_{r_0}}{r_0} \Delta t - [-\rho_0 a_0 + \rho_1 a_1 \sin\theta_1 + \rho_4 a_4 \sin\theta_4][Z - u_{z_5}] - 2[R_1 + R_4]\} / \{\rho_0 a_0 + \rho_1 a_1 \cos\theta_1 + \rho_4 a_4 \cos\theta_4\}$$

$$\Delta p^{(1)} = \{Y - p_5 - R_5 [\frac{a_0^2}{a_5^2} - 1]\} / a_0^2$$

where Y is the prescribed pressure and Z is the normal velocity.

The iteration scheme for determining the flow properties at a corner point is similar to the boundary point. The reader is referred to subroutine CORNPT in the computer program for the corner point iteration process.

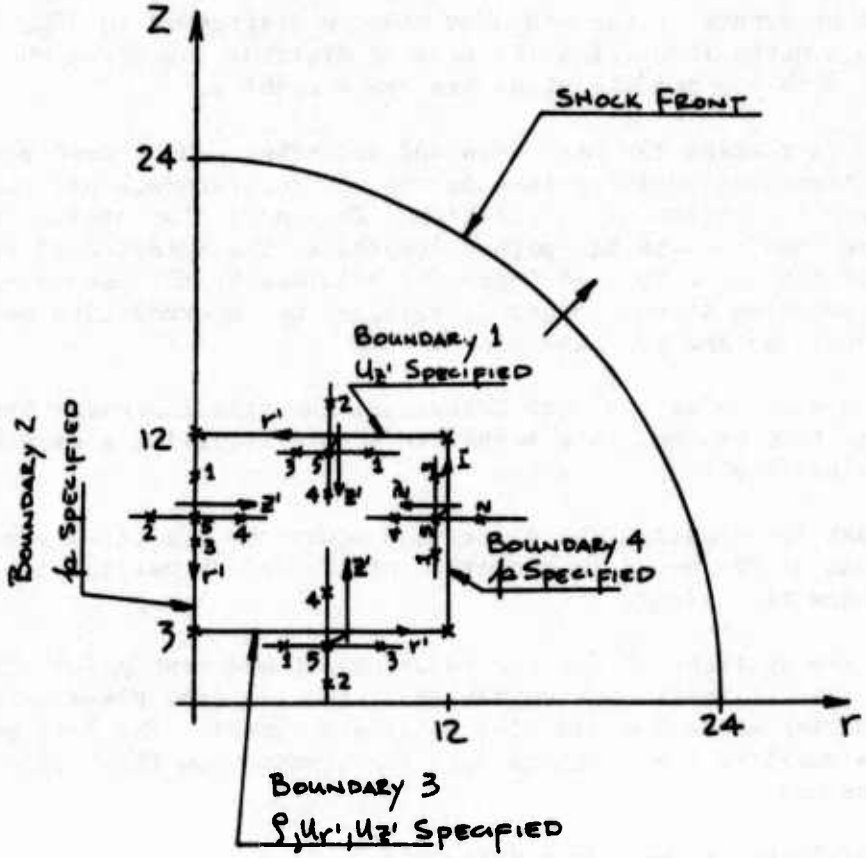


Figure C1. Schematic of Flow Field Showing Characteristic Base Points.

Appendix D - Limitations of the Computer Code and Input Preparation

Structure of the Computer Code

The basic structure of the computer code is diagrammed in Fig. D.1. Of the nine main parts into which the code is divided, the first two read in the problem data and nondimensionalize the variables.

The third part scans the mesh data and determines which mesh points are active or inactive; active points in the rectangular mesh are those which lie within the region of calculation. This part also decides which mesh points are "odd"; these are points located in the interior of the flow domain but too close to a boundary for solution by the general-point routine. The solution at odd points is obtained by interpolation between neighboring points on the new time plane.

The fourth section of the code determines the time increment between the old and new time planes, in a manner to insure stability according to the Courant-Friedrichs-Levy criterion.

In the next three parts, the difference equations are integrated to obtain solutions at the boundary corner points, boundary points and general points in the new time plane.

The last two sections obtain the solutions at odd mesh points by interpolation, then transfer the variables of the new time plane into the arrays where variables in the old time plane are stored. The last part also re-dimensionalizes the solution data for printout on those time planes selected by the user.

Limitations and Instructions for Modification

As listed in Appendix E, the code is limited to problems involving a 25 x 25 (or smaller) mesh, fewer than 10 boundaries each with not more than 20 coordinate points; tabular boundary-condition data may not exceed 20 entries. These maximum values may be enlarged by increasing the subscripts of the appropriate variables in the COMMON and DIMENSION statements wherever they appear within the code; however, this will also increase the core requirement of the computer code. An example is given on page 48.

The boundaries of a problem are numbered counter-clockwise starting in the upper-right-hand corner, and boundary data must be input in the same order. The entire flow domain must lie in the first quadrant of the

r,z coordinate system, and the flow entrance boundary must be on that side of the flow domain which is toward the r-axis (as in Fig. D.2). The general-point mesh coordinates are represented by r_{ij} and z_{ij}, where i is the axial index and j the radial index. The boundary-point coordinates are r_{bij} and z_{bij}, where i indicates the number of the boundary and j is the index along the boundary.

The program assumes uniform initial conditions. If the user chooses to specify other initial conditions, he may do so by inputting all the initial data at each mesh point. The general mesh-point coordinate data must begin in the lower-left-hand corner (nearest the origin) of the rectangular mesh. The axial rows increase monotonically along the z-axis, and the radial columns increase along the r-axis.

The manner of locating the boundary mesh points is determined by the angle θ between the boundary and the r-axis. If $-45^{\circ} < \theta < 45^{\circ}$ or $135^{\circ} < \theta \leq 225^{\circ}$, then the boundary mesh points are located at the intersections of the vertical lines in the mesh grid with the boundary (see Fig. D.2). For other values of the boundary angle, intersections at the horizontal mesh grid lines are used.

As listed in Appendix E, the code uses the equation of state for a perfect gas. To employ some other equation, the user must modify the function SOUND(ρ,p), which computes the sound speed from the density ρ and pressure p.

A variety of boundary conditions have already been implemented, including a constant pressure line, stationary solid boundary, free surface, and flow entrance boundary. Provision has also been made in the code for additional boundary conditions to be developed in the future; these include a moving solid boundary, contact discontinuity, and shock discontinuity. For completeness, input data coding information on accessing these currently non-functioning parts of the program has been included in the following pages.

If the user wishes to specify boundary conditions computed from an analytical solution, he must supply the subroutine EXACT(r,z,t), specify MCODE(I) = 1 for that boundary, and possibly modify parts of subroutines CORCOM and BCOMP. However, it is recommended that the boundary conditions be specified in tabular form.

Increasing the Code's Problem-Size Capacity

The changes necessary to relax the problem-size limitations discussed above will be illustrated by an example.

To solve a problem requiring a 26 x 27 mesh, 11 boundaries, 23 points along the longest boundary, and 28 entries in the boundary-condition data table, the following alterations must be made:

Change the unlabelled COMMON block, wherever it appears in the code, to

```
COMMON RM(26,27),ZM(26,27),P(26,27,2),A(26,27,2),RHO(26,27,2),
UR(26,27,2),UZ(26,27,2),NOR,NOZ,RBM(11,30,2),ZBM(11,30,2),
RB(11,24,2),ZB(11,24,2),THETA(11,30,2),PB(11,30,2),AB(11,30,2),
RHOB(11,30,2),URB(11,30,2),UZB(11,30,2),NACTVE(26,27),NODD(26,
27),NBACT(11,30,2),NBODD(11,30,2),ITYPE(11,30,2),INUM(11,30,2),
LBPT(11),IAXES(11,30)
```

Change the COMMON block labelled BOUND, wherever it appears, to

```
COMMON/BOUND/JCODE(11),KCODE(11,3),LCODE(11),MCODE(11),BTIME
(11,28,3),BCON(11,23,28),BSPCE(11,23),ACON(11,23,28),CCON
(11,23,28)
```

Instructions for Use of Code

This appendix presents the computer code input data and test case.

The input consists of the following items, where * indicates fixed-point (integer) data:

1. Units identification card.
   Type 'SI' (left-justified) for S.I. Units or 'English' for English (f.p.s.) Units.

2. The first card-series contains the following input parameters: FORMAT (2(I3,2X), 6E10.0,/E10.0, I1, 4X,I2)

   a. NBP* - number of boundaries
   b. NTIME* - number of time-plane calculations
   c. DELR - radial mesh increment, m or ft
   d. DELZ - axial mesh increment, m or ft
   e. RINIT - initial radius, m or ft
   f. RFINAL - final radius, m or ft
   g. ZINIT - initial axial-direction value, m or ft
   h. ZFINAL - final axial-direction value, m or ft
   i. TIMEK - stability time constant; $\leq .5$
   j. KPRINT* - convergence-loop print indicator
      = 0, no intermediate print output
      = 1, intermediate print output
   k. MSKIP* - indicates number of time planes to be skipped in the printout

3. On the second card in the first card-series: FORMAT (6E10.0,I1)

   a. PINIT - reference pressure for non-dimensional purposes, N/m^2 or lb/ft^2
   b. RHØINT - reference density, kg/m^3, or slugs-per-cu-ft
   c. AINIT - reference sound speed, m/sec, or fps
   d. TIME - initial time of solution, sec
   e. WIDTH - reference dimension for non-dimensional purposes, m or ft
   f. KAXIS - indicates whether flow is two-dimensional or axi-symmetric
      = 0, two-dimensional flow
      = 1, axi-symmetric flow

g. NCØDE* - indicates if initial mesh data will be imput or whether the program will generate the data
= 0, the program will generate the mesh data (user must supply initializing subroutine INITIAL)
= 1, the mesh data will be input

4. The second card-series contains the boundary-mesh input data

a. The first card in this series contains the number of spatial points on a boundary and a boundary condition function indicator FORMAT (10(I2,4X)
1. LCØDE(I)* - number of spatial points on the boundary
2. MCØDE(I)* - boundary condition function indicator
= 0, boundary condition will be input
= 1, boundary condition will be computed analytically (user must supply SUBROUTINES EXACT and INITIAL)

b. The second card in this series contains the spatial coordinates of the boundary points. FORMAT (5E10.0)
1. RB(I,J) - coordinate of the jth point on the ith boundary in the radial direction.
2. ZB(I,J) - coordiante of the jth point on the ith boundary in the axial direction

NOTE: J varies from 1 to LCØDE(I) and I varies from 1 to NBP.

5. The third card-series contains the boundary condition data for the boundary mesh points. FØRMAT (10(I2,4X))

a. The first card specifies the type of boundary condition and the number of time values that the boundary condition is specified.
1. JCØDE(I)* - specifies type of boundary condition
= 1, constant pressure line
= 2, normal velocity line or stationary solid boundary
= 3, free surface
= 4, contact discontinuity
= 5, moving solid boundary
= 6, shock front
= 7, entrance boundary
2. KCØDE(I,K)*- specifies number of time values
= 0, indicates boundary condition independent of time

NOTE: K varies from 1 to 3 depending on the boundary condition.

b. The second card specifies the value of time for the boundary condition. FORMAT (5E10.0)
   1. BTIME(I,J,K) -time, seconds

   NOTE: K varies from 1 to KCØDE(I,1). If KCØDE(I,1) is zero, no card will be input.

c. The third card in this series specifies the value of the boundary condition at each point on the boundary as a function of time. FØRMAT(5E10.0)
   1. BCØN(I,J,K)- value is dimensionless. It is either dimensionalized with respect to the reference pressure or reference sound speed.

   NOTE: I varies from one to NBP, J varies from one to LCØDE(I), and K varies from one to KCØDE(I,1).

d. If JCØDE(I) is equal to 7, a second and third boundary condition are input, i.e., (b) and (c) are repeated.

6. The fourth card-series contains the general-mesh input data. FORMAT(7E10.0)

   a. The cards contain the radial and axial coordinates of the mesh point and corresponding properties.
      1. RM(I,J) - radial coordinate of the mesh point, m or ft
      2. ZM(I,J) - axial coordinate of the mesh point, m or ft
      3. UR(I,J,1) - dimensionless radial velocity ratio
      4. UZ(I,J,1) - dimensionless axial velocity ratio
      5. P(I,J,1) - dimensionless density ratio
      6. RHØ(I,J,1) - dimensionless density ratio
      7. A(I,J,1) - dimensionless sound speed ratio

      NOTE: This card set only appears if NCØDE is greater than zero. I varies from 1 to the number of rows and J varies from 1 to the number of columns.

7. The fifth card-series contains the boundary-mesh imput data.

   a. The cards contain the radial and axial coordinates of the boundary mesh point and the corresponding properties. The boundary mesh points should be input in the same direction as the boundary points in (4b).
      1. RBM(I,J,1) - radial coordinate of the boundary mesh point, m or ft
      2. ZBM(I,J,1) - axial coordinate of the boundary mesh point, m or ft

3. URB(I,J,1) - dimensionless radial velocity ratio
4. UZB(I,J,1) - dimensionless axial velocity ratio
5. PB(I,J,1) - dimensionless pressure ratio
6. RHØB(I,J,1)- dimensionless density ratio
7. AB(I,J,1) - dimensionless sound-speed ratio

Listing of Sample Problem Input Data

```
ENGLISH
004  005   .041666667.0416666670.0        1.0        0.25       1.0
.4         0    01
2116.8     0.0024497 11C0.0     C.0001591 2.0
 2   01                                                          FIRST BOUN.
1.0        1.0        0.C        1.0
02   01                                                          SEC. BCUN.
0.0        1.C        C.0        C.25
02   01                                                          THIRD BCUN.
0.0        0.25       1.0        C.25
02   01                                                          FOUR BCUN.
1.0        0.25       1.0        1.0
02                                                               FIRST BOUN.
01
07
01
```

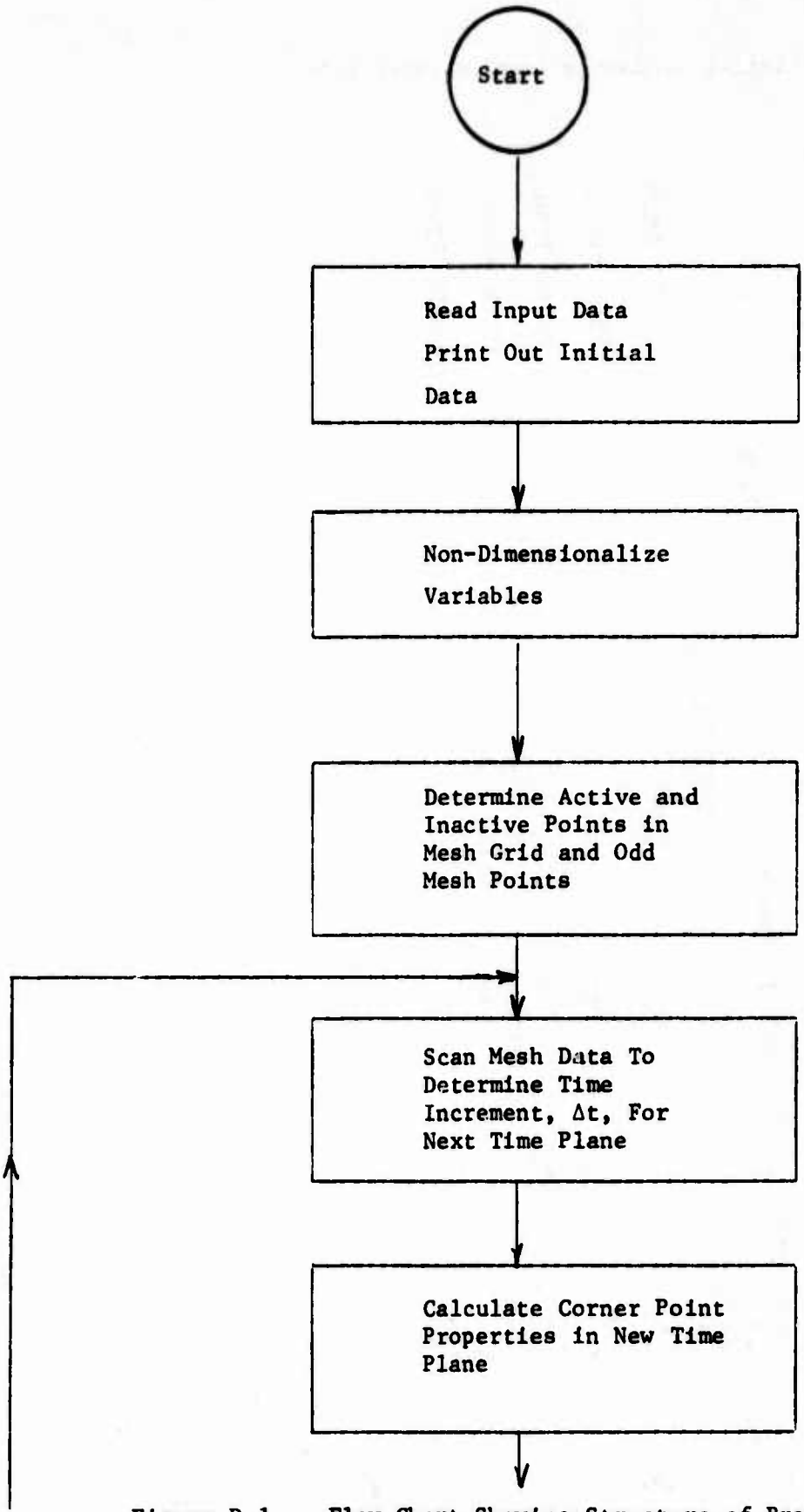


Figure D.1 Flow Chart Showing Structure of Program

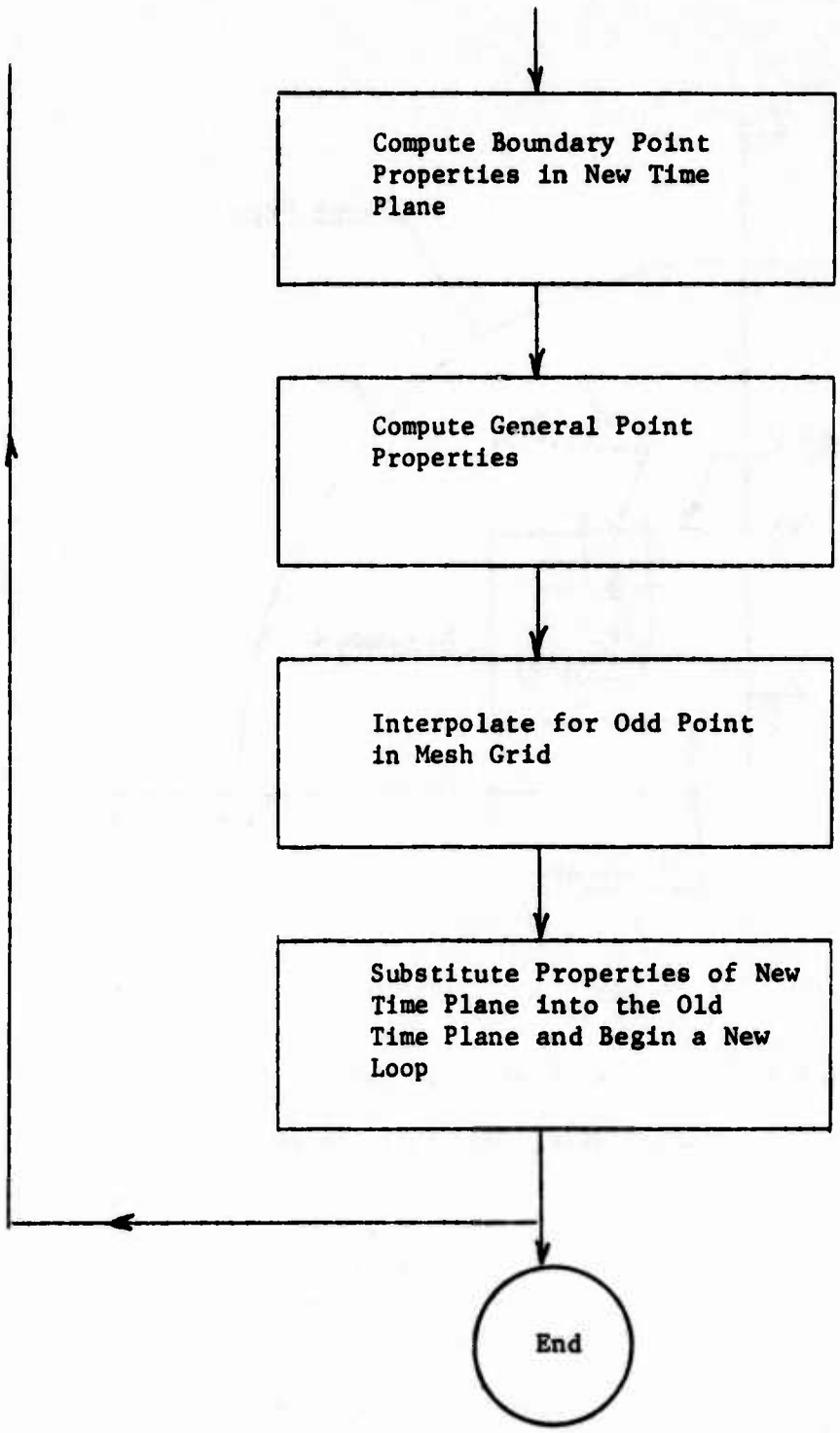


Figure D.1 Flow Chart Showing Structure of Program

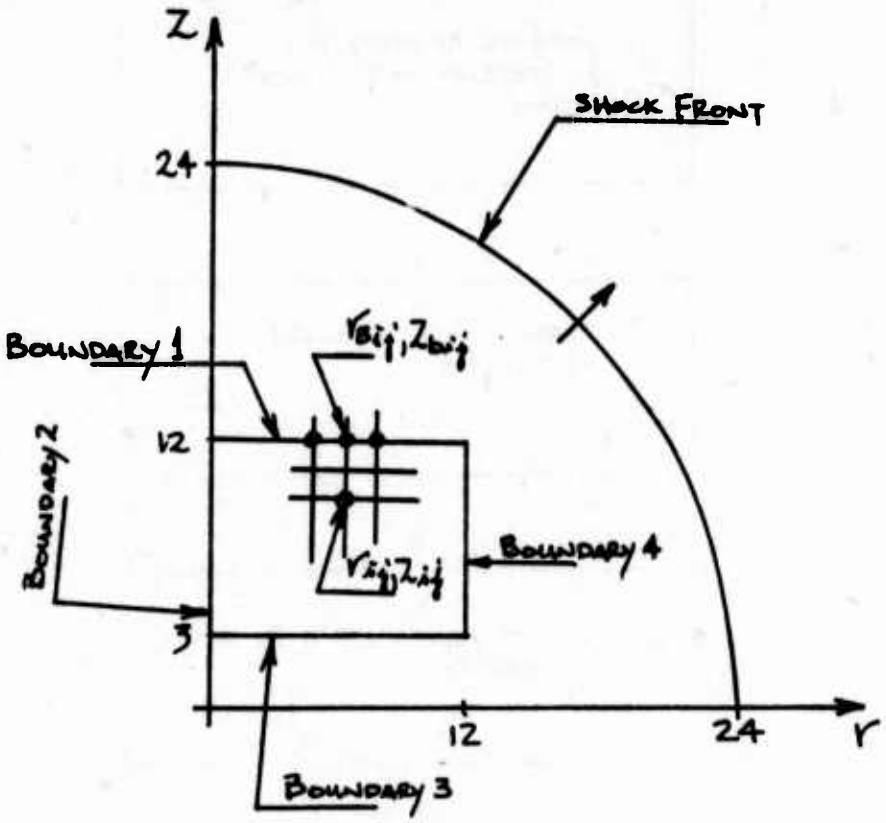


Figure D.2 Schematic of Blast Wave

Illustrating Mesh Coordinates

Appendix E

Listing of the

MCDU 43 Computer Code

Contents

```
      REAL KAXIS
      DIMENSIONNSI(6),NDAT(20),PROP1(30),X(30),ERROR(5)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/EXT/URR,UZZ,PRAT,RRAT,ARAT
      COMMON/STATE/PINIT,RHOINT,AINIT
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      PI=3.1415926535898
      DATANGAT/'  TI','ME  ','MILL','ISEC','PRES','SURE','NORM',' VEL',
     1'VELO','CITY','LB/S','Q-FT',' FT/','SEC ',' FT/','SEC ',
     2'DENS','ITY ','SL/C','U-FT'/
      DATANSI/'NT/S','Q-M ','  M/','SEC ','  M/','SEC '/
      DATA SI/'SI'/
  100 PRINT820C
      READ8001,SSI
      IF(SSI.NE.SI)GOTO102
      DO64I=1,6
      J=I+10
   64 NDAT(J)=NSI(I)
  102 READ(5,8COC)NBP,NTIME,DELR,DELZ,RINIT,RFINAL,ZINIT,
     .ZFINAL,TIMEK,KPRINT,MSKIP
C     READ INITIAL REFERENCE CONDITIONS
      READ(5,8022)PINIT,RHOINT,AINIT,TIME,WIDTH,KAXIS,NCODE
      CONST=PINIT/(RHOINT*AINIT*AINIT)
C     READ BOUNDARY DATA
      DO 110 I=1,NBP
      READ(5,8002)LCODE(I),MCODE(I)
      KSPCE=LCODE(I)
  110 READ(9,8021)((RB(I,J,1),ZB(I,J,1)),J=1,KSPCE)
C     SET UP BOUNDARIES
      DO 114 I=1,NBP
      KSPCE=LCODE(I)
      DO 114 J=1,KSPCE
      RB(I,J,2)=RB(I,J,1)
  114 ZB(I,J,2)=ZB(I,J,1)
C     READ BOUNDARY CONDITIONS
      DO 200 I=1,NBP
      READ(5,8002)JCODE(I),(KCODE(I,K),K=1,3)
      IF((JCODE(I).EQ.4).OR.(JCODE(I).EQ.6).OR.(JCODE(I).EQ.8))
     .GO TO 200
      IF(MCODE(I).EQ.1) GO TO 200
      KSPCE=LCODE(I)
      KTIME=KCODE(I,1)
      IF(KTIME.EQ.0)KTIME=1
      DO 202 K=1,KTIME
      IF(KCODE(I,1).GT.0)READ(5,8021)BTIME(I,K,1)
      READ(5,8021)(BCON(I,J,K),J=1,KSPCE)
  202 CONTINUE
      IF(JCODE(I).NE.7)GO TO 200
```

```
      KTIME=KCODE(I,2)
      DO 204 K=1,KTIME
      IF(KTIME.EQ.0)KTIME=1
      IF(KCODE(I,2).GT.0)READ(5,8021)BTIME(I,K,2)
      READ(5,8021)(ACON(I,J,K),J=1,KSPCE)
  204 CONTINUE
      KTIME=KCODE(I,3)
      IF(KTIME.EQ.0)KTIME=1
      DO 206 K=1,KTIME
      IF(KCODE(I,3).GT.0)READ(5,8021)BTIME(I,K,3)
      READ(5,8021)(CCON(I,J,K),J=1,KSPCE)
  206 CONTINUE
  200 CONTINUE
      CALL ADJLST
      NOR=IFIX((RFINAL-RINIT+DELR/10.)/DELR)+1
      NOZ=IFIX((ZFINAL-ZINIT+DELZ/10.)/DELZ)+1
C     READ IN MESH COORDINATES AND PROPERTIES
C     I REFERS TO THE ROW AND J REFERS TO THE COLUMN
      IF(NCODE.EQ.0)GO TO 400
      DO 302 I=1,NOZ
      DO 302 J=1,NOR
  302 READ(5,8020)RM(I,J),ZM(I,J),UR(I,J,1),UZ(I,J,1),P(I,J,1),RHO(I,J,1
     .),A(I,J,1)
      CALL SEARCH(DELR,DELZ,RINIT,ZINIT,1)
      CALL BSERCH(DELR,DELZ,1)
      DO 306 I=1,NBP
      KSPCE=LBPT(I)
      DO 306 J=1,KSPCE
  306 READ(5,8020)RBM(I,J,1),ZBM(I,J,1),URB(I,J,1),UZB(I,J,1),PB(I,J,1),
     .RHOB(I,J,1),AB(I,J,1)
      GO TO 410
  400 CONTINUE
C     SET INITIAL VALUES IN MESH
      CALL SEARCH(DELR,DELZ,RINIT,ZINIT,1)
      CALL BSERCH(DELR,DELZ,1)
      CALL INITIAL(DELR,DELZ,TIME)
  410 CONTINUE
C     WRITE OUT INPUT DATA
      WRITE(6,8230)
      WRITE(6,8232)
      WRITE(6,8270)NBP,NTIME,DELR,DELZ,RINIT,RFINAL,ZINIT,ZFINAL
      WRITE(6,8240)
      WRITE(6,8242)
      IF(SI.EQ.SSI)PRINT8245
      IF(SI.NE.SSI)PRINT8244
      WRITE(6,8280)PINIT,RHOINT,AINIT,WIDTH,TIMEK
      PRINT8003,KAXIS,NCODE,KPRINT,NBP,NTIME,NBT,NPLOT
      WRITE(6,8250)
      WRITE(6,8252)
      IF(SI.EQ.SSI)PRINT8255
      IF(SI.NE.SSI)PRINT8254
C     WRITE OUT BOUNDARY POINTS
      DO 700 I=1,NBP
      KSPCE=LCODE(I)
      WRITE(6,8256)I
  700 WRITE(6,8290)((L,RB(I,L,1),ZB(I,L,1)),L=1,KSPCE)
C     WRITE OUT BOUNDARY CONDITIONS
      WRITE(6,8400)
```

```
      DO 702 I=1,NBP
      WRITE(6,8220)
      NBP1=I+1
      KL=JCODE(I)
      GO TO(720,722,720,702,722,702,724),KL
  720 KJ=5
      KK=6
      GO TO 726
  722 KJ=7
      KK=13
      GO TO 726
  724 KJ=9
      KK=13
  726 CONTINUE
      KTIME=KCODE(I,1)
      KSPCE=LCODE(I)
      IF(MCODE(I).GT.0)GO TO 701
      IF(KTIME.EQ.0)GO TO 704
      DO 703 K=1,KTIME
      WRITE(6,8424)
      WRITE(6,8410)BTIME(I,K,1)
      WRITE(6,8420)I,NBP1,NDAT(KJ),NDAT(KJ+1)
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 706 J=1,KSPCE
  706 WRITE(6,8425)BSPCE(I,J),BCON(I,J,K)
  703 CONTINUE
      IF(JCODE(I).NE.7)GO TO 702
C     WRITE OUT SECOND BOUNDARY CONDITION
  728 CONTINUE
      KJ=17
      KK=18
      KTIME=KCODE(I,2)
      IF(KTIME.EQ.0)GO TO 705
      DO 730 K=1,KTIME
      WRITE(6,8424)
      WRITE(6,8410)BTIME(I,K,2)
      WRITE(6,8420)I,NBP1,NDAT(KJ)
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 732 J=1,KSPLE
  732 WRITE(6,8425)BSPCE(I,J),ACON(I,J,K)
  730 CONTINUE
  740 CONTINUE
      KJ=9
      KK=13
      KTIME=KCODE(I,3)
      IF(KTIME.EQ.0)GO TO 707
      DO 734 K=1,KTIME
      WRITE(6,8424)
      WRITE(6,8410)BTIME(I,K,3)
      WRITE(6,8420)I,NBP1,NDAT(KJ),NDAT(KJ+1)
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 736 J=1,KSPCE
  736 WRITE(6,8425)BSPCE(I,J),CCON(I,J,K)
  734 CONTINUE
      GO TO 702
  704 WRITE(6,8424)
      WRITE(6,8412)
      WRITE(6,8420)I,NBP1,NDAT(KJ),NDAT(KJ+1)
```

```
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 708 J=1,KSPCE
  708 WRITE(6,8425)BSPCE(I,J),BCON(I,J,1)
      IF(JCODE(I).NE.7)GO TO 702
      GO TO 728
  705 WRITE(6,8424)
      WRITE(6,8412)
      WRITE(6,8420)I,NBP1,NDAT(KJ),NDAT(KJ+1)
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 738 J=1,KSPCE
  738 WRITE(6,8425)BSPCE(I,J),ACON(I,J,1)
      GO TO 740
  707 WRITE(6,8424)
      WRITE(6,8412)
      WRITE(6,8420)I,NBP1,NDAT(KJ),NDAT(KJ+1)
      WRITE(6,8422)NDAT(KK),NDAT(KK+1)
      DO 742 J=1,KSPCE
  742 WRITE(6,8425)BSPCE(I,J),CCON(I,J,1)
      GO TO 702
  701 WRITE(6,8428)I
  702 CONTINUE
C     WRITE OUT MESH POINT COORDINATES AND PROPERTIES
      WRITE(6,8260)
      WRITE(6,8262)
      WRITE(6,8264)
      DO 710 I=1,NOZ
      DO 710 J=1,NOR
  710 WRITE(6,8300)I,J,RM(I,J),ZM(I,J),P(I,J,1),RHO(I,J,1),
     .A(I,J,1),UR(I,J,1),UZ(I,J,1)
C     WRITE OUT BOUNDARY DATA
      WRITE(6,8258)
      WRITE(6,8262)
      WRITE(6,8264)
      DO 776 I=1,NBP
      KSPCE=LBPT(I)
      DO 776 K=1,KSPCE
  776 WRITE(6,8300)I,K,RBM(I,K,1),ZBM(I,K,1),PB(I,K,1),RHOB(I,K,1),
     .AB(I,K,1),URB(I,K,1),UZB(I,K,1)
C     NON-DIMENSIONALIZE VARIABLES
      DELZ=DELZ/WIDTH
      DELR=DELR/WIDTH
      RINIT=RINIT/WIDTH
      ZINIT=ZINIT/WIDTH
      RFINAL=RFINAL/WIDTH
      ZFINAL=ZFINAL/WIDTH
      DO 750 I=1,NOZ
      DO 750 J=1,NOR
      RM(I,J)=RM(I,J)/WIDTH
  750 ZM(I,J)=ZM(I,J)/WIDTH
      DO 752 I=1,NBP
      KSPCE=LCODE(I)
      DO 752 K=1,KSPCE
      RB(I,K,1)=RB(I,K,1)/WIDTH
      ZB(I,K,1)=ZB(I,K,1)/WIDTH
      RB(I,K,2)=RB(I,K,2)/WIDTH
  752 ZB(I,K,2)=ZB(I,K,2)/WIDTH
      DO 754 I=1,NBP
      KSPCE=LBPT(I)
```

```
      DO 754 K=1,KSPCE
      RBM(I,K,1)=RBM(I,K,1)/WIDTH
      ZBM(I,K,1)=ZBM(I,K,1)/WIDTH
      RBM(I,K,2)=RBM(I,K,1)
  754 ZBM(I,K,2)=ZBM(I,K,1)
C     NON-DIMENSIONALIZE BOUNDARY CONDITIONS
      DO 760 I=1,NBP
      IF(MCODE(I).EQ.1)GO TO 760
      KJ=JCODE(I)
      KSPCE=LCODE(I)
      INDICE=0
      INDEX=0
      DO 764 J=1,KSPCE
      KTIME=KCODE(I,1)
      IF(KTIME.EQ.0)KTIME=1
      DO 766 K=1,KTIME
      IF((KCODE(I,1).GT.0).AND.(INDEX.EQ.0))
     .BTIME(I,K,1)=BTIME(I,K,1)*AINIT    /WIDTH
      GO TO(772,774,766,766,774,766,777),KJ
  772 BCON(I,J,K)=BCON(I,J,K)/PINIT
      GO TO 766
  774 BCON(I,J,K)=BCON(I,J,K)/AINIT
      GO TO 766
  777 BCON(I,J,K)=BCON(I,J,K)/AINIT
  766 CONTINUE
      INDEX=1
      IF(KJ.NE.7)GO TO 764
      KTIME=KCODE(I,2)
      IF(KTIME.EQ.0)KTIME=1
      DO 780 K=1,KTIME
      IF((KCODE(I,2).GT.0).AND.(INDICE.EQ.0))
     .BTIME(I,K,2)=BTIME(I,K,2)*AINIT    /WIDTH
  780 ACON(I,J,K)=ACON(I,J,K)/RHOINT
      KTIME=KCODE(I,3)
      IF(KTIME.EQ.0)KTIME=1
      DO 782 K=1,KTIME
      IF((KCODE(I,3).GT.0).AND.(INDICE.EQ.0))
     .BTIME(I,K,3)=BTIME(I,K,3)*AINIT    /WIDTH
  782 CCON(I,J,K)=CCON(I,J,K)/AINIT
      INDICE=1
  764 CONTINUE
  760 CONTINUE
C     BEGIN SOLUTION TO PROBLEM
C     FIRST DETERMINE TIME INCREMENT BY SCANNING DATA
C     SCAN THE Z DIRECTION FIRST
      IPASS=1
      TIME=TIME*AINIT    /WIDTH
      DO 790 L=2,NTIME
      CALL BSERCH(DELR,DELZ,2)
C     SCAN TIME FOR GENERAL POINTS IN Z DIRECTION
      DELTA=100.
      DO 800 K=3,NOZ
      K1=K-2
      DO 800 J=1,NOR
      IF((NACTVE(K,J).EQ.0).OR.(NACTVE(K1,J).EQ.0))GO TO 800
      CALL SCAN(ZM(K,J),ZM(K1,J),UZ(K1,J,1),A(K1,J,1),
     .UZ(K,J,1),A(K,J,1),DELTA,CMIN)
  800 CONTINUE
```

```
C       SCAN THE RADIAL DIRECTION
        DO 810 K=1,NOZ
        DO 810 J=3,NOR
        J1=J-2
        IF((NACTVE(K,J1).EQ.0).OR.(NACTVE(K,J).EQ.0)) GO TO 810
        CALL SCAN(RM(K,J),RM(K,J1),UR(K,J,1),A(K,J,1),UR(K,J1,1),
       .A(K,J1,1),DELTA,DMIN)
  810   CONTINUE
C       SCAN BOUNDARIES FOR TIME INCREMENT
        DO 820 I=1,NBP
        KSPCE=LBPT(I)
        DO 820 K=3,KSPCE
        K1=K-2
        IF((NBACT(I,K1,1).EQ.0).OR.(NBACT(I,K,1).EQ.0))GO TO 820
        ANGLE=ATAN2((ZBM(I,K,1)-ZBM(I,K1,1)),
       .(RBM(I,K,1)-RBM(I,K1,1)))
        RPR1=0.
        RPR2=SQRT(((ZBM(I,K,1)-ZBM(I,K1,1))**2)+
       .((RBM(I,K,1)-RBM(I,K1,1))**2))
        UR1=URB(I,K1,1)*COS(ANGLE)+UZB(I,K1,1)*SIN(ANGLE)
        UR2=URB(I,K ,1)*COS(ANGLE)+UZB(I,K ,1)*SIN(ANGLE)
        CALL SCAN(RPR2,RPR1,UR1,AB(I,K1,1),UR2,AB(I,K,1),
       .DELTA,DMIN)
  820   CONTINUE
        WRITE(6,8600)DELTA,DMIN
C       MULTIPLY TIME BY CONSTANT
        DELTA=TIMEK*DELTA
        WRITE(6,8600)DELTA
C       INTEGRATE UNSTEADY EQUATIONS TO THE NEW TIME PLANE
C       INCREMENT IN THE AXIAL DIRECTION FOR THE OUTSIDE
C       LOOP AND THEN THE RADIAL DIRECTION FOR THE INSIDE LOOP
C       CONSIDER THE FIRST ROW BOUNDARY
        TIME=TIME+DELTA
C       COMPUTE CORNER POINTS
        DO 1202 I=1,NBP,2
        DO 1202 LI=1,2
        GO TO(1210,1220),LI
  1210  CONTINUE
        J=1
        IBOUND=3
        IF(I.EQ.3)IBOUND=6
        K=I-1
        IF(K.LT.1)K=NBP
        M=LBPT(K)
        GO TO 1230
  1220  CONTINUE
        J=LBPT(I)
        IBOUND=2
        IF(I.EQ.3)IBOUND=5
        K=I+1
        IF(K.GT.NBP)K=1
        M=1
  1230  CONTINUE
        CALL CORNPT(I,J,CONST,KAXIS,TIME,DELTA,KPRINT,IBOUND)
        PB(K,M,2)=PB(I,J,2)
        PB(K,M,2)=PB(I,J,2)
        RHOB(K,M,2)=RHOB(I,J,2)
        URB(K,M,2)=URB(I,J,2)
```

```
      UZB(K,M,2)=UZB(I,J,2)
      AB(K,M,2)=AB(I,J,2)
      RBM(K,M,2)=RBM(I,J,2)
 1202 ZBM(K,M,2)=ZBM(I,J,2)
C     COMPUTE BOUNDARY POINT
      DO 1200 I=1,NBP
      KSPCE=LBPT(I)-1
      IBOUND=1
      IF((JCODE(I).EQ.7).OR.(JCODE(I).EQ.9))IBOUND=4
      DO 1200 K=2,KSPCE
 1200 CALL BCNPT(I,K,CONST,KAXIS,TIME,DELTA,KPRINT,IBOUND)
C     COMPUTE GENERAL POINTS
      DO 1300 I=1,NOZ
      DO 1300 J=1,NOR
 1300 CALL GENCOR(I,J,CONST,KAXIS,TIME,DELTA,KPRINT)
C     INTERPOLATE FOR ODD BOUNDARY POINTS
      CALL ODDPT(3,2)
C     INTERPOLATE FOR ODD MESH POINTS
      CALL CODPT(IPASS,2)
      IPASS=2
C     STORE PROPERTIES AND DIMENSIONALIZE
      DO 900 I=1,NOZ
      DO 900 K=1,NOR
      IF(NACTVE(I,K).EQ.0)GO TO 900
      P(I,K,1)=P(I,K,2)
      RHO(I,K,1)=RHO(I,K,2)
      UR(I,K,1)=UR(I,K,2)
      UZ(I,K,1)=UZ(I,K,2)
      A(I,K,1)=A(I,K,2)
  900 CONTINUE
      DO 902 I=1,NBP
      KSPCE=LBPT(I)
      DO 902 K=1,KSPCE
      NBACT(I,K,1)=NBACT(I,K,2)
      NBODD(I,K,1)=NBODD(I,K,2)
      THETA(I,K,1)=THETA(I,K,2)
      ITYPE(I,K,1)=ITYPE(I,K,2)
      INUM(I,K,1)=INUM(I,K,2)
      IF(NBACT(I,K,2).EQ.0) GO TO 902
      RBM(I,K,1)=RBM(I,K,2)
      ZBM(I,K,1)=ZBM(I,K,2)
      PB(I,K,1)=PB(I,K,2)
      RHOB(I,K,1)=RHOB(I,K,2)
      AB(I,K,1)=AB(I,K,2)
      URB(I,K,1)=URB(I,K,2)
      UZB(I,K,1)=UZB(I,K,2)
      THETA(I,K,1)=THETA(I,K,2)
  902 CONTINUE
      CALL ADJUST
      IF(MOD(L,MSKIP).GT.0)GO TO 790
      TIME1=TIME*WIDTH/AINIT
C     WRITE OUT RESULTS FOR TIME PLANE
      WRITE(6,8650)TIME1
      WRITE(6,8640)L
      WRITE(6,8610)
      WRITE(6,8620)
      WRITE(6,8220)
      DO 910 I=1,NOZ
```

```
      DO 910 K=1,NOR
      IF(NACTVE(I,K).EQ.0) GO TO 910
      WRITE(6,8630)I,K,P(I,K,2),RHO(I,K,2),A(I,K,2),UR(I,K,2),UZ(I,K,2)
  910 CONTINUE
      WRITE(6,8650)TIME1
      WRITE(6,8800)
      WRITE(6,8610)
      WRITE(6,8620)
      WRITE(6,8220)
      DO 912 I=1,NOZ
      DO 912 K=1,NOR
      IF(NACTVE(I,K).EQ.0) GO TO 912
      RMA=RM(I,K)*WIDTH
      ZMA=ZM(I,K)*WIDTH
      CALL EXACT(RMA,ZMA,TIME1)
      WRITE(6,8630)I,K,PRAT,RRAT,ARAT,URR,UZZ
      DO1J=1,5
    1 ERROR(J)=0.
      IF(PRAT.NE.0.)ERROR(1)=1.-P(I,K,2)/PRAT
      IF(RRAT.NE.0.)ERROR(2)=1.-RHO(I,K,2)/RRAT
      IF(ARAT.NE.0.)ERROR(3)=1.-A(I,K,2)/ARAT
      IF(URR.NE.0.)ERROR(4)=1.-UR(I,K,2)/URR
      IF(UZZ.NE.0.)ERROR(5)=1.-UZ(I,K,2)/UZZ
      WRITE(6,8632)(ERROR(J),J=1,5)
  912 CONTINUE
C     WRITE OUT BOUNDARY POINTS
      WRITE(6,8650)TIME1
      WRITE(6,8640)L
      WRITE(6,8900)
      WRITE(6,8902)
      WRITE(6,8220)
      DO 930 I=1,NBP
      KSPCE=LBPT(I)
      DO 930 K=1,KSPCE
      IF(NBACT(I,K,2).EQ.0) GO TO 930
      RMA=RBM(I,K,2)*WIDTH
      ZMA=ZBM(I,K,2)*WIDTH
      WRITE(6,8904)I,K,ZMA,RMA,PB(I,K,2),RHOB(I,K,2),AB(I,K,2),
     .URB(I,K,2),UZB(I,K,2)
  930 CONTINUE
      WRITE(6,8650)TIME1
      WRITE(6,8800)
      WRITE(6,8220)
      WRITE(6,8610)
      WRITE(6,8620)
      WRITE(6,8220)
      DO 932 I=1,NBP
      KSPCE=LBPT(I)
      DO 932 K=1,KSPCE
      IF(NBACT(I,K,2).EQ.0) GO TO 932
      RMA=RBM(I,K,2)*WIDTH
      ZMA=ZBM(I,K,2)*WIDTH
      CALL EXACT(RMA,ZMA,TIME1)
      WRITE(6,8630)I,K,PRAT,RRAT,ARAT,URR,UZZ
      DO2J=1,5
    2 ERROR(J)=0.
      IF(PRAT.NE.0.)ERROR(1)=1.-PB(I,K,2)/PRAT
      IF(RRAT.NE.0.)ERROR(2)=1.-RHOB(I,K,2)/RRAT
```

```
      IF(ARAT.NE.0.)ERROR(3)=1.-AB(I,K,2)/ARAT
      IF(URR.NE.0.)ERROR(4)=1.-URB(I,K,2)/URR
      IF(UZZ.NE.0.)ERROR(5)=1.-UZB(I,K,2)/UZZ
      WRITE(6,8632)(ERROR(J),J=1,5)
932   CONTINUE
790   CONTINUE
      GO TO 100
8000  FORMAT(2(I3,2X),6E10.0,/,E10.0,I1,4X,I2)
8001  FORMAT(A2)
8002  FORMAT(10(I2,4X))
8003  FORMAT(1HO,'KAXIS = ',I2,4X,'NCODE = ',I2,4X,'KPRINT = ',I2,/
     1' NBP = ',I2,4X,'NTIME = ',I3,4X,'NBT = ',I2,4X,'NPLOT = ',I2)
8020  FORMAT(7E10.0)
8021  FORMAT(5E10.0)
8022  FORMAT(6E10.0,I1)
8200  FORMAT(21X,'MODU 43, TWO-DIMENSIONAL UNSTEADY FLOW ANALYSIS'/)
8220  FORMAT(1H ,/)
8230  FORMAT(1HO,2(9HNUMBER OF,5X),12HR-INCREMENTS,3X,
     .12HZ-INCREMENTS,3X,9HINITIAL R,6X,11HMAXIMUM RAD,
     .4X,9HINITIAL Z,6X,7HFINAL Z)
8232  FORMAT(1H ,10HBOUND.PTS.,4X,11HTIME PLANES,3X,
     .6(6H FEET ,9X))
8240  FORMAT(1HO,20X,28HINITIAL REFERENCE CONDITIONS)
8242  FORMAT(1H ,8HPRESSURE,7X,7HDENSITY,8X,11HSOUND SPEED,4X,
     .10HREF.LENGTH,5X,10HTIME PLANE)
8244  FORMAT(1H ,8HLB/FT**2,7X,11HSLUGS/FT**3,4X,6HFT/SEC,9X,
     .4HFEET  ,9X,8HCONSTANT)
8245  FORMAT(1H ,8HNT/M**2 ,7X,11HKG/M**3    ,4X,6HM/SEC ,9X,
     .6HMETERS,9X,8HCONSTANT)
8250  FORMAT(1H1,20X,'BOUNDARY POINT COORDINATES')
8252  FORMAT(1H ,3(4HSECT,3X,6HRADIAL,9X,5HAXIAL,10X))
8254  FORMAT(1H ,7X,3(4HFEET  ,9X,4HFEET  ,16X))
8255  FORMAT(1H ,7X,3(6HMETERS,9X,6HMETERS,16X))
8256  FORMAT(1H ,15HBOUNDARY NUMBER,I3)
8258  FORMAT('1BOUNDARY POINT COORDINATES AND INITIAL PROPERTIES')
8260  FORMAT(1H1,10X,23HMESH POINT COORDINATES ,
     .22HAND INITIAL PROPERTIES)
8262  FORMAT(5HOMESH,3X,6HRADIAL,9X,5HAXIAL,10X,8HPRESSURE,
     .7X,7HDENSITY,8X,11HSOUND SPEED,4X,10HRADIAL VEL,5X,9HAXIAL VEL)
8264  FORMAT(' NO',5X,2(4HCOOR,11X),5(5HRATIO,10X))
8270  FORMAT(1H ,2(I4,10X),6(F9.6,6X))
8280  FORMAT(1H ,7(F9.4,6X))
8290  FORMAT(1H ,3(I3,2X,F9.4,6X,F9.4,6X))
8300  FORMAT(1H ,I2,1H,,I2,7(F14.9,1X))
8400  FORMAT(1H1,15X,19HBOUNDARY CONDITIONS)
8410  FORMAT(1H ,10HBOUND SEG ,5X,6HTIME =,F11.8,4H SEC)
8412  FORMAT(1H ,10HBOUND SEG ,5X,15HTIME IS INDEPNT)
8420  FORMAT(1X,I3,' TO',I3,6X,'DISTANCE RATIO',6X,2A4)
8422  FORMAT(37X,2A4)
8424  FORMAT(1H ,//)
8425  FORMAT(1H ,20X,F7.4,10X,F10.6)
8426  FORMAT(1H ,20X,7HINDEPNT,10X,F10.6)
8428  FORMAT(' BOUNDARY CONDITION IS A FUNCTION AT BOUNDARY',I3)
8600  FORMAT(1H ,10E10.3)
8610  FORMAT(1HO,10HMESH POINT,10X,8HPRESSURE,7X,7HDENSITY,8X,
     .11HSOUND SPEED,4X,9HRAD. VEL.,6X,10HAXIAL VEL.)
8620  FORMAT(1H ,20X,5(5HRATIO,10X))
8630  FORMAT(1H ,4X,I3,1H,,I3,4X,5(1X,F14.9))
```

```
8632 FORMAT(1H ,15HPER CENT ERROR ,5(1X,F14.9))
8640 FORMAT(1H ,12HTIME PLANE =,I4)
8650 FORMAT(1H1,7HTIME = ,F12.9,4H SEC)
8800 FORMAT(1H ,15X,14HEXACT SOLUTION)
8900 FORMAT(1HO,7X,12HBOUND.RADIUS,3X,11HBOUND.AXIAL,4X,8HPRESSURE,7X,
    .7HDENSITY,8X,11HSCUND SPEED,4X,9HRAD. VEL.,6X,
    .10HAXIAL VEL.)
8902 FORMAT(1H ,38X,5(5HRATIC,10X))
8904 FORMAT(1H ,I2,1H,,I2,7(F14.9,1X))
     END
```

```
      SUBROUTINE ABDIF(IB,ETA,KVALUE)
C     THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE
C     DERIVATIVES FOR THE ABNORMAL GENERAL POINT
      COMMON/BUFFER/DURR(15),DURZ(15),DUZR(15),DUZZ(15),DAR(15),DAZ(15)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),URD(15),
     .UZD(15),LSUB(15)
      DIMENSION DIST(15),PURR(15),PUZR(15),PAR(15)
C     VERTICAL INTERSECTION
      GO TO(100,300),IB
  100 GO TO(110,130,110,130),KVALUE
  110 CONTINUE
C     COMPUTE R PRIME DERIVATIVE FOR 1,4,7
      DIST(7)=0.
      DIST(4)=(RPD(4)-RPD(7))*COS(ETA)+(ZPD(4)-ZPD(7))*SIN(ETA)
      DIST(1)=(RPD(1)-RPD(7))*COS(ETA)+(ZPD(1)-ZPD(7))*SIN(ETA)
      CALL FINITE(1,4,7,URD,DIST,PURR(1),PURR(4),PURR(7))
      CALL FINITE(1,4,7,UZD,DIST,PUZR(1),PUZR(4),PUZR(7))
      CALL FINITE(1,4,7,AD,DIST,PAR(1),PAR(4),PAR(7))
C     COMPUTE R DERIVATIVES
      DO 112 K=2,3
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL LAGRNE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  112 CALL LAGRNE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE AXIAL DERIVATIVES
      DO 114 K=1,7,3
      K1=K+1
      K2=K+2
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  114 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE R DERIVATIVES AT 1,4,7
      DO 116 K=1,7,3
      DURR(K)=(PURR(K)-DURZ(K)*SIN(ETA))/COS(ETA)
      DUZR(K)=(PUZR(K)-DUZZ(K)*SIN(ETA))/COS(ETA)
  116 DAR(K)=(PAR(K)-DAZ(K)*SIN(ETA))/COS(ETA)
      RETURN
C     COMPUTE R PRIME DERIVATIVE FOR 3,6,9
  130 DIST(3)=0.
      DIST(6)=(RPD(6)-RPD(3))*COS(ETA)+(ZPD(6)-ZPD(3))*SIN(ETA)
      DIST(9)=(RPD(9)-RPD(3))*COS(ETA)+(ZPD(9)-ZPD(3))*SIN(ETA)
      CALL FINITE(3,6,9,URD,DIST,PURR(3),PURR(6),PURR(9))
      CALL FINITE(3,6,9,UZD,DIST,PUZR(3),PUZR(6),PUZR(9))
      CALL FINITE(3,6,9,AD,DIST,PAR(3),PAR(6),PAR(9))
C     COMPUTE R DERIVATIVES
      DO 132 K=1,2
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  132 CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE AXIAL DERIVATIVES
      DO 134 K=1,7,3
      K1=K+1
      K2=K+2
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
```

```
  134 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE R DERIVATIVES AT 3,6,9
      DO 136 K=3,9,3
      DURR(K)=(PURR(K)-DURZ(K)*SIN(ETA))/COS(ETA)
      DUZR(K)=(PUZR(K)-DUZZ(K)*SIN(ETA))/COS(ETA)
  136 DAZ(K)=(PAR(K)-DAZ(K)*SIN(ETA))/COS(ETA)
      RETURN
C     HORIZONTAL INTERSECTION
  300 GO TO(310,310,330,330),KVALUE
C     COMPUTE R PRIME DERIVATIVE 1,2,3
  310 DIST(1)=0.
      DIST(2)=(RPD(2)-RPD(1))*COS(ETA)+(ZPD(2)-ZPD(1))*SIN(ETA)
      DIST(3)=(RPD(3)-RPD(1))*COS(ETA)+(ZPD(3)-ZPD(1))*SIN(ETA)
      CALL FINITE(1,2,3,URD,DIST,PURR(1),PURR(2),PURR(3))
      CALL FINITE(1,2,3,UZD,DIST,PUZR(1),PUZR(2),PUZR(3))
      CALL FINITE(1,2,3,AD,DIST,PAR(1),PAR(2),PAR(3))
C     COMPUTE R DERIVATIVES WITH LAGRANGE
      DO 312 K=1,3
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL LAGRNE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  312 CALL LAGRNE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE AXIAL DERIVATIVES
      DO 314 K=4,7,3
      K1=K+1
      K2=K+2
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  314 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE AXIAL DERIVATIVES AT 1,2,3
      DO 316 K=1,3
      DURZ(K)=(PURR(K)-DURR(K)*COS(ETA))/SIN(ETA)
      DUZZ(K)=(PUZR(K)-DUZR(K)*COS(ETA))/SIN(ETA)
  316 DAZ(K)=(PAR(K)-DAR(K)*COS(ETA))/SIN(ETA)
      RETURN
C     COMPUTE R PRIME DERIVATIVE AT 7,8,9
  330 DIST(9)=0.
      DIST(8)=(RPD(8)-RPD(9))*COS(ETA)+(ZPD(8)-ZPD(9))*SIN(ETA)
      DIST(7)=(RPD(7)-RPD(9))*COS(ETA)+(ZPD(7)-ZPD(9))*SIN(ETA)
      CALL FINITE(7,8,9,URD,DIST,PURR(7),PURR(8),PURR(9))
      CALL FINITE(7,8,9,UZD,DIST,PUZR(7),PUZR(8),PUZR(9))
      CALL FINITE(7,8,9,AD,DIST,PAR(7),PAR(8),PAR(9))
C     COMPUTE R DERIVATIVES WITH LAGRANGE
      DO 332 K=1,3
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL LAGRNE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  332 CALL LAGRNE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE AXIAL DERIVATIVES
      DO 334 K=1,4,3
      K1=K+1
      K2=K+2
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  334 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE AXIAL DERIVATIVES AT 7,8,9
```

```
      DO 336 K=7,9
      DURZ(K)=(PURR(K)-DURR(K)*COS(ETA))/SIN(ETA)
      DUZZ(K)=(PUZR(K)-DUZR(K)*COS(ETA))/SIN(ETA)
  336 DAZ(K)=(PAR(K)-DAR(K)*COS(ETA))/SIN(ETA)
      RETURN
      END
```

```
      SUBROUTINE ABGEN(RNEW,ZNEW,IB,KVALUE,M,ETA)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/GENPT/IJ,IK,I1,I2,J1,J2,NGEN(9),XY(3)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),URD(15),
     .UZD(15),LSUB(15)
      COMMON/XCHNG/ISTRE,KSTRE,IL,IM,ID,ID1,ID2
C     THE PURPOSE OF THIS SUBROUTINE IS TO ADJUST THE
C     GENERAL POINT DOMAIN POINTS FOR AN ABNORMAL GENERAL
C     POINT
C     SET ROW AND COLUMN INDICES OF MISSING POINT
      GO TO(100,110,120,130),KVALUE
  100 IROW=I1
      ICOL=J2
      GO TO 140
  110 IROW=I2
      ICOL=J2
      GO TO 140
  120 IROW=I1
      ICOL=J1
      GO TO 140
  130 IROW=I2
      ICOL=J1
C     DETERMINE NEAREST BOUNDARY POINT
  140 DO 200 I=1,NBP
      KSPCE=LBPT(I)
      ISTRE=I
C     ASSUME NEAREST BOUNDARY POINT IS A VERTICAL INTERSECTION
      KNUM=ICOL
      KOUNT=1
  208 CONTINUE
      DO 210 K=1,KSPCE
      KSTRE=K
      KCMPR=INUM(I,K,M)
      IF(KNUM.NE.KCMPR)GO TO 210
C     CHECK INTERSECTION TYPE,1=VERTICAL,2=HORIZONTAL
      KTYPE=ITYPE(I,K,M)
      IF(KTYPE.NE.KOUNT)GO TO 210
C     CHECK POSITION OF BOUNDARY POINT
      GO TO(300,400),KOUNT
C     VERTICAL INTERSECTION
  300 GO TO(310,320,330,340),KVALUE
  310 IF(ZPD(2).GT.ZBM(I,K,M))GO TO 220
      GO TO 210
  320 IF(ZPD(2).LT.ZBM(I,K,M))GO TO 220
      GO TO 210
  330 IF(ZPD(8).GT.ZBM(I,K,M))GO TO 220
      GO TO 210
  340 IF(ZPD(8).LT.ZBM(I,K,M))GO TO 220
      GO TO 210
C     HORIZONTAL INTERSECTION
  400 GO TO(410,420,430,440),KVALUE
  410 IF(RPD(4).LT.RBM(I,K,M))GO TO 220
```

```
      GO TO 210
  420 IF(RPD(6).LT.RBM(I,K,M))GO TO 220
      GO TO 210
  430 IF(RPD(4).GT.RBM(I,K,M))GO TO 220
      GO TO 210
  440 IF(RPD(6).GT.RBM(I,K,M))GO TO 220
  210 CONTINUE
      GO TO(230,200),KOUNT
C     ASSUME HORIZONTAL INTERSECTION
  230 KOUNT=2
      KNUM=IROW
      GO TO 208
  200 CONTINUE
      WRITE(6,8000)IJ,IK
      CALL EXIT
C     CHANGE BOUNDARY POINTS INTO DOMAIN POINTS
  220 FTA=THETA(ISTRE,KSTRE,M)
      GO TO(600,700),KOUNT
C     VERTICAL INTERSECTION FIRST
  600 IB=1
      LSUB(1)=7
      LSUB(2)=4
      LSUB(3)=1
      LSUB(4)=8
      LSUB(5)=5
      LSUB(6)=2
      LSUB(7)=9
      LSUB(8)=6
      LSUB(9)=3
      GO TO(610,620,630,640),KVALUE
C     POINT 1 DOMAIN REPLACEMENT
  610 IL=KSTRE-1
      IM=KSTRE-2
      ID=1
      ID1=4
      ID2=7
      CALL REPLCT(M)
      RETURN
C     POINT 3 DOMAIN REPLACEMENT
  620 IL=KSTRE+1
      IM=KSTRE+2
      ID=3
      ID1=6
      ID2=9
      CALL REPLCT(M)
      RETURN
C     POINT 7 DOMAIN REPLACEMENT
  630 IL=KSTRE+1
      IM=KSTRE+2
      ID=7
      ID1=4
      ID2=1
      CALL REPLCT(M)
      RETURN
C     POINT 9 DOMAIN REPLACEMENT
  640 IL=KSTRE-1
      IM=KSTRE-2
      ID=9
```

```
      ID1=6
      ID2=3
      CALL REPLCT(M)
      RETURN
C     HORIZONTAL INTERSECTION
  700 IB=2
      LSUB(1)=7
      LSUB(2)=8
      LSUB(3)=9
      LSUB(4)=4
      LSUB(5)=5
      LSUB(6)=6
      LSUB(7)=1
      LSUB(8)=2
      LSUB(9)=3
      GO TO(710,720,730,740),KVALUE
C     POINT 1 DOMAIN REPLACEMENT
  710 IL=KSTRE+1
      IM=KSTRE+2
      ID=1
      ID1=2
      ID2=3
      CALL REPLCT(M)
      RETURN
C     POINT 3 DOMAIN REPLACEMENT
  720 IL=KSTRE-1
      IM=KSTRE-2
      ID=3
      ID1=2
      ID2=1
      CALL REPLCT(M)
      RETURN
C     POINT 7 DOMAIN REPLACEMENT
  730 IL=KSTRE-1
      IM=KSTRE-2
      ID=7
      ID1=8
      ID2=9
      CALL REPLCT(M)
      RETURN
C     POINT 9 DOMAIN REPLACEMENT
  740 IL=KSTRE+1
      IM=KSTRE+2
      ID=9
      ID1=8
      ID2=7
      RETURN
 8000 FORMAT(1H ,34HERROR IN SUBROUTINE ABGEN AT POINT,
     .I3,1H,,I2)
      END
```

```
      SUBROUTINE ACTVTY(KVALUE,NOACT)
      COMMON/GENPT/IJ,IK,I1,I2,J1,J2,NGEN(9),XY(3)
C     THE PURPOSE OF THIS SUBROUTINE IS TO CHECK DOMAIN
C     POINT ACTIVITY AND SPECIFY WHICH POINT IS INACTIVE
C     AND THE TYPE OF INTERSECTION
      NOACT=0
C     DETERMINE NUMBER OF INACTIVE POINTS
      DO 100 I=1,9
      IF(NGEN(I).EQ.0)NOACT=NOACT+1
  100 CONTINUE
C     SET KVALUE,ASSUME IT IS POINT 1
      KVALUE=1
      IF(NGEN(3).EQ.0)KVALUE=2
      IF(NGEN(7).EQ.0)KVALUE=3
      IF(NGEN(9).EQ.0)KVALUE=4
      RETURN
      END
```

```
      SUBROUTINE ADJUST
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     1LBPT(10),IAXES(10,30)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
C     THIS SUBROUTINE ADJUSTS BOUNDARY SPACING
      DO 202 I=1,NBP
      KSPCE=LCODE(I)
      TOTAL=SQRT(((RB(I,1,1)-RB(I,KSPCE,1))**2)+
     .((ZB(I,1,1)-ZB(I,KSPCE,1))**2))
      XI=ATAN2((ZB(I,KSPCE,1)-ZB(I,1,1)),(RB(I,KSPCE,1)-RB(I,1,1)))
      BSPCE(I,1)=0.
      DO 204 J=2,KSPCE
      ETA=ATAN2((ZB(I,J,1)-ZB(I,1,1)),(RB(I,J,1)-RB(I,1,1)))
      BSPCE(I,J)=TOTAL*COS(ETA-XI)
  204 CONTINUE
  202 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BCOMP(IBOUND,TIME,DELTA,KAXIS,CONST,
     .NCOUNT,KPRINT,I,J)
C     THIS ROUTINE INTEGRATES THE BOUNDARY POINT COMPATIBILITY
C     EQUATIONS
      REAL KAXIS
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/STATE/PINIT,RHOINT,AINIT
      COMMON/EXT/URR,UZZ,Y,RRAT,ARAT
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5),PAR(5),PAZ(5)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/SETUP/RAS(4),RAC(4),R(5),DELP,DELRH,DELUR,DELUZ,S(5)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      DIMENSION DIST(3),F(3)
      DATA KK/1/
      PI=3.141592653589793
      TOL=0.0000001
      TOL=0.0001
      CALL BEGIN(IBOUND,DELTA,KAXIS,CONST)
      ROR=RNEW
      ZOR=ZNEW
      IF(JCODE(I).EQ.3)KK=1
      GO TO(100,110),KK
  100 KK=2
      IF(MCODE(I).EQ.0) GO TO 200
      RORA=ROR*WIDTH
      ZORA=ZOR*WIDTH
      TIME1=TIME*WIDTH/AINIT
      CALL EXACT(RORA,ZORA,TIME1)
      IF((JCODE(I).NE.2).AND.(JCODE(I).NE.7))GO TO 110
      X=-URR*SIN(ANGLE)+UZZ*COS(ANGLE)
      IF(JCODE(I).NE.7)GO TO 110
      W=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      U=RRAT
      ROR1=RBM(I,J-1,1)*WIDTH
      ZOR1=ZBM(I,J-1,1)*WIDTH
      ROR2=RBM(I,J+1,1)*WIDTH
      ZOR2=ZBM(I,J+1,1)*WIDTH
      DIST(1)=0.
      DIST(2)=(RBM(I,J,1)-RBM(I,J-1,1))*COS(ANGLE)+
     .(ZBM(I,J,1)-ZBM(I,J-1,1))*SIN(ANGLE)
      DIST(3)=(RBM(I,J+1,1)-RBM(I,J-1,1))*COS(ANGLE)+
     .(ZBM(I,J+1,1)-ZBM(I,J-1,1))*SIN(ANGLE)
      F(2)=W
      CALL EXACT(ROR1,ZOR1,TIME1)
      F(1)=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      CALL EXACT(ROR2,ZOR2,TIME1)
      F(3)=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      CALL FINITE(1,2,3,F,DIST,ART,DWR,ART)
      GO TO 110
```

```
 200  CONTINUE
      KV=I
      NK=JCODE(KV)
      NTIME=KCODE(KV,1)
      KSPCE=LCODE(KV)
      KL=J
      CALL FIND(KV,KL,NK,NTIME,KSPCE,ROR,ZOR,Y,W,DZZ,
     .DWR,ART,TIME)
      IF((NK.EQ.2).OR.(NK.EQ.7))X=Y
 110  CONTINUE
C     INTEGRATE COMPATIBILITY EQUATIONS
      NK=JCODE(I)
      GO TO(300,310,300,320,310,330,340),NK
 300  XA=SOUND(RHNEW,Y)
      DELUR1=-.5*RHNEW*XA*(2.*URG(5)-URG(1)-URG(3))-.5*
     .(RAS(1)-RAS(3))*DELUZ+R(3)-R(1)
      DELUR1=DELUR1/(.5*(2.*RHNEW*XA+RAC(1)-RAC(3)))
      URNEW1=URG(5)+DELUR1
      IF(ROR.EQ.0.) GO TO 302
      PART=URNEW/ROR
      GO TO 304
 302  PART=PURR(5)
 304  CONTINUE
      DELUZ1=-2.*(Y-PG(5))*CONST-.5*(RAC(1)+RAC(3)+
     .2.*RAC(4))*DELUR1-RHNEW*XA*XA*KAXIS*PART*
     .DELTA-.5*XA*RHNEW*(URG(3)-URG(1)+2.*(UZG(4)-
     .UZG(5)))+2.*R(5)-(R(1)+R(3)+2.*R(4))
      DELUZ1=DELUZ1/(.5*(-2.*RHNEW*XA+RAS(1)+RAS(3)+
     .2.*RAS(4)))
      DELRH1=(Y-PG(5))*CONST-R(5)*(XA*XA/(AG(5)*AG(5))-1.)
      DELRH1=DELRH1/(XA*XA)
      RHNEW1=DELRH1+RHCG(5)
      UZNEW1=UZG(5)+DELUZ1
      IF(ITEST.NE.1) GO TO 400
      ITEST=2
      GO TO 410
 400  CONTINUE
      RATIO=.01
      IF(ABS(URNEW).GT.0.)RATIO=ABS((URNEW1-URNEW)/URNEW)
      IF(RATIO.GT.TOL)GOTO410
      IF(ABS(UZNEW).GT.0.)RATIO=ABS((UZNEW1-UZNEW)/UZNEW)
      IF(RATIO.GT.TOL)GOTO410
      IF(ABS((RHNEW1-RHNEW)/RHNEW).GT.0.0000001)GO TO 410
      NCOUNT=2
      KK=1
 410  URNEW=URNEW1
      UZNEW=UZNEW1
      RHNEW=RHNEW1
      PNEW=Y
      ANEW=SOUND(RHNEW,PNEW)
      IF(KPRINT.GT.0) CALL PRINT
      RETURN
 310  XA=SOUND(RHNEW,PNEW)
      IF(ROR.EQ.0.) GO TO 312
      PART=URNEW/ROR
      GO TO 314
 312  PART=PURR(5)
 314  CONTINUE
```

```
      DELP1=-.25*(RAC(1)+RAC(3)+2.*RAC(4))*DELUR-.5*RHNEW*
     .XA*XA*KAXIS*PART*DELTA-.25*RHNEW*XA*(URG(3)-
     .URG(1)+2.*(UZG(4)-UZG(5)))-.25*(X-UZG(5))*(-2.*
     .RHNEW*XA+RAS(1)+RAS(3)+2.*RAS(4))+R(5)-.5*(R(1)+
     .R(3)+2.*R(4))
      DELP1=DELP1/CONST
      PNEW1=PG(5)+DELP1
      XA=SOUND(RHNEW,PNEW1)
      DELUR1=-.5*RHNEW*XA*(2.*URG(5)-URG(1)-URG(3))-
     ..5*(X-UZG(5))*(RAS(1)-RAS(3))+R(3)-R(1)
      DELUR1=DELUR1/(.5*(2.*RHNEW*XA+RAC(1)-RAC(3)))
      URNEW1=URG(5)+DELUR1
      PART=URNEW1/ROR
      IF(ROR.EQ.0.)PART=PURR(5)
      DELRH1=4.*DELP1*CONST+.5*DELUR1*(RAC(1)+RAC(3)+
     .2.*PAC(4))+RHNEW*XA*XA*KAXIS*PART*DELTA+.5*
     .RHNEW*XA*(URG(3)-URG(1))+RHNEW*XA*(UZG(4)-UZG(5))+
     ..5*(X-UZG(5))*(-2.*RHNEW*XA+RAS(1)+RAS(3)+
     .2.*RAS(4))-
     .2.*((XA/AG(5))**2)*R(5)+(R(1)+R(3)+2.*R(4))
      DELRH1=DELRH1/(2.*XA*XA)
      RHNEW1=DELRH1+RHOG(5)
      IF(ITEST.NE.1) GO TO 500
      ITEST=2
      GO TO 510
  500 CONTINUE
      RATIO=.01
      IF(ABS(URNEW).GT.0.)RATIO=ABS((URNEW1-URNEW)/URNEW)
      IF(RATIO.GT.TOL)GOTO510
      IF(ABS((PNEW1-PNEW)/PNEW).GT.TOL)GOTO510
      IF(ABS((RHNEW1-RHNEW)/RHNEW).GT.TOL)GOTO510
      NCOUNT=2
      KK=1
  510 URNEW=URNEW1
      UZNEW=X
      PNEW=PNEW1
      RHNEW=RHNEW1
      ANEW=SOUND(RHNEW,PNEW)
      IF(KPRINT.GT.0)CALL PRINT
      RETURN
  320 CONTINUE
  330 CONTINUE
  340 XA=SOUND(RHNEW,PNEW)
      IF(ROR.EQ.0.)GO TO 342
      PART=W/RCR
      GO TO 344
  342 PART=PURR(5)
  344 CONTINUE
      DELP1=-.5*(RHNEW*XA*XA*(KAXIS*PART+DWR))*
     .DELTA-R(4)-.5*RAC(4)*(W-URG(5))+
     ..5*RHNEW*XA*(UZG(5)-UZG(4))-
     ..5*(-RHNEW*XA+RAS(4))*(X-UZG(5))
      DELP1=DELP1/CONST
      PNEW1=PG(5)+DELP1
      ANEW1=SOUND(RHNEW,PNEW1)
      DELRH1=(2.*DELP1*CONST+.25*(RAC(1)+RAC(3)+2.*RAC(4))*(W-URG(5))+
     ..25*RHNEW*ANEW1*(URG(3)-URG(1)+2.*(UZG(4)-UZG(5)))+
     ..25*(-2.*RHNEW*ANEW1+RAS(1)+RAS(3)+2.*RAS(4))*(X-UZG(5))+
```

```
     ..5*RHNEW*ANEW1*ANEW1*KAXIS*PART*DELTA+.5*(R(1)+R(3))+
     .(R(4)-ANEW1*ANEW1*R(5)/(AG(5)*AG(5))))/
     .(ANEW1*ANEW1)
      RHNEW1=RHOG(5)+DELRH1
      ANEW1=SOUND(RHNEW1,PNEW1)
      IF(ITEST.NE.1)GO TO 600
      ITEST=2
      GO TO 610
  600 IF(ABS((PNEW1-PNEW)/PNEW).GT.TOL)GOTO610
      IF(ABS((RHNEW1-RHNEW)/RHNEW).GT.TOL)GOTO610
      NCOUNT=2
      KK=1
  610 PNEW=PNEW1
      RHNEW=RHNEW1
      URNEW=W
      UZNEW=X
      ANEW=ANEW1
      IF(KPRINT.GT.0)CALL PRINT
      RETURN
      END
```

```
      SUBROUTINE BCPROP(I1,I2,I3,I4,I,M,PROP,X,PROPB,XIND)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NCZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      DIMENSION PROP(1),X(1),PROPB(10,30,2)
C     IND=1,VERTICAL INTERSECTION,IND=2,HORIZONTAL INTERSECTION
      IND=ITYPE(I,I4,M)
C     VERTICAL INTERSECTION
      GO TO(100,300),IND
  100 CONTINUE
      PROP(1)=PROPB(I,I1,M)
      X(1)=RBM(I,I1,M)
      PROP(2)=PROPB(I,I2,M)
      X(2)=RBM(I,I2,M)
      PROP(3)=PROPB(I,I3,M)
      X(3)=RBM(I,I3,M)
      XIND=RBM(I,I4,M)
      RETURN
  300 CONTINUE
      PROP(1)=PROPB(I,I1,M)
      X(1)=ZBM(I,I1,M)
      PROP(2)=PROPB(I,I2,M)
      X(2)=ZBM(I,I2,M)
      PROP(3)=PROPB(I,I3,M)
      X(3)=ZBM(I,I3,M)
      XIND=ZBM(I,I4,M)
      RETURN
      END
```

```
      SUBROUTINE BEGIN(IBOUND,DELTA,KAXIS,CONST)
      REALKAXIS
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5),PAR(5),PAZ(5)
      COMMON/SETUP/RAS(4),RAC(4),R(5),DELP,DELRH,DELUR,DELUZ,S(5)
C     THIS SUBROUTINE DETERMINES R(I)
      PI=3.1415926535898
      ROR=RNEW
      ZOR=ZNEW
      DELP=PNEW-PG(5)
      DELRH=RHNEW-RHOG(5)
      DELUR=URNEW-URG(5)
      DELUZ=UZNEW-UZG(5)
C     SET RHO*A*SIN EQUAL TO RAS
C     SET RHO*A*COS EQUAL TO RAC
      DO 100 I=1,4
      IF(IBOUND.EQ.0) GO TO 102
      IF((IBOUND.GT.0).AND.(I.EQ.2))GO TO 100
      IF(((IBOUND.EQ.2).OR.(IBOUND.EQ.5)).AND.(I.EQ.3))GO TO 100
      IF(((IBOUND.EQ.3).OR.(IBOUND.EQ.6)).AND.(I.EQ.1))GO TO 100
  102 AK=(FLOAT(I)-1.)/2.
      D=-PAR(I)*SIN(AK*PI)+PAZ(I)*COS(AK*PI)+
     .PURZ(I)*(COS(AK*PI)**2)-PUZR(I)*(SIN(AK*PI)**2)
      THETA=AK*PI+DELTA*C
      RAS(I)=AG(I)*RHOG(I)*SIN(THETA)
      RAC(I)=AG(I)*RHOG(I)*COS(THETA)
      IF(RG(I).NE.0.)GOTO50
      PART=PURR(I)
      GOTO60
   50 PART=URG(I)/RG(I)
   60 CONTINUE
      S(I)=KAXIS*PART+PURR(I)*(SIN(THETA)**2)-
     .(PURZ(I)+PUZR(I))*SIN(THETA)*COS(THETA)+
     .PUZZ(I)*(COS(THETA)**2)
      R(I)=CONST*(PG(5)-PG(I))+.5*(URG(5)-URG(I))*RAC(I)+
     ..5*(UZG(5)-UZG(I))*RAS(I)+.5*RHOG(I)*AG(I)*AG(I)*S(I)*DELTA
  100 CONTINUE
      IF(RG(5).NE.0.)GOTO70
      PART=PURR(5)
      GOTO80
   70 PART=URG(5)/RG(5)
   80 CONTINUE
      R(5)=.5*RHOG(5)*AG(5)*AG(5)*(KAXIS*PART+
     .PURR(5)+PUZZ(5))*DELTA
      RETURN
      END
```

```
      SUBROUTINE BCNPT(I,J,CONST,KAXIS,TIME,DELTA,KPRINT,IBOUND)
      REAL KAXIS
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RE(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),
     .AG(5),PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
C     THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE PROPERTIES
C     AT A BOUNDARY POINT
      INDEX=0
      ITEST=1
      IF((NBACT(I,J,2).EQ.0).OR.(NBODD(I,J,2).EQ.1)) RETURN
      ANGLE=THETA(I,J,1)
      RNEW=RBM(I,J,1)
      ZNEW=ZBM(I,J,1)
      IB=ITYPE(I,J,1)
C     SET CHARACTERISTIC LIMIT
C     SET BOUNDARY CONDITION
      CALL BOUNDY(I,J)
      IF(KPRINT.EQ.1)CALLPRINT
C     SET UP DERIVATIVES
      CALL DIFBD(I,J)
C     GUESS PROPERTIES
      CALL GUESS(1,I,J)
 1000 CALL INTEG(I,J,DELTA)
      CALL PHYCHR(IBOUND,DELTA)
C     INTERPOLATE FOR PROPERTIES
      CALL INPROP(IB,IBOUND)
      CALL CONVRT(IBOUND)
      NCOUNT=1
      CALL BCOMP(IBOUND ,TIME,DELTA,KAXIS,CONST,NCOUNT,KPRINT,I,J)
      INDEX=INDEX+1
      IF(INDEX.LT.25) GO TO 1010
      WRITE(6,8700)I,J
      CALL PRINT
      CALL EXIT
 1010 IF(NCOUNT.EQ.1) GO TO 1000
      ITEST=1
C     STORE PROPERTIES IN NEW TIME PLANE
      PB(I,J,2)=PNEW
      RHOB(I,J,2)=RHNEW
      URB(I,J,2)=URNEW*COS(ANGLE)-UZNEW*SIN(ANGLE)
      UZB(I,J,2)=URNEW*SIN(ANGLE)+UZNEW*COS(ANGLE)
      AB(I,J,2)=SOUND(RHNEW,PNEW)
      RBM(I,J,2)=RNEW
      ZBM(I,J,2)=ZNEW
      RETURN
 8700 FORMAT(1H ,33HPROGRAM LOOPING AT BOUNDARY POINT,
     .I2,1H,,I2)
      END
```

```
      SUBROUTINE BOUNDY(I,J)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),
     .AG(5),PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,
     .RNEW,ZNEW
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/XCHNG/IS,KS,KS1,KS2,ID,ID1,ID2
      COMMON/EXCHNG/IE,IF,IG,IC1,IC2,NUMBR
      PI=3.1415926535898
C     THIS SUBROUTINE SETS UP DOMAIN POINTS FOR BOUNDARY
C     INTERPOLATION
C     DETERMINE TYPE OF INTERSECTION
C     1=VERTICAL,2=HORIZONTAL
      IART=ITYPE(I,J,1)
      GO TO(100,300),IART
C     VERTICAL INTERSECTION
  100 CONTINUE
      NUMBR=INUM(I,J,1)
C     ASSUME THAT ROWS WILL INCREASE
      IROW=1
      IF((ANGLE.GT..75*PI).AND.(ANGLE.LE.1.25*PI))IROW=2
      GO TO(110,120),IROW
  110 DO 112 K=1,NOZ
      JK=K
      IF(ZM(K,NUMBR).GT.ZBM(I,J,1))GO TO 114
  112 CONTINUE
      WRITE(6,8000)I,J
      CALL EXIT
  114 LK=JK+1
      MK=JK+2
  116 CONTINUE
C     SET UP BOUNDARY DOMAIN POINTS
      ID=1
      ID1=2
      ID2=3
      IS=I
      KS=J-1
      KS1=J
      KS2=J+1
      CALL REPLCT(1)
C     DETERMINE IF THE ROW JK CUTS THE POSITIVE R
C     PRIME AXIS,IAXIS=1,POSITIVE,IAXIS=2,NEGATIVE,
C     IAXIS=3,POSITIVE NO CUT,IAXIS=4,NEGATIVE NO CUT
      IAXIS=2
      IF(((ANGLE.GT.0.).AND.(ANGLE.LE..25*PI)).OR.
     .((ANGLE.GT.PI).AND.(ANGLE.LE.1.25*PI)))IAXIS=1
      ZA=ZM(JK,NUMBR)
      IAXES(I,J)=IAXIS
      GO TO(130,132),IAXIS
  130 CONTINUE
      CALL SET(ANGLE,NUMBR,IART,IAXIS,IC1,IC2)
```

```
C       DETERMINE IF FIRST ROW CUTS POSITIVE R-PRIME AXIS
        VALUE1=AMAX1(ZBM(I,J,1),ZBM(I,J+1,1),ZA)
        VALUE2=AMIN1(ZBM(I,J,1),ZBM(I,J+1,1),ZA)
        IF((ZA.EQ.VALUE1).OR.(ZA.EQ.VALUE2))GO TO 134
C       SET UP DOMAIN POINTS ON ROW JK
        RPD(4)=RM(JK,IC1)
        ZPD(4)=ZM(JK,IC1)
        PD(4)=P(JK,IC1,1)
        RHOD(4)=RHO(JK,IC1,1)
        URD(4)=UR(JK,IC1,1)
        UZD(4)=UZ(JK,IC1,1)
        AD(4)=A(JK,IC1,1)
        RPD(5)=RM(JK,NUMBR)
        ZPD(5)=ZM(JK,NUMBR)
        PD(5)=P(JK,NUMBR,1)
        RHOD(5)=RHO(JK,NUMBR,1)
        URD(5)=UR(JK,NUMBR,1)
        UZD(5)=UZ(JK,NUMBR,1)
        AD(5)=A(JK,NUMBR,1)
C       INTERPOLATE FOR BOUNDARY POINT PROPERTIES
        ZPD(6)=ZM(JK,NUMBR)
        RPD(6)=RBM(I,J-1,1)+(RBM(I,J+1,1)-RBM(I,J+1,1))*
       .(ZPD(6)-ZBM(I,J-1,1))/(ZBM(I,J+1,1)-ZBM(I,J-1,1))
        CALL INTERP(PD,RPD,PD(6),RPD(6),3,3)
        CALL INTERP(RHOD,RPD,RHOD(6),RPD(6),3,3)
        CALL INTERP(URD,RPD,URD(6),RPD(6),3,3)
        CALL INTERP(UZD,RPD,UZD(6),RPD(6),3,3)
        CALL INTERP(AD,RPD,AD(6),RPD(6),3,3)
        LSUB(1)=1
        LSUB(2)=2
        LSUB(3)=3
        LSUB(4)=7
        LSUB(5)=8
        LSUB(6)=9
        LSUB(7)=10
        LSUB(8)=11
        LSUB(9)=12
        GO TO 136
  134   CONTINUE
        IAXES(I,J)=3
        DO201K=1,9
  201   LSUB(K)=K
C       INCREMENT COLUMN +,-,ONE
        IE=4
        IE=5
        IG=6
        CALL TRNSFR(IART,1,JK)
  136   CONTINUE
C       SET UP DOMAIN POINTS ON ROW LK
        IE=7
        IE=8
        IG=9
        CALL TRNSFR(IART,1,LK)
C       SET UP DOMAIN POINTS ON ROW MK
        IE=10
        IE=11
        IG=12
        CALL TRNSFR(IART,1,MK)
```

```
      RETURN
  132 CONTINUE
C     DETERMINE IF FIRST ROW CUTS NEGATIVE R-AXIS
      CALL SET(ANGLF,NUMBR,IART,IAXIS,IC1,IC2)
      VALUE1=AMAX1(ZBM(I,J,1),ZBM(I,J-1,1),ZA)
      VALUE2=AMIN1(ZBM(I,J,1),ZBM(I,J-1,1),ZA)
      IF((ZA.EQ.VALUE1).OR.(ZA.EQ.VALUE2))GO TO 140
C     SET UP DOMAIN POINTS ON ROW JK
      ZPD(4)=ZM(JK,NUMBR)
      RPD(4)=RBM(I,J-1,1)+(RBM(I,J+1,1)-RBM(I,J-1,1))*
     .(ZPD(4)-ZBM(I,J-1,1))/(ZBM(I,J+1,1)-ZBM(I,J-1,1))
      CALL INTERP(PD,RPD,PD(4),RPD(4),3,3)
      CALL INTERP(RHOD,RPD,RHOD(4),RPD(4),3,3)
      CALL INTERP(URD,RPD,URD(4),RPD(4),3,3)
      CALL INTERP(UZD,RPD,UZD(4),RPD(4),3,3)
      CALL INTERP(AD,RPD,AD(4),RPD(4),3,3)
      RPD(5)=RM(JK,NUMBR)
      ZPD(5)=ZM(JK,NUMBR)
      PD(5)=P(JK,NUMBR,1)
      RHOD(5)=RHO(JK,NUMBR,1)
      URD(5)=UR(JK,NUMBR,1)
      UZD(5)=UZ(JK,NUMBR,1)
      AD(5)=A(JK,NUMBR,1)
      RPD(6)=RM(JK,IC2)
      ZPD(6)=ZM(JK,IC2)
      PD(6)=P(JK,IC2,1)
      RHOD(6)=RHO(JK,IC2,1)
      URD(6)=UR(JK,IC2,1)
      UZD(6)=UZ(JK,IC2,1)
      AD(6)=A(JK,IC2,1)
      LSUB(1)=1
      LSUB(2)=2
      LSUB(3)=3
      LSUB(4)=7
      LSUB(5)=8
      LSUB(6)=9
      LSUB(7)=10
      LSUB(8)=11
      LSUB(9)=12
      GO TO 142
  140 CONTINUE
C     INCREMENT COLUMN +,-,ONE
      IAXES(I,J)=4
      DO202K=1,9
  202 LSUB(K)=K
      IE=4
      IF=5
      IG=6
      CALL TRNSFR(IART,1,JK)
  142 CONTINUE
C     SET UP DOMAIN POINTS ON ROW LK
      IE=7
      IF=8
      IG=9
      CALL TRNSFR(IART,1,LK)
C     DETERMINE ROW MK
      IE=10
      IF=11
```

```
      IG=12
      CALL TRNSFR(IART,1,MK)
      RETURN
  120 CONTINUE
C     SCAN ROWS DECREASING TO FIND ROW JK
      DO 122 K=1,NOZ
      JK=NOZ-K+1
      IF(ZM(JK,NUMBR).LT.ZBM(I,J,1))GO TO 124
  122 CONTINUE
      WRITE(6,8000)I,J
      CALL EXIT
  124 LK=JK-1
      MK=JK-2
      GO TO 116
C     HORIZONTAL INTERSECTION
  300 CONTINUE
      NUMBR=INUM(I,J,1)
C     ASSUME THAT THE COLUMNS WILL INCREASE
      ICOL=1
      IF((ANGLE.GT..25*PI).AND.(ANGLE.LE..75*PI))ICOL=2
      GO TO(310,320),ICOL
  310 DO 312 K=1,NOR
      JK=K
      IF(RM(NUMBR,K).GT.RBM(I,J,1))GO TO 314
  312 CONTINUE
      WRITE(6,8000)I,J
      CALL EXIT
  314 LK=JK+1
      MK=JK+2
  316 CONTINUE
C     SET UP BOUNDARY DOMAIN POINTS
      ID=1
      ID1=2
      ID2=3
      IS=I
      KS=J-1
      KS1=J
      KS2=J+1
      CALL REPLCT(1)
C     DETERMINE IF THE COLUMN JK CUTS THE POSITIVE R
C     PRIME AXIS,IAXIS=1,POSITIVE,IAXIS=2,NEGATIVE,
C     IAXIS=3,POSITIVE NO CUT,IAXIS=4,NEGATIVE NO CUT
      IAXIS=1
      IF(((ANGLE.GT..25*PI).AND.(ANGLE.LE..5*PI)).OR.
     .((ANGLE.GT.1.25*PI).AND.(ANGLE.LE.1.5*PI)))IAXIS=2
      RA=RM(NUMBR,JK)
      IAXES(I,J)=IAXIS
      GO TO(330,340),IAXIS
  330 CONTINUE
      CALL SET(ANGLE,NUMBR,IART,IAXIS,IC1,IC2)
C     DETERMINE IF FIRST ROW CUTS POSITIVE AXIS
      VALUE1=AMAX1(RBM(I,J,1),RBM(I,J+1,1),RA)
      VALUE2=AMIN1(RBM(I,J,1),RBM(I,J+1,1),RA)
      IF((RA.EQ.VALUE1).OR.(RA.EQ.VALUE2))GO TO 332
C     SET UP DOMAIN POINTS ON COLUMN JK
      RPD(4)=RM(IC1,JK)
      ZPD(4)=ZM(IC1,JK)
      PD(4)=P(IC1,JK,1)
```

```
      RHOD(4)=RHO(IC1,JK,1)
      URD(4)=UR(IC1,JK,1)
      UZD(4)=UZ(IC1,JK,1)
      AD(4)=A(IC1,JK,1)
      RPD(5)=RM(NUMBR,JK)
      ZPD(5)=ZM(NUMBR,JK)
      PD(5)=P(NUMBR,JK,1)
      RHOD(5)=RHO(NUMBR,JK,1)
      URD(5)=UR(NUMBR,JK,1)
      UZD(5)=UZ(NUMBR,JK,1)
      AD(5)=A(NUMBR,JK,1)
C     INTERPOLATE FOR BOUNDARY POINT PROPERTIES
      RPD(6)=RM(NUMBR,JK)
      ZPD(6)=ZBM(I,J,1)+(ZBM(I,J+1,1)-ZBM(I,J,1))*
     .(RPD(6)-RBM(I,J,1))/(RBM(I,J+1,1)-RBM(I,J,1))
      CALL INTERP(PD,ZPD,PD(6),ZPD(6),3,3)
      CALL INTERP(RHOD,ZPD,RHOD(6),ZPD(6),3,3)
      CALL INTERP(URD,ZPD,URD(6),ZPD(6),3,3)
      CALL INTERP(UZD,ZPD,UZD(6),ZPD(6),3,3)
      CALL INTERP(AD,ZPD,AD(6),ZPD(6),3,3)
      LSUB(1)=1
      LSUB(2)=2
      LSUB(3)=3
      LSUB(4)=7
      LSUB(5)=8
      LSUB(6)=9
      LSUB(7)=10
      LSUB(8)=11
      LSUB(9)=12
      GO TO 334
  332 CONTINUE
C     SET COLUMN JK
      IAXES(I,J)=3
      DO203K=1,9
  203 LSUB(K)=K
      IE=4
      IF=5
      IG=6
      CALL TRNSFR(IART,1,JK)
  334 CONTINUE
C     SET UP DOMAIN POINTS ON COLUMN LK
      IE=7
      IF=8
      IG=9
      CALL TRNSFR(IART,1,LK)
C     SET UP COLUMN MK
      IF=10
      IF=11
      IG=12
      CALL TRNSFR(IART,1,MK)
      RETURN
  340 CONTINUE
C     DETERMINE IF FIRST ROW CUTS NEGATIVE R AXIS
      CALL SET(ANGLE,NUMBR,IART,IAXIS,IC1,IC2)
      VALUE1=AMAX1(RBM(I,J,1),RBM(I,J-1,1),RA)
      VALUE2=AMIN1(RBM(I,J,1),RBM(I,J-1,1),RA)
      IF((RA.EQ.VALUE1).OR.(RA.EQ.VALUE2))GO TO 342
C     INTERPOLATE FOR PROPERTIES
```

```
      RPD(4)=RM(NUMBR,JK)
      ZPD(4)=ZBM(I,J,1)+(ZBM(I,J+1,1)-ZBM(I,J,1))*
     .(RPD(4)-RBM(I,J,1))/(RBM(I,J+1,1)-RBM(I,J,1))
      CALL  INTERP(PD,ZPD,PD(4),ZPD(4),3,3)
      CALL  INTERP(RHOD,ZPD,RHOD(4),ZPD(4),3,3)
      CALL  INTERP(URD,ZPD,URD(4),ZPD(4),3,3)
      CALL  INTERP(UZD,ZPD,UZD(4),ZPD(4),3,3)
      CALL  INTERP(AD,ZPD,AD(4),ZPD(4),3,3)
      RPD(5)=RM(NUMBR,JK)
      ZPD(5)=ZM(NUMBR,JK)
      PD(5)=P(NUMBR,JK,1)
      RHOD(5)=RHO(NUMBR,JK,1)
      URD(5)=UR(NUMBR,JK,1)
      UZD(5)=UZ(NUMBR,JK,1)
      AD(5)=A(NUMBR,JK,1)
      RPD(6)=RM(IC2,JK)
      ZPD(6)=ZM(IC2,JK)
      PD(6)=P(IC2,JK,1)
      RHOD(6)=RHO(IC2,JK,1)
      URD(6)=UR(IC2,JK,1)
      UZD(6)=UZ(IC2,JK,1)
      AD(6)=A(IC2,JK,1)
      LSUB(1)=1
      LSUB(2)=2
      LSUB(3)=3
      LSUB(4)=7
      LSUB(5)=8
      LSUB(6)=9
      LSUB(7)=10
      LSUB(8)=11
      LSUB(9)=12
      GO TO 344
  342 CONTINUE
      IAXES(I,J)=4
      DO204K=1,9
  204 LSUB(K)=K
C     SET COLUMN JK
      IE=4
      IF=5
      IG=6
      CALL TRNSFR(IART,1,JK)
  344 CONTINUE
C     FORM COLUMN LK
      IE=7
      IF=8
      IG=9
      CALL TRNSFR(IART,1,LK)
C     SET COLUMN MK
      IE=10
      IF=11
      IG=12
      CALL TRNSFR(IART,1,MK)
      RETURN
  320 CONTINUE
C     SCAN DECREASING COLUMNS TO FIND COLUMN JK
      DO 322 K=1,NOR
      JK=NOR-K+1
      IF(RM(NUMBR,JK).LT.RBM(I,J,1))GO TO 324
```

```
  322 CONTINUE
      WRITE(6,8000)I,J
      CALL EXIT
  324 LK=JK-1
      MK=JK-2
      GO TO 316
 8000 FORMAT(1H ,31HCOLUMN AND ROW SEARCH FAILED IN,
     .11H SUB BOUNDY,I2,1H,,I2)
      END
```

```
      SUBROUTINE BSERCH(DELR,DELZ,M)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      PI=3.1415926535898
C     DETERMINE HORIZONTAL AND VERTICAL MESH-BOUNDARY INTERSECTIONS
      DO 600 I=1,NBP
      KSPCE=LCODE(I)-1
      NBR=1
      ZBM(I,NBR,M)=ZB(I,1,M)
      RBM(I,NBR,M)=RB(I,1,M)
      DO 602 J=1,KSPCE
      BETA=ATAN2((ZB(I,J+1,M)-ZB(I,J,M)),(RB(I,J+1,M)-RB(I,J,M)))
      IF(BETA.LE.(-PI/4.))BETA=BETA+2.*PI
C     ASSUME A CONSTANT Z LINE INTERSECTION,IART=2
      IART=2
      IF(((BETA.GT.(-PI/4.)).AND.(BETA.LE.(PI/4.))).OR.
     .((BETA.GT.(.75*PI)).AND.(BETA.LE.(1.25*PI))))IART=1
      GO TO (610,620),IART
  620 CONTINUE
C     HORIZONTAL INTERSECTION
      DO 630 L=1,NOZ
      KL=NOZ-L+1
      ZA=ZM(L,1)
      IF((BETA.GT.1.25*PI).AND.(BETA.LE.1.75*PI))ZA=ZM(KL,1)
      VALUE1=AMAX1(ZB(I,J+1,M),ZB(I,J,M),ZA)
      VALUE2=AMIN1(ZB(I,J+1,M),ZB(I,J,M),ZA)
      IF((ZA.EQ.VALUE1).OR.(ZA.EQ.VALUE2)) GO TO 630
      NBR=NBR+1
      ZBM(I,NBR,M)=ZA
      RBM(I,NBR,M)=RB(I,J,M)+(RB(I,J+1,M)-RB(I,J,M))*
     .(ZA-ZB(I,J,M))/(ZB(I,J+1,M)-ZB(I,J,M))
      ITYPE(I,NBR,M)=IART
      INUM(I,NBR,M)=L
      IF((BETA.GT.1.25*PI).AND.(BETA.LE.1.75*PI))INUM(I,NBR,M)=KL
  630 CONTINUE
      GO TO 602
  610 CONTINUE
C     COLUMN INTERSECTION
      DO 640 L=1,NOR
      KL=NOR-L+1
      RA=RM(1,L)
      IF((BETA.GT..75*PI).AND.(BETA.LE.1.25*PI))RA=RM(1,KL)
      VALUE1=AMAX1(RB(I,J+1,M),RB(I,J,M),RA)
      VALUE2=AMIN1(RB(I,J+1,M),RB(I,J,M),RA)
      IF((RA.EQ.VALUE1).OR.(RA.EQ.VALUE2)) GO TO 640
      NBR=NBR+1
      RBM(I,NBR,M)=RA
      ZBM(I,NBR,M)=ZB(I,J,M)+(ZB(I,J+1,M)-ZB(I,J,M))*
     .(RA-RB(I,J,M))/(RB(I,J+1,M)-RB(I,J,M))
      ITYPE(I,NBR,M)=IART
```

```
      INUM(I,NBR,M)=L
      IF((BETA.GT..75*PI).AND.(BETA.LE.1.25*PI))INUM(I,NBR,M)=KL
  640 CONTINUE
  602 CONTINUE
      NBR=NBR+1
      ZBM(I,NBR,M)=ZB(I,KSPCE+1,M)
      RBM(I,NBR,M)=RB(I,KSPCE+1,M)
      LBPT(I)=NBR
      ITYPE(I,1,M)=ITYPE(I,2,M)
      ITYPE(I,NBR,M)=ITYPE(I,NBR-1,M)
  600 CONTINUE
C     DETERMINE ANGLE OF BOUNDARY
      DO 702 K=1,NBP
      KSPCE=LBPT(K)-1
      KP=LBPT(K)
      THETA(K,1,M)=ATAN2((ZBM(K,2,M)-ZBM(K,1,M)),
     .(RBM(K,2,M)-RBM(K,1,M)))
      IF(THETA(K,1,M).LT.0.)THETA(K,1,M)=THETA(K,1,M)+2.*PI
      THETA(K,KP,M)=ATAN2((ZBM(K,KP,M)-ZBM(K,KP-1,M)),
     .(RBM(K,KP,M)-RBM(K,KP-1,M)))
      IF(THETA(K,KP,M).LT.0.)THETA(K,KP,M)=THETA(K,KP,M)+2.*PI
      DO 702 L=2,KSPCE
      THETA(K,L,M)=ATAN2((ZBM(K,L+1,M)-ZBM(K,L-1,M)),
     .(RBM(K,L+1,M)-RBM(K,L-1,M)))
  702 IF(THETA(K,L,M).LT.0.)THETA(K,L,M)=THETA(K,L,M)+2.*PI
C     DETERMINE IF BOUNDARY POINT IS ACTIVE
      VALUE=AMAX1(DELR,DELZ)*.001
      DMEAS=AMIN1(DELR,DELZ)*.9
      DO 430 I=1,NBP
      KSPCE=LBPT(I)
      DO 430 J=1,KSPCE
      NBACT(I,J,M)=1
      NBODD(I,J,M)=0
      RMA=RBM(I,J,M)
      ZMA=ZBM(I,J,M)
      DO 430 K=1,NBP
      IF(K.EQ.I)GO TO 430
      KB=LBPT(K)-1
      DO 434 L=1,KB
      ANGLE=ATAN2((ZBM(K,L+1,M)-ZBM(K,L,M)),
     .(RBM(K,L+1,M)-RBM(K,L,M)))
      ZPR=-(RMA-RBM(K,L,M))*SIN(ANGLE)+
     .(ZMA-ZBM(K,L,M))*COS(ANGLE)
      RPR=(RMA-RBM(K,L,M))*COS(ANGLE)+(ZMA-ZBM(K,L,M))*
     .SIN(ANGLE)
      DIST=SQRT((ZPR**2)+(RPR**2))
      IF(ZPR.GT.(-VALUE)) GO TO 434
      NBACT(I,J,M)=0
  434 IF(DIST.LT.DMEAS)NBODD(I,J,M)=1
  430 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CNRBD(I,J,K,IBOUND,KNT,PHI,BETA)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/XCHNG/IS,KS,KS1,KS2,ID,ID1,ID2
      COMMON/YCHNG/KB,KP,IR,IC1,IC2,IE1,IE2,IE3
C     THE PURPOSE OF THIS SUBROUTINE IS TO SET UP THE
C     DOMAIN POINTS FOR THE CORNER
C     SET GRID POINTS
      LSUB(1)=7
      LSUB(2)=4
      LSUB(3)=1
      LSUB(4)=8
      LSUB(5)=5
      LSUB(6)=2
      LSUB(7)=9
      LSUB(8)=6
      LSUB(9)=3
      GO TO(100,300),KNT
C     SET UP DOMAIN FOR FIRST CORNER POINT
  100 CONTINUE
      ID=7
      ID1=4
      ID2=1
      KS=J
      KS1=J+1
      KS2=J+2
      IS=I
      CALL REPLCT(1)
C     SET UP COLUMNS OR ROWS FOR DOMAIN
      KTYPE=ITYPE(I,J+1,1)
      IPT1=INUM(I,J+1,1)
      IPT2=INUM(I,J+2,1)
      PHI=THETA(I,J,1)
      CALL SOLCT(I,K,1,J,IR1,IR2,IPT1,IPT2,KPT1,KPT2,PHI,
     .BETA,KTYPE,LTYPE)
C     SET UP REMAINING DOMAIN POINTS
      KB=K
      KP=KPT1
      IR=IR1
      IC1=IPT2
      IC2=IPT1
      IE1=2
      IE2=5
      IE3=8
      CALL TRSCNR(1,KTYPE)
      KP=KPT2
      IR=IR2
      IE1=3
      IE2=6
      IE3=9
      CALL TRSCNR(1,KTYPE)
      RETURN
```

```
C       SET UP DOMAIN FOR SECOND CORNER POINT
 300    CONTINUE
        ID=7
        ID1=4
        ID2=1
        KS=J-2
        KS1=J-1
        KS2=J
        IS=1
        CALL REPLCT(1)
C       SET UP COLUMNS OR ROWS FOR DOMAIN
        KTYPE=ITYPE(I,J-1,1)
        IPT1=INUM(I,J-1,1)
        IPT2=INUM(I,J-2,1)
        PHI=THETA(I,J,1)
        CALL SOLCT(I,K,1,J,IR1,IR2,IPT1,IPT2,KPT1,KPT2,PHI,
       .BETA,KTYPE,LTYPE)
C       SET UP REMAINING DOMAIN POINTS
        KB=K
        KP=KPT1
        IR=IR1
        IC1=IPT2
        IC2=IPT1
        IF1=8
        IE2=5
        IE3=2
        CALL TRSCNR(1,KTYPE)
        KP=KPT2
        IR=IR2
        IE1=9
        IE2=6
        IE3=3
        CALL TRSCNR(1,KTYPE)
        RETURN
        END
```

```
      SUBROUTINE COMPEQ(IBCUND,TIME,DELTA,KAXIS,CONST,
     .NCOUNT,KPRINT)
      REALKAXIS
C
C     COMPUTE GENERAL POINT COMPATIBILITY EQUATIONS
C
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5), PAR(5),PAZ(5)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/STATE/PINIT,RHOINT,AINIT
      COMMON/SETUP/RAS(4),RAC(4),R(5),DELP,DELRH,DELUR, DELUZ
      DATA KK/1/,PI/3.14159265/
      CALL BEGIN(IBOUND,DELTA,KAXIS,CONST)
      ROR=RNEW
      ZOR=ZNEW
C     COMPUTE GENERAL POINT
      SUM1=0.
      SUM2=0.
      SUM3=0.
      DO 110 I=1,4
      SUM1=SUM1+RAS(I)
      SUM2=SUM2+RAC(I)
      SUM3=SUM3+R(I)
  110 CONTINUE
      XA=SOUND(RHNEW,PNEW)
      IF(ROR.NE.0.)GOTO10
      PART=PURR(5)
      GOTO11
   10 PART=URNEW/ROR
   11 CONTINUE
      DELP1=-.25*SUM2*DELUR-.25*SUM1*DELUZ-.5*RHNEW*
     .XA*XA*KAXIS*PART*DELTA-.25*XA*RHNEW*(URG(3)-
     .URG(1)+UZG(4)-UZG(2))+R(5)-.5*SUM3
      DELP1=DELP1/CONST
      PNEW1=PG(5)+DELP1
      XA=SOUND(RHNEW,PNEW1)
      DELUR1=-.5*(RAS(1)-RAS(3))*DELUZ-.5*RHNEW*XA*
     .(2.*URG(5)-URG(1)-URG(3))+R(3)-R(1)
      DELUR1=DELUR1/(.5*(2.*RHNEW*XA+RAC(1)-RAC(3)))
      URNEW1=URG(5)+DELUR1
      IF(ROR.NE.0.)GOTO20
      PART=PURR(5)
      GOTO21
   20 PART=URNEW1/ROR
   21 CONTINUE
      DELUZ1=-.5*(RAC(2)-RAC(4))*DELUR1-.5*RHNEW*XA*
     .(2.*UZG(5)-UZG(2)-UZG(4))+R(4)-R(2)
      DELUZ1=DELUZ1/(.5*(2.*RHNEW*XA+RAS(2)-RAS(4)))
      DELRH1=4.*DELP1*CONST+.5*SUM2*DELUR1+.5*SUM1*
     .DELUZ1+RHNEW*XA*XA*KAXIS*PART*DELTA+.5*XA*
     .RHNEW*(URG(3)-URG(1)+UZG(4)-UZG(2))-2.*((XA/AG(5))**2)*
```

```
     .R(5)+SUM3
      DELRH1=DELRH1/(2.*XA*XA)
      UZNEW1=UZG(5)+DELUZ1
      RHNEW1=RHOG(5)+DELRH1
      IF(ITEST.NE.1)GO TO 132
      ITEST=2
      GO TO 130
  132 RATIO=.01
      IF(ABS(URNEW).GT.0.)RATIO=ABS((URNEW1-URNEW)/URNEW)
      IF(RATIO.GT..01)GO TO 130
      IF(ABS(UZNEW).GT.0.)RATIO=ABS((UZNEW1-UZNEW)/UZNEW)
      IF(RATIO.GT..01)GO TO 130
      IF(ABS((PNEW1-PNEW)/PNEW).GT..01)GO TO 130
      IF(ABS((RHNEW1-RHNEW)/RHNEW).GT..01)GO TO 130
      ACOUNT=2
  130 PNEW=PNEW1
      RHNEW=RHNEW1
      URNEW=URNEW1
      UZNEW=UZNEW1
      IF(KPRINT.GT.0)CALL PRINT
      RETURN
      END
```

```
      SUBROUTINE CONVRT(IBOUND)
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),DAR(15),
     .DAZ(15)
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5),PAR(5),
     .PAZ(5)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      DIMENSION AURR(5),AUZR(5),AURZ(5),AUZZ(5),AAR(5),
     .AAZ(5),BURR(5),BUZR(5),BURZ(5),BUZZ(5)
C     THE PURPOSE OF THIS ROUTINE IS TO TRANSFORM THE DERIVATIVES
C     TO THE PRIMED SYSTEM
C     STORE DERIVATIVES
      DO 100 K=1,5
      IF((IBOUND.GT.0).AND.(K.EQ.2))GO TO 100
C     IBOUND=1,DROP 2,IBOUND=2,DROP 3,IBOUND=3,DROP 1
      IF(((IBOUND.EQ.2).OR.(IBOUND.EQ.5)).AND.(K.EQ.3))GO TO 100
      IF(((IBOUND.EQ.3).OR.(IBOUND.EQ.6)).AND.(K.EQ.1))GO TO 100
      AURR(K)=PURR(K)
      AUZR(K)=PUZR(K)
      AURZ(K)=PURZ(K)
      AUZZ(K)=PUZZ(K)
      AAR(K)=PAR(K)
      AAZ(K)=PAZ(K)
C     CONVERT DERIVATIVES TO PRIME COORDINATES
      PAR(K)=AAR(K)*COS(ANGLE)+AAZ(K)*SIN(ANGLE)
      PAZ(K)=-AAR(K)*SIN(ANGLE)+AAZ(K)*COS(ANGLE)
      BURR(K)=AURR(K)*COS(ANGLE)+AURZ(K)*SIN(ANGLE)
      BURZ(K)=-AURR(K)*SIN(ANGLE)+AURZ(K)*COS(ANGLE)
      BUZR(K)=AUZR(K)*COS(ANGLE)+AUZZ(K)*SIN(ANGLE)
      BUZZ(K)=-AUZR(K)*SIN(ANGLE)+AUZZ(K)*COS(ANGLE)
      PURR(K)=BURR(K)*COS(ANGLE)+BUZR(K)*SIN(ANGLE)
      PUZR(K)=-BURR(K)*SIN(ANGLE)+BUZR(K)*COS(ANGLE)
      PURZ(K)=BURZ(K)*COS(ANGLE)+BUZZ(K)*SIN(ANGLE)
      PUZZ(K)=-BURZ(K)*SIN(ANGLE)+BUZZ(K)*COS(ANGLE)
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CORCOM(IBCUND,TIME,DELTA,KAXIS,CONST,
     .NCOUNT,KPRINT,I,J)
C     THIS ROUTINE INTEGRATES CORNER POINT COMPATIBILITY EQUATIONS
      REAL KAXIS
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBOCD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/SETUP/RAS(4),RAC(4),R(5),DELP,DELRH,DELUR,DELUZ,S(5)
      COMMON/EXT/URR,UZZ,Y,RRAT,ARAT/STATE/PINIT,RHOINT,AINIT
      COMMON/PARTIAL/PURR(5),PUPZ(5),PUZR(5),PUZZ(5),PAR(5),PAZ(5)
      DIMENSION DIST(3),F(3)
      DATA KK/1/
      PI=3.141592653589793
      TOL=0.0000001
      TOL=0.0001
      CALL BEGIN(IBOUND,DELTA,KAXIS,CONST)
      ROR=RNEW
      ZOR=ZNEW
      IF(JCODE(I).EQ.3)KK=1
      GO TO(100,110),KK
  100 KK=2
      K=I-1
      IF(J.EQ.LBPT(I))K=I+1
      IF(K.EQ.0) K=NBP
      IF(MCODE(I).EQ.0) GO TO 200
      RORA=ROR*WIDTH
      ZORA=ZOR*WIDTH
      TIME1=TIME*WIDTH/AINIT
      CALL EXACT(RORA,ZORA,TIME1)
      Z=-URR*SIN(ANGLE)+UZZ*COS(ANGLE)
      IF(JCODE(I).NE.7)GO TO 110
      W=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      PP=Y
      U=RRAT
      IK=1
      IF(J.EQ.LBPT(I))IK=2
      GO TO(120,130),IK
  120 J1=J+1
      J2=J+2
      GO TO 140
  130 J1=J-1
      J2=J-2
  140 CONTINUE
      ROR1=RBM(I,J1,1)*WIDTH
      ZOR1=ZBM(I,J1,1)*WIDTH
      ROR2=RBM(I,J2,1)*WIDTH
      ZOR2=ZBM(I,J2,1)*WIDTH
      DIST(1)=0.
      DIST(2)=(RBM(I,J1,1)-RBM(I,J,1))*COS(ANGLE)+(ZBM(I,J1,1)-
```

```
     .ZBM(I,J,1))*SIN(ANGLE)
      DIST(3)=(RBM(I,J2,1)-RBM(I,J,1))*COS(ANGLE)+
     .(ZBM(I,J2,1)-ZBM(I,J,1))*SIN(ANGLE)
      F(1)=W
      CALL EXACT(ROR1,ZOR1,TIME1)
      F(2)=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      CALL EXACT(ROR2,ZOR2,TIME1)
      F(3)=URR*COS(ANGLE)+UZZ*SIN(ANGLE)
      CALL FINITE(1,2,3,F,DIST,DZZ,ART,ART)
      GO TO 110
  200 CONTINUE
      KV=I
      NK=JCODE(KV)
C     ALLOW FOR OTHER CORNER POINTS
      KSPCE=LCODE(KV)
      NTIME=KCODE(KV,1)
      CALL FIND(KV,J1,NK,NTIME,KSPCE,ROR,ZOR,Z,W,DZZ,DWR,
     .ART,TIME)
      IF(IK.EQ.2)DZZ=ART
      KV=K
      NK=JCODE(KV)
      NTIME=KCODE(KV,1)
      KSPCE=LCODE(KV)
C     FIND PRESSURE
      CALL FIND(KV,J1,NK,NTIME,KSPCE,ROR,ZOR,Y,ART,ART,
     .ART,ART,TIME)
  110 CONTINUE
C     INTEGRATE COMPATIBILITY EQUATIONS
      NK=IBOUND-1
      GO TO (300,310,320,340,350),NK
  300 CONTINUE
C     IBOUND=2
      XA=SOUND(RHNEW,Y)
      IF(ROR.EQ.0.) GO TO 302
      PART=URNEW/ROR
      GO TO 304
  302 PART=PURR(5)
  304 CONTINUE
      DELUR1=-2.*(Y-PG(5))*CONST-RHNEW*XA*XA*KAXIS*
     .PART*DELTA-RHNEW*XA*(URG(5)-URG(1)+UZG(4)-UZG(5))-
     .(Z-UZG(5))*(-RHNEW*XA+RAS(1)+RAS(4))+2.*(
     .P(5)-P(1)-P(4))
      DELUR1=DELUR1/(RHNEW*XA+RAC(1)+RAC(4))
      URNEW1=URG(5)+DELUR1
      IF(ROR.EQ.0.)GO TO 306
      PART=URNEW1/ROR
      GO TO 308
  306 PART=PURR(5)
  308 CONTINUE
      DELRH1=(Y-PG(5))*CONST-R(5)*(XA*XA/(AG(5)*AG(5))-1.)
      DELRH1=DELRH1/(XA*XA)
      GO TO 500
  310 CONTINUE
      XA=SOUND(RHNEW,Y)
      IF(ROR.EQ.0.) GO TO 312
      PART=URNEW/ROR
      GO TO 314
  312 PART=PURR(5)
```

```
314   CONTINUE
      DELUR1=-2.*(Y-PG(5))*CONST-RHNEW*XA*XA*KAXIS*
     .PART*DELTA-RHNEW*XA*(URG(3)-URG(5)+UZG(4)-UZG(5))-
     .(Z-UZG(5))*(-RHNEW*XA+RAS(3)+RAS(4))+2.*(R(5)-
     .R(3)-R(4))
      DELUR1=DELUR1/(-RHNEW*XA+RAC(3)+RAC(4))
      URNEW1=URG(5)+DELUR1
      IF(ROR.EQ.C.)GO TO 316
      PART=URNEW1/ROR
      GO TO 318
  316 PART=PURR(5)
  318 CONTINUE
      DELRH1=(Y-PG(5))*CONST-R(5)*(XA*XA/(AG(5)*AG(5))-1.)
      DELRH1=DELRH1/(XA*XA)
500   CONTINUE
      RHNEW1=RHOG(5)+DELRH1
      URNEW1=DELUR1+URG(5)
      IF(ITEST.NE.1) GO TO 510
      ITEST=2
      GO TO 520
510   RATIO=.01
      IF(ABS(URNEW).GT.0.)RATIO=ABS((URNEW1-URNEW)/URNEW)
      IF(RATIO.GT.TOL)GOTO520
      IF(ABS((RHNEW1-RHNEW)/RHNEW).GT.TOL)GOTO520
      NCOUNT=2
      KK=1
520   URNEW=URNEW1
      RHNEW=RHNEW1
      PNEW=Y
      UZNEW=Z
      ANEW=SOUND(RHNEW,PNEW)
      IF(KPRINT.GT.0) CALL PRINT
      RETURN
  320 CONTINUE
340   XA=SOUND(RHNEW,PP)
      IF(ROR.EQ.0.)GO TO 342
      PART=PURR(5)
      GO TO 344
  342 PART=CZZ
  344 CONTINUE
      DELRH1=(2.*(PP-PG(5))*CONST+
     ..5*(RHNEW*XA+RAC(1)+RAC(4))*(W-URG(5))+
     ..5*PHNEW*XA*(UZG(4)-UZG(5)+URG(5)-URG(1))+
     ..5*(-RHNEW*XA+RAS(1)+RAS(4))*(Z-UZG(5))+
     ..5*RHNEW*XA*XA*KAXIS*PART*DELTA+
     .R(1)+R(4)-XA*XA*R(5)/(AG(5)*AG(5)))/(XA*XA)
      RHNEW1=DELRH1+RHOG(5)
620   IF(ITEST.NE.1)GO TC 600
      ITEST=2
      GO TO 610
  600 IF(ABS((RHNEW1-RHNEW)/RHNEW).GT.TOL)GOTO610
      NCOUNT=2
      KK=1
610   RHNEW=RHNEW1
      PNEW=PP
      URNEW=W
      UZNEW=Z
      ANEW=SOUND(RHNEW,PNEW)
```

```
      IF(KPRINT.GT.0)CALL PRINT
      RETURN
350   XA=SOUND(RHNEW,PP)
      IF(ROR.EQ.0.) GO TO 352
      PART=W/ROR
      GO TO 354
352   PART=PURR(5)
354   CONTINUE
      DELRH1=(2.*(PP-PG(5))*CONST+
     ..5*(-RHNEW*XA+RAC(3)+RAC(4))*(W-URG(5))+
     ..5*RHNEW*XA*(URG(3)+UZG(4)-UZG(5)-URG(5))+
     ..5*(-RHNEW*XA+RAS(3)+RAS(4))*(Z-UZG(5))-
     .XA*XA*R(5)/(AG(5)*AG(5))+R(3)+R(4)+
     ..5*RHNEW*XA*XA*KAXIS*PART*DELTA)/(XA*XA)
      RHNEW1=DELRH1+RHOG(5)
      GO TO 620
      END
```

```
      SUBROUTINECORNPT(I,J,CONST,KAXIS,TIME,DELTA,KPRINT,IBOUND)
      REALKAXIS
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),
     .AG(5),PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,
     .RNEW,ZNEW
C     THE PURPOSE OF THIS SUBROUTINE IS TO COMPUTE THE PROPERTIES
C     AT A CORNER POINT
      INDEX=0
      ITEST=1
      IF(NBACT(I,J,2).EQ.0)RETURN
      ANGLE=THETA(I,J,1)
      RNEW=RBM(I,J,1)
      ZNEW=ZBM(I,J,1)
C     SET ADJOINING BOUNDARY
      KNT=1
      IF(LBPT(I).EQ.J)KNT=2
      GO TO(110,120),KNT
C     FIRST CORNER POINT ON BOUNDARY
  110 K=I-1
      IF(K.LT.1)K=NBP
      IB=ITYPE(I,J+1,1)
      GO TO 130
C     SECOND CORNER POINT ON BOUNDARY
  120 K=I+1
      IF(K.GT.NBP)K=1
      IB=ITYPE(I,J-1,1)
  130 CONTINUE
C     SET CHARACTERISTIC LIMIT
C     SET UP CORNER DOMAIN POINTS
      CALL CNRBD(I,J,K,IBOUND,KNT,PHI,BETA)
      IF(KPRINT.EQ.1)CALLPRINT
C     DETERMINE DERIVATIVES AT DOMAIN POINTS
      CALL PREPRS(2,IDUM,IDUM,PHI,BETA,KNT)
C     GUESS AT PROPERTIES
      CALL GUESS(1,I,J)
  300 CONTINUE
      CALL INTEG(I,J,DELTA)
      CALL PHYCHR(IBOUND,DELTA)
C     INTERPOLATE FOR PROPERTIES
      CALL INPROP(IB,IBOUND)
      CALL CONVRT(IBOUND)
      NCOUNT=1
      CALL CORCOM(IBOUND,TIME,DELTA,KAXIS,CONST,NCOUNT,
     .KPRINT,I,J)
      INDEX=INDEX+1
      IF(INDEX.LT.25)GO TO 1010
      WRITE(6,8700)I,J
      CALL PRINT
```

```
      CALL EXIT
 1010 IF(NCOUNT.EQ.1)GO TO 300
      ITEST=1
C     STORE PROPERTIES IN NEW TIME PLANE
      PB(I,J,2)=PNEW
      RHOB(I,J,2)=RHNEW
      URB(I,J,2)=URNEW*COS(ANGLE)-UZNEW*SIN(ANGLE)
      UZB(I,J,2)=URNEW*SIN(ANGLE)+UZNEW*COS(ANGLE)
      AB(I,J,2)=SOUND(RHNEW,PNEW)
      RBM(I,J,2)=RNEW
      ZBM(I,J,2)=ZNEW
      RETURN
 9700 FORMAT(1H ,31HPROGRAM LOOPING AT CORNER POINT,I2,
     .1H,,I2)
      END
```

```
      SUBROUTINE DERTVE(RPD,ZPD,URD,UZD,AD,DURR,DUZR,
     .DAR,DURZ,DUZZ,DAZ,IAXIS,ALPHA,BETA)
      DIMENSION RPD(1),ZPD(1),URD(1),UZD(1),AD(1),DURR(1),DUZP(1),
     .DAR(1),DURZ(1),DUZZ(1),DAZ(1),DIST(15),PURR(15),PUZR(15),PAR(15)
C     THE PURPOSE OF THIS SUBROUTINE IS TO THE DOMAIN
C     DERIVATIVES
      GO TO (110,150,130,170),IAXIS
C     CONSIDER POSITIVE R INTERSECTION
C     COMPUTE DERIVATIVES IN THE R DIRECTIONS FIRST
  110 DO 112 K=7,10,3
      K1=K+1
      K2=K+2
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  112 CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE R PRIME DERIVATIVES
      CALL RPRIME(DIST,RPD,ZPD,PURR,PUZR,PAR,URD,UZD,AD)
C     COMPUTE DERIVATIVES ON BOUNDARY WITH LAGRANGE
      DIST(6)=SQRT((RPD(6)-RPD(1))**2+(ZPD(6)-ZPD(1))**2)
      CALL LAGRNE(2,6,3,URD,DIST,ART,PURR(6),ART)
      CALL LAGRNE(2,6,3,UZD,DIST,ART,PUZR(6),ART)
      CALL LAGRNE(2,6,3,AD,DIST,ART,PAR(6),ART)
C     COMPUTE UNEVEN R DERIVATIVES WITH LAGRANGE
      CALL LAGRNE(4,5,6,URD,RPD,DURR(4),DURR(5),DURR(6))
      CALL LAGRNE(4,5,6,UZD,RPD,DUZR(4),DUZR(5),DUZR(6))
      CALL LAGRNE(4,5,6,AD,RPD,DAR(4),DAR(5),DAR(6))
C     COMPUTE AXIAL DERIVATIVES
      DO 116 K=4,5
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL FINITE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  116 CALL FINITE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE UNEVEN DERIVATIVES WITH LAGRANGE
      DO 118 K=1,2
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),ART,ART)
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),ART,ART)
  118 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),ART,ART)
      CALL LAGRNE(3,9,12,URD,ZPD,DURZ(3),DURZ(9),DURZ(12))
      CALL LAGRNE(3,9,12,UZD,ZPD,DUZZ(3),DUZZ(9),DUZZ(12))
      CALL LAGRNE(3,9,12,AD,ZPD,DAZ(3),DAZ(9),DAZ(12))
C     DETERMINE REMAINING DERIVATIVES FROM R PRIME AND Z
      DO 120 K=1,3
      DURR(K)=(PURR(K)-DURZ(K)*BETA)/ALPHA
      DUZR(K)=(PUZR(K)-DUZZ(K)*BETA)/ALPHA
  120 DAR(K)=(PAR(K)-DAZ(K)*BETA)/ALPHA
      DURZ(6)=(PURR(6)-DURR(6)*ALPHA)/BETA
      DUZZ(6)=(PUZR(6)-DUZR(6)*ALPHA)/BETA
      DAZ(6)=(PAR(6)-DAR(6)*ALPHA)/BETA
      RETURN
C     FIND DERIVATIVES WHEN ROW JK DOES NOT INTERSECT
C     POSITIVE R PRIME AXIS
  130 CONTINUE
C     COMPUTE R PRIME DERIVATIVES
      CALL RPRIME(DIST,RPD,ZPD,PURR,PUZR,PAR,URD,UZD,AD)
C     COMPUTE R DERIVATIVES
```

```
      DO 132 K=4,10,3
      K1=K+1
      K2=K+2
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  132 CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE Z DERIVATIVES
      DO 136 K=4,6
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL FINITE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  136 CALL FINITE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE UNEVEN DERIVATIVES IN Z DIRECTION
      DO 138 K=1,3
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),ART,ART)
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),ART,ART)
  138 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),ART,ART)
C     COMPUTE R DERIVATIVES FROM R PRIME AND Z
      DO 140 K=1,3
      DURR(K)=(PURR(K)-DURZ(K)*BETA)/ALPHA
      DUZR(K)=(PUZR(K)-DUZZ(K)*BETA)/ALPHA
  140 DAR(K)=(PAR(K)-DAZ(K)*BETA)/ALPHA
      RETURN
C     CONSIDER NEGATIVE R PRIME INTERSECTION
  150 CONTINUE
C     COMPUTE R PRIME DERIVATIVES
      CALL RPRIME(DIST,RPD,ZPD,PURR,PUZR,PAR,URD,UZD,AD)
C     COMPUTE DERIVATIVE ON BOUNDARY WITH LAGRANGE
      DIST(4)=SQRT((RPD(4)-RPD(1))**2+(ZPD(4)-ZPD(1))**2)
      CALL LAGRNE(4,2,3,URD,DIST,PURR(4),ART,ART)
      CALL LAGRNE(4,2,3,UZD,DIST,PUZR(4),ART,ART)
      CALL LAGRNE(4,2,3,AD,DIST,PAR(4),ART,ART)
C     COMPUTE R DERIVATIVES
      DO 152 K=7,10,3
      K1=K+1
      K2=K+2
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  152 CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE UNEVEN R DERIVATIVE
      CALL LAGRNE(4,5,6,URD,RPD,DURR(4),DURR(5),DURR(6))
      CALL LAGRNE(4,5,6,UZD,RPD,DUZR(4),DUZR(5),DUZR(6))
      CALL LAGRNE(4,5,6,AD,RPD,DAR(4),DAR(5),DAR(6))
C     COMPUTE Z DERIVATIVES
      DO 156 K=5,6
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL FINITE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  156 CALL FINITE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE UNEVEN Z DERIVATIVES
      CALL LAGRNE(1,7,10,URD,ZPD,DURZ(1),DURZ(7),DURZ(10))
      CALL LAGRNE(1,7,10,UZD,ZPD,DUZZ(1),DUZZ(7),DUZZ(10))
      CALL LAGRNE(1,7,10,AD,ZPD,DAZ(1),DAZ(7),DAZ(10))
      DO 158 K=2,3
```

```
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),ART,ART)
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),ART,ART)
  158 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),ART,ART)
C     COMPUTE REMAINING DERIVATIVES FROM R,R-PRIME,Z
      DO 160 K=1,3
      DURR(K)=(PURR(K)-DURZ(K)*BETA)/ALPHA
      DUZR(K)=(PUZR(K)-DUZZ(K)*BETA)/ALPHA
  160 DAR(K)=(PAR(K)-DAZ(K)*BETA)/ALPHA
      DURZ(4)=(PURR(4)-DURR(4)*ALPHA)/BETA
      DUZZ(4)=(PUZR(4)-DUZR(4)*ALPHA)/BETA
      DAZ(4)=(PAR(4)-DAR(4)*ALPHA)/BETA
      RETURN
C     FIND DERIVATIVES WHEN ROW JK DOES NOT INTERSECT
C     NEGATIVE R PRIME AXIS
  170 CONTINUE
C     COMPUTE R PRIME DERIVATIVES
      CALL RPRIME(DIST,RPD,ZPD,PURR,PUZR,PAR,URD,UZD,AD)
C     COMPUTE R DERIVATIVES
      DO 172 K=4,10,3
      K1=K+1
      K2=K+2
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
  172 CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
C     COMPUTE Z DERIVATIVES
      DO 176 K=4,6
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL FINITE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
  176 CALL FINITE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
C     COMPUTE UNEVEN DERIVATIVES IN Z DIRECTION
      DO 178 K=1,3
      K1=K+3
      K2=K+6
      CALL LAGRNE(K,K1,K2,URD,ZPD,DURZ(K),ART,ART)
      CALL LAGRNE(K,K1,K2,UZD,ZPD,DUZZ(K),ART,ART)
  178 CALL LAGRNE(K,K1,K2,AD,ZPD,DAZ(K),ART,ART)
C     COMPUTE R DERIVATIVES FROM R PRIME AND Z
      DO 180 K=1,3
      DURR(K)=(PURR(K)-DURZ(K)*BETA)/ALPHA
      DUZR(K)=(PUZR(K)-DUZZ(K)*BETA)/ALPHA
  180 DAR(K)=(PAR(K)-DAZ(K)*BETA)/ALPHA
      RETURN
      END
```

```
      SUBROUTINE DIFBD(I,J)
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),
     .DAR(15),DAZ(15)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      PI=3.1415926535898
C     THE PURPOSE OF THIS SUBROUTINE IS TO EVALUATE THE DOMAIN
C     DERIVATIVES
      NCODE=ITYPE(I,J,1)
      GO TO(100,300),NCODE
  100 CONTINUE
C     CONSIDER VERTICAL INTERSECTION
      IAXIS=IAXES(I,J)
      ALPHA=COS(ANGLE)
      BETA=SIN(ANGLE)
      CALL DERIVE(RPD,ZPD,URD,UZD,AD,DURR,DUZR,DAR,DUPZ,
     .DUZZ,DAZ,IAXIS,ALPHA,BETA)
      RETURN
C     COMPUTE HORIZONTAL INTERSECTION
  300 CONTINUE
      IAXIS=IAXES(I,J)
      ALPHA=SIN(ANGLE)
      BETA=COS(ANGLE)
      CALL DERIVE(ZPD,RPD,URD,UZD,AD,DURZ,DUZZ,DAZ,
     .DURR,DUZR,DAR,IAXIS,ALPHA,BETA)
      RETURN
      END
```

```
      SUBROUTINE DIFFER
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),
     .DAR(15),DAZ(15)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
C     COMPUTE RADIAL DERIVATIVES FIRST
      DO 100 K=1,3
      K1=K+3
      K2=K+6
      CALL FINITE(K,K1,K2,URD,RPD,DURR(K),DURR(K1),DURR(K2))
      CALL FINITE(K,K1,K2,UZD,RPD,DUZR(K),DUZR(K1),DUZR(K2))
      CALL FINITE(K,K1,K2,AD,RPD,DAR(K),DAR(K1),DAR(K2))
  100 CONTINUE
C     COMPUTE AXIAL DERIVATIVES
      DO 200 K=1,7,3
      K1=K+1
      K2=K+2
      CALL FINITE(K,K1,K2,URD,ZPD,DURZ(K),DURZ(K1),DURZ(K2))
      CALL FINITE(K,K1,K2,UZD,ZPD,DUZZ(K),DUZZ(K1),DUZZ(K2))
      CALL FINITE(K,K1,K2,AD,ZPD,DAZ(K),DAZ(K1),DAZ(K2))
  200 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE   EXACT(R,Z,TIME)
C     THIS ROUTINE COMPUTES BLAST WAVE SOLUTION
      DOUBLE PRECISION VINIT,A,B,C,F,F1,F2,V1,V2,DELTA,TOL
      COMMON/STATE/PRESS,RHO1,AINIT/EXT/UR,UZ,PRAT,RRAT,AA
      DATA ENERGY/1.5482649E+06/,GAMMA/1.4/
      QUAN=ENERGY/RHO1
      XS=SQRT(SQRT(QUAN))*SQRT(TIME)
      US=SQRT(QUAN)/((GAMMA+1.)*XS)
      USA=US/AINIT
      PR=ENERGY/(2.*(GAMMA+1.)*XS*XS*PRESS)
      ALPHA=ATAN2(Z,R)
      X=SQRT((R**2)+(Z**2))/XS
      VINIT=10.0C0000100/28.0D0
      TOL=0.5D-6
      DELTA=1.0D-4
      I=0
      J=1
  120 A=12.CD0*VINIT/5.0D0
      B=(84.0D0*VINIT-3.0D1)/5.0D0
      C=2.4CD0*(1.0D0-1.4D0*VINIT)
      F=((B**(1.0D0/7.0DC))/DSQRT(A*C))-X
      GO TO (200,210,220),J
  200 F1=F
      V1=VINIT
      F3=F1
      SI1=SIGN(1.,F3)
      GO TO 230
  210 F2=F
      V2=VINIT
      F3=F2
      SI2=SIGN(1.,F3)
      IF(DABS(F).LE.TOL)GOTO110
      IF(SI1.NE.SI2) GO TO 220
      F1=F2
      V1=V2
  230 J=2
      VINIT=VINIT+DELTA
      IF(VINIT.LE.(10.0D0/24.0D0))GOTO120
      PRINT1
    1 FORMAT(' ***  FAILURE OF EXACT SOLUTION  ***')
      STOP
  220 IF(DABS(F).LE.TOL)GOTO110
      DELTA=DELTA/2.0D0
      VINIT=VINIT-DELTA
      I=I+1
      IF(I.GE.25)GO TO 110
      GO TO 120
  110 CONTINUE
  130 DENSTY=(8**(5./7.))*((6.-12.*VINIT)**(-10./3.))*(C**(10./3.))
      URATIC=2.4*VINIT*X
      PRATIC=A*((6.-12.*VINIT)**(-7./3.))*(C**(7./3.))
      UR=URATIO*USA*COS(ALPHA)
      UZ=URATIO*USA*SIN(ALPHA)
      PRAT=PRATIC*PR
      RRAT=6.0*DENSTY
      AA=SQRT(PRAT/RRAT)
      RETURN
      END
```

```
      SUBROUTINEFIND(KV,KK,NK,NTIME,KSPCE,ROR,ZOZ,Y,W,DZZ,DWR,ART,TIME)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LRPT(10),IAXES(10,30)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON/ALPHA/WW
      DIMENSION YCON(20),XCON(20),ZCON(20),WCON(20),VCON(20),
     .F(3),XY(3),UCON(20),SCON(20)
C     INTERPOLATE BOUNDARY CONDITIONS
      ETA=ATAN2((ZB(KV,KSPCE,1)-ZB(KV,1,1)),
     .(RB(KV,KSPCE,1)-RB(KV,1,1)))
      IF((ZOR.EQ.ZB(KV,1,1)).AND.(ROR.EQ.RB(KV,1,1)))
     .GO TO 10
      XI=ATAN2((ZOR-ZB(KV,1,1)),(ROR-RB(KV,1,1)))
      DIST=SQRT((ROR-RB(KV,1,1))**2+(ZOR-ZB(KV,1,1))**2)
      DIST=DIST*COS(XI-ETA)
      GO TO 20
   10 DIST=0.
   20 IF(NTIME.EQ.0)GO TO 100
      DO 110 J=1,KSPCE
      DO 120 I=1,NTIME
      YCON(I)=BCON(KV,J,I)
  120 XCON(I)=BTIME(KV,I,1)
      CALL INTERP(YCON,XCON,ZCON(J),TIME,NTIME,3)
  110 CONTINUE
      DO 130 I=1,KSPCE
  130 XCON(I)=BSPCE(KV,I)
      CALL INTERP(ZCON,XCON,DMY,DIST,KSPCE,3)
      GO TO(132,134,134,134,134,134,136),NK
  132 Y=DMY
      RETURN
  134 Y=DMY
      WW=THETA(KV,KK,1)
      RETURN
  136 Y=DMY
  300 CONTINUE
      KTIME=KCODE(KV,2)
      IF(KTIME.EQ.0)GO TO 310
      DO 312 J=1,KSPCE
      DO 314 I=1,KTIME
      WCON(I)=ACON(KV,J,I)
  314 XCON(I)=BTIME(KV,I,2)
      CALL INTERP(WCON,XCON,VCON(J),TIME,KTIME,3)
  312 CONTINUE
      DO 315 I=1,KSPCE
  315 XCON(I)=BSPCE(KV,I)
  316 CONTINUE
      CALL INTERP(VCON,XCON,W,DIST,KSPCE,3)
      KTIME=KCODE(KV,3)
      IF(KTIME.EQ.0)GO TO 330
      DO 320 J=1,KSPCE
      DO 322 I=1,KTIME
```

```
      UCON(I)=CCCN(KV,J,I)
  322 XCON(I)=BTIME(KV,I,3)
      CALL INTERP(UCON,XCON,SCON(J),TIME,KTIME,3)
  320 CONTINUE
      DO 324 I=1,KSPCE
  324 XCON(I)=BSPCE(KV,I)
  326 CONTINUE
      CALL INTERP(SCON,XCON,Y,DIST,KSPCE,3)
      IF(Y.LT.0.)Y=0.
C     DETERMINE DERIVATIVE
      XZ1=ZBM(KV,KK-1,1)-ZB(KV,1,1)
      XR1=RBM(KV,KK-1,1)-RB(KV,1,1)
      IF((XZ1.EQ.0.).AND.(XR1.EQ.0.))GO TO 510
      DIST1=SQRT(XZ1**2+XR1**2)
      XI1=ATAN2(XZ1,XR1)
      GO TO 512
  510 CONTINUE
      DIST1=0.
      XI1=ETA
  512 CONTINUE
      XZ2=ZBM(KV,KK+1,1)-ZB(KV,1,1)
      XR2=RBM(KV,KK+1,1)-RB(KV,1,1)
      IF((XZ2.EQ.0.).AND.(XR2.EQ.0.))GO TO 520
      DIST2=SQRT(XZ2**2+XR2**2)
      XI2=ATAN2(XZ2,XR2)
      GO TO 522
  520 CONTINUE
      DIST2=0.
      XI2=ETA
  522 CONTINUE
      DIST1=DIST1*COS(XI1-ETA)
      DIST2=DIST2*COS(XI2-ETA)
      XZ3=ZBM(KV,KK,1)-ZB(KV,1,1)
      XR3=RBM(KV,KK,1)-RB(KV,1,1)
      XI3=ATAN2(XZ3,XR3)
      DIST3=SQRT(XR3**2+XZ3**2)
      DIST3=DIST3*COS(XI3-ETA)
      CALL INTERP(VCON,XCON,W2,DIST3,KSPCE,3)
      CALL INTERP(ZCON,XCON,X2,DIST3,KSPCE,3)
      CALL INTERP(VCON,XCON,W1,DIST1,KSPCE,3)
      CALL INTERP(VCON,XCON,W3,DIST2,KSPCE,3)
      CALL INTERP(ZCON,XCON,X1,DIST1,KSPCE,3)
      CALL INTERP(ZCON,XCON,X3,DIST2,KSPCE,3)
      F(1)=X1
      F(2)=X2
      F(3)=X3
      XZ1=ZBM(KV,KK,1)-ZBM(KV,KK-1,1)
      XR1=RBM(KV,KK,1)-RBM(KV,KK-1,1)
      XZ2=ZBM(KV,KK+1,1)-ZBM(KV,KK-1,1)
      XR2=RBM(KV,KK+1,1)-RBM(KV,KK-1,1)
      XY(1)=0.
      XY(2)=SQRT(XZ1**2+XR1**2)
      XY(3)=SQRT(XZ2**2+XR2**2)
      CALL LAGRNE(1,2,3,F,XY,DZZ,DWR,DXX)
      RETURN
  100 DO 200 I=1,KSPCE
      ZCON(I)=BCCN(KV,I,1)
  200 XCON(I)=BSPCE(KV,I)
```

```
      CALL INTERP(ZCON,XCON,DMY,DIST,KSPCE,3)
      GO TO(210,212,212,212,212,212,214),NK
210   Y=DMY
      RETURN
212   Z=DMY
      WW=THETA(KV,KK,1)
      RETURN
 214  X=DMY
      GO TO 300
310   CONTINUE
      DO 400 I=1,KSPCE
      VCON(I)=ACCN(KV,I,1)
400   XCON(I)=BSPCE(KV,I)
      GO TO 316
330   CONTINUE
      DO 410 I=1,KSPCE
      SCON(I)=CCCN(KV,I,1)
410   XCON(I)=BSPCE(KV,I)
      GO TO 326
      END
```

```
      SUBROUTINE FINITE(L,M,N,PROP,X,PPL,PPM,PPN)
      DIMENSION PROP(1),X(1),F(3),DELF(3),DDF(1),Y(3)
C     SET UP DIFFERENCE TABLE
      F(1)=PROP(L)
      F(2)=PROP(M)
      F(3)=PROP(N)
      Y(1)=X(L)
      Y(2)=X(M)
      Y(3)=X(N)
      DELF(1)=F(2)-F(1)
      DELF(2)=F(3)-F(2)
      DDF(1)=DELF(2)-DELF(1)
      H=Y(2)-Y(1)
C     FORWARD DIFFERENTIATION
      PPL=(DELF(1)-.5*DDF(1))/H
C     CENTRAL DIFFERENTIATION
      PPM=(DELF(1)+DELF(2))/(2.*H)
C     BACKWARD DIFF.
      PPN=(DELF(2)+.5*DDF(1))/H
      RETURN
      END
```

```
      FUNCTION FLAGRE(F,Y,YIND)
      DIMENSION F(1),Y(1)
C     THIS FUNCTION COMPUTES THE LAGRANGE DERIVATIVE
      A=Y(1)-Y(2)
      B=Y(1)-Y(3)
      C=Y(2)-Y(3)
      FLAGRE=(2.*YIND-Y(2)-Y(3))*F(1)/(A*B)-
     .(2.*YIND-Y(1)-Y(3))*F(2)/(A*C)+
     .(2.*YIND-Y(1)-Y(2))*F(3)/(B*C)
      RETURN
      END
```

```
      SUBROUTINE GENCOR(I,J,CONST,KAXIS,TIME,DELTA,KPRINT)
      REALKAXIS
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/GENPT/IJ,IK,I1,I2,J1,J2,NGEN(9),XY(3)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
C     THIS SUBROUTINE COMPUTES A GENERAL POINT
      INDEX=0
      ITEST=1
C     CHECK IF POINT IS ACTIVE OR ODD
      IF(NACTVE(I,J).EQ.0) GO TO 1310
      IF(NODD(I,J).EQ.1) GO TO 1310
      ANGLE=0.
      IJ=I
      IK=J
      I1=I-1
      I2=I+1
      J1=J-1
      J2=J+1
      RNEW=RM(I,J)
      ZNEW=ZM(I,J)
      CALL GENERAL(RNEW,ZNEW,IB)
      CALL ACTVTY(KVALUE,NOACT)
      IF(NOACT.GT.1)NODD(I,J)=1
      IF(NODD(I,J).EQ.1) GO TO 1310
      NK=NOACT+1
      GO TO(100,110),NK
  100 CONTINUE
      IB=1
      CALL DIFFER
      GO TO 200
  110 CONTINUE
C     CONSIDER ABNORMAL GENERAL POINT
      CALL ABGEN(RNEW,ZNEW,IB,KVALUE,1,ETA)
      CALL PREPRS(1,IB,KVALUE,ETA,BETA,IDUM)
  200 CONTINUE
C     GUESS AT PROPERTIES
      CALL GUESS(0,I,J)
 1320 CONTINUE
      CALL PHYCHR(0,DELTA)
C     INTERPOLATE FOR PROPERTIES
      CALL INPROP(IB,0)
C     INTEGRATE COMPATIBILITY EQUATIONS
      NCOUNT=1
      CALL COMPEQ(0,TIME,DELTA,KAXIS,CONST,NCOUNT,KPRINT)
      INDEX=INDEX+1
      IF(INDEX.LT.25) GO TO 1330
      WRITE(6,8700)I,J
      CALL PRINT
      CALL EXIT
 1330 IF(NCOUNT.EQ.1) GO TO 1320
      ITEST=1
```

```
C       STORE PROPERTIES IN NEW TIME PLANE
        P(I,J,2)=PNEW
        RHO(I,J,2)=RHNEW
        UR(I,J,2)=URNEW
        UZ(I,J,2)=UZNEW
        A(I,J,2)=SOUND(RHNEW,PNEW)
 1310 CONTINUE
        RETURN
 8700 FORMAT(1H ,24HPROGRAM LOOPING AT POINT,I2,1H,,I2)
        END
```

```
      SUBROUTINE GENERAL(RNEW,ZNEW,IB)
C     THIS SUBROUTINE SETS UP THE DOMAIN POINTS FOR A GENERAL POINT
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25)
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/GENPT/IJ,IK,I1,I2,J1,J2,NGEN(9),XY(3)
      RPD(1)=RM(I1,J2)
      ZPD(1)=ZM(I1,J2)
      PD(1)=P(I1,J2,1)
      AD(1)=A(I1,J2,1)
      RHOD(1)=RHO(I1,J2,1)
      URD(1)=UR(I1,J2,1)
      UZD(1)=UZ(I1,J2,1)
      RPD(2)=RM(IJ,J2)
      ZPD(2)=ZM(IJ,J2)
      PD(2)=P(IJ,J2,1)
      AD(2)=A(IJ,J2,1)
      RHOD(2)=RHO(IJ,J2,1)
      URD(2)=UR(IJ,J2,1)
      UZD(2)=UZ(IJ,J2,1)
      RPD(3)=RM(I2,J2)
      ZPD(3)=ZM(I2,J2)
      PD(3)=P(I2,J2,1)
      AD(3)=A(I2,J2,1)
      RHOD(3)=RHO(I2,J2,1)
      URD(3)=UR(I2,J2,1)
      UZD(3)=UZ(I2,J2,1)
      RPD(4)=RM(I1,IK)
      ZPD(4)=ZM(I1,IK)
      PD(4)=P(I1,IK,1)
      AD(4)=A(I1,IK,1)
      RHOD(4)=RHO(I1,IK,1)
      URD(4)=UR(I1,IK,1)
      UZD(4)=UZ(I1,IK,1)
      RPD(5)=RM(IJ,IK)
      ZPD(5)=ZM(IJ,IK)
      PD(5)=P(IJ,IK,1)
      AD(5)=A(IJ,IK,1)
      RHOD(5)=RHO(IJ,IK,1)
      URD(5)=UR(IJ,IK,1)
      UZD(5)=UZ(IJ,IK,1)
      RPD(6)=RM(I2,IK)
      ZPD(6)=ZM(I2,IK)
      PD(6)=P(I2,IK,1)
      AD(6)=A(I2,IK,1)
      RHOD(6)=RHO(I2,IK,1)
      URD(6)=UR(I2,IK,1)
      UZD(6)=UZ(I2,IK,1)
      RPD(7)=RM(I1,J1)
      ZPD(7)=ZM(I1,J1)
      PD(7)=P(I1,J1,1)
      AD(7)=A(I1,J1,1)
      RHOD(7)=RHO(I1,J1,1)
```

```
URD(7)=UR(I1,J1,1)
UZD(7)=UZ(I1,J1,1)
RPD(8)=RM(IJ,J1)
ZPD(8)=ZM(IJ,J1)
PD(8)=P(IJ,J1,1)
AD(8)=A(IJ,J1,1)
RHOD(8)=RHC(IJ,J1,1)
URD(8)=UR(IJ,J1,1)
UZD(8)=UZ(IJ,J1,1)
RPD(9)=RM(I2,J1)
ZPD(9)=ZM(I2,J1)
PD(9)=P(I2,J1,1)
AD(9)=A(I2,J1,1)
RHOD(9)=RHC(I2,J1,1)
URD(9)=UR(I2,J1,1)
UZD(9)=UZ(I2,J1,1)
NGEN(1)=NACTVE(I1,J2)
NGEN(2)=NACTVE(IJ,J2)
NGEN(3)=NACTVE(I2,J2)
NGEN(4)=NACTVE(I1,IK)
NGEN(5)=NACTVE(IJ,IK)
NGEN(6)=NACTVE(I2,IK)
NGEN(7)=NACTVE(I1,J1)
NGEN(8)=NACTVE(IJ,J1)
NGEN(9)=NACTVE(I2,J1)
IB=1
LSUB(1)=7
LSUB(2)=4
LSUB(3)=1
LSUB(4)=8
LSUB(6)=2
LSUB(5)=5
LSUB(7)=9
LSUB(8)=6
LSUB(9)=3
RETURN
END
```

```
      FUNCTION GRS (X,NDX,Y,NDY,XV,N,NRANGE)
C     THIS ROUTINE PERFORMS PARABOLIC INTERPOLATION
      DIMENSION X(NDX,3),Y(NDY,3),DX(3),DY(3),YP(2)
      DATA DX/3*0./,DY/3*0./,YP/2*0./
      NRANGE=0
      IF(XV.LT.X(1,1))NRANGE=-1
      IF(XV.GT.X(1,N))NRANGE=+1
      DO10I=1,N
      IF(XV-X(1,I))20,70,10
   10 CONTINUE
      I=N
   20 IF(I.GT.2)GO TO 30
      N1=3
      N2=2
      N3=1
      NP=3
      GOTO55
   30 IF(I.LT.N  )GO TO 40
      NP=3
      GOTO50
   40 NP=4
      N4=I+1
   50 N1=I-2
      N2=I-1
      N3=I
   55 DX(1)=X(1,N2)-X(1,N1)
                                  DY(1)=Y(1,N2)-Y(1,N1)
      DX(2)=X(1,N3)-X(1,N2)
                                  DY(2)=Y(1,N3)-Y(1,N2)
                                                          R=(XV-X(1,N2))/DX(
      YP(1)=(DY(1)*DX(2)**2+DY(2)*DX(1)**2)/(DX(1)*(DX(1)+DX(2)))
      IF(NP.EQ.4)GO TO 60
      GRS=Y(1,N2)+R*(YP(1)+R*(DY(2)-YP(1)))
      RETURN
   60 DX(3)=X(1,N4)-X(1,N3)
                                  DY(3)=Y(1,N4)-Y(1,N3)
      YP(2)=(DY(2)*DX(3)**2+DY(3)*DX(2)**2)/(DX(3)*(DX(2)+DX(3)))
      GRS=Y(1,N2)+R*(YP(1)+R*(3.*DY(2)-2.*YP(1)-YP(2)+R*(YP(1)+YP(2)-2
     1 DY(2))))
      RETURN
   70 GRS=Y(1,I)
      RETURN
      END
```

```
      FUNCTION GR1(T,X,N)
C     LINEAR INTERPOLATION
      DIMENSION T(2)
      DO 1 I=2,N
      K=I+I
      IF(X-T(K-1))2,2,1
    1 CONTINUE
    2 GR1=T(K-2)+(T(K)-T(K-2))/(T(K-1)-T(K-3))*(X-T(K-3))
      RETURN
      END
```

```
      SUBROUTINE GUESS(IBOUND,I,J)
C     THIS SUBROUTINE MAKES AN INITIAL GUESS FOR THE PROPERTIES
C     AT THE NEW POINT
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      NK=IBOUND+1
      GO TO(100,110,110,110),NK
  100 PNEW=P(I,J,1)
      RHNEW=RHO(I,J,1)
      ANEW=A(I,J,1)
      URNEW=UR(I,J,1)
      UZNEW=UZ(I,J,1)
      GO TO 120
  110 PNEW=PB(I,J,1)
      RHNEW=RHOB(I,J,1)
      ANEW=AB(I,J,1)
      URNEW=URB(I,J,1)*COS(ANGLE)+UZB(I,J,1)*SIN(ANGLE)
      UZNEW=-URB(I,J,1)*SIN(ANGLE)+UZB(I,J,1)*COS(ANGLE)
  120 CONTINUE
C     SET BASE POINTS OF CONE
      DO 840 KK=1,5
      URG(KK)=URNEW
      UZG(KK)=UZNEW
      PG(KK)=PNEW
      RHOG(KK)=RHNEW
      AG(KK)=ANEW
  840 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INITIAL(DELR,DELZ,TIME)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NCR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/EXT/URR,UZZ,PRAT,RRAT,ARAT
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
C     FOR BLAST WAVE GENERATE INITIAL DATA
      DO 520 I=1,NOZ
      DO 520 J=1,NOR
      RMA=RM(I,J)
      ZMA=ZM(I,J)
      CALL EXACT(RMA,ZMA,TIME)
      UR(I,J,1)=URR
      UZ(I,J,1)=UZZ
      P(I,J,1)=PRAT
      RHO(I,J,1)=RRAT
      A(I,J,1)=ARAT
  520 CONTINUE
      PRINT 500
  500 FORMAT(' INITIAL DATA GENERATED BY SEDOV',1H',
     .'S EXACT SOLUTION')
      DO 522 I=1,NBP
      KSPCE=LBPT(I)
      DO 522 K=1,KSPCE
      RMA=RBM(I,K,1)
      ZMA=ZBM(I,K,1)
      CALL EXACT(RMA,ZMA,TIME)
      URB(I,K,1)=URR
      UZB(I,K,1)=UZZ
      PB(I,K,1)=PRAT
      RHOB(I,K,1)=RRAT
      AB(I,K,1)=ARAT
  522 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INPROP(IB,IBOUND)
      DIMENSION PROP(10),X(10),P1(3),RHO1(3),UZ1(3),UR1(3),
     .A1(3),DUM1(3),DUM2(3),DUM3(3),DUM4(3),DUM5(3),DUM6(3),XY(3)
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5),PAR(5),PAZ(5)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),
     .DAR(15),DAZ(15)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
C     INTERPOLATE FOR PROPERTIES OF 4 BI-CHARACTERISTICS AND
C     PARTICLE PATH
      KV=3
      DO 100 I=1,5
      IF((IBOUND.GT.0).AND.(I.EQ.2))GO TO 100
      IF(((IBOUND.EQ.2).OR.(IBOUND.EQ.5)).AND.(I.EQ.3))GO TO 100
      IF(((IBOUND.EQ.3).OR.(IBOUND.EQ.6)).AND.(I.EQ.1))GO TO 100
      GO TO(102,104),IB
C     VERTICAL INTERSECTION
 102  XIND=RG(I)
      YIND=ZG(I)
      GO TO 106
C     HORIZONTAL INTERSECTION
 104  XIND=ZG(I)
      YIND=RG(I)
 106  CONTINUE
      DO 110 K=1,KV
      GO TO (202,204,206),K
 202  IK=LSUB(1)
      JK=LSUB(2)
      LK=LSUB(3)
      GO TO 116
 204  IK=LSUB(4)
      JK=LSUB(5)
      LK=LSUB(6)
      GO TO 116
 206  IK=LSUB(7)
      JK=LSUB(8)
      LK=LSUB(9)
 116  CALL PREPDO(IK,JK,LK,URD,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,UR1(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,UZD,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,UZ1(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,PD,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,P1(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,RHOD,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,RHO1(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,DURZ,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM1(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,DUZZ,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM2(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,DAZ,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM3(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,DURR,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM4(K),XIND,3,3)
      CALL PREPDO(IK,JK,LK,DUZR,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM5(K),XIND,3,3)
```

```
      CALL PREPDO(IK,JK,LK,CAR,IB,RPD,ZPD,PROP,X)
      CALL INTERP(PROP,X,DUM6(K),XIND,3,3)
  110 CONTINUE
C     PERFORM SECOND INTERPOLATION
      IA=LSUB(1)
      IC=LSUB(3)
      ID=LSUB(4)
      IE=LSUB(7)
      IF=LSUB(9)
      GO TO(230,232),IB
  230 XY(1)=ZPD(IA)+(ZPD(IC)-ZPD(IA))*(RG(I)-RPD(IA))/(RPD(IC)-RPD(IA))
      XY(2)=ZPD(ID)
      XY(3)=ZPD(IE)+(ZPD(IF)-ZPD(IE))*(RG(I)-RPD(IE))/(RPD(IF)-RPD(IE))
      GO TO 234
  232 XY(1)=RPD(IA)+(RPD(IC)-RPD(IA))*(ZG(I)-ZPD(IA))/(ZPD(IC)-ZPD(IA))
      XY(2)=RPD(ID)
      XY(3)=RPD(IE)+(RPD(IF)-RPD(IE))*(ZG(I)-ZPD(IE))/(ZPD(IF)-ZPD(IE))
  234 CONTINUE
      CALL INTERP(UR1,XY,URGA,YIND,3,3)
      CALL INTERP(UZ1,XY,UZGA,YIND,3,3)
      URG(I)=URGA*COS(ANGLE)+UZGA*SIN(ANGLE)
      UZG(I)=-URGA*SIN(ANGLE)+UZGA*COS(ANGLE)
      CALL INTERP(P1,XY,PG(I),YIND,3,3)
      CALL INTERP(RHO1,XY,RHOG(I),YIND,3,3)
      AG(I)=SOUND(RHOG(I),PG(I))
C     DETERMINE PARTIAL DERIVATIVES IN Z DIRECTION
      CALL INTERP(DUM1,XY,PURZ(I),YIND,3,3)
      CALL INTERP(DUM2,XY,PUZZ(I),YIND,3,3)
      CALL INTERP(DUM3,XY,PAZ(I),YIND,3,3)
      CALL INTERP(DUM4,XY,PURR(I),YIND,3,3)
      CALL INTERP(DUM5,XY,PUZR(I),YIND,3,3)
      CALL INTERP(DUM6,XY,PAR(I),YIND,3,3)
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTEG(I,J,DELTA)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
C     DETERMINE NEW POSITION OF BOUNDARY POINT
      LK=JCODE(I)
      GO TO(10,10,30,40,50,60,10),LK
C     LK=1,CONSTANT PRESSURE LINE
C     LK=2,NORMAL VELOCITY LINE
C     LK=3,FREE SURFACE
C     LK=4,CONTACT DISCONTINUITY
C     LK=5,MOVING SOLID BOUNDARY
C     LK=6,SHOCK FRONT
C     LK=7,CASCADE LINE
   10 RNEW=RBM(I,J,1)
      ZNEW=ZBM(I,J,1)
      RETURN
C     IN GLOBAL COORDINATES
   30 RNEW=RBM(I,J,1)+.5*(URNEW*COS(ANGLE)-UZNEW*SIN(ANGLE)+
     .URB(I,J,1))*DELTA
      ZNEW=ZBM(I,J,1)+.5*(URNEW*SIN(ANGLE)+UZNEW*COS(ANGLE)+
     .UZB(I,J,1))*DELTA
      RETURN
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE INTERP(PROP1,X,Y,XIND,NOV,NORD)
C     THIS SUBROUTINE INTERPOLATES USING LAGRANGIAN DIFFERENCES
      DIMENSION PROP1(1),X(1),T(4)
C     DETERMINE WHETHER TO USE FORWARD OR BACKWARD
      IF(NOV.LE.2)GO TO 10
      Y=GRS(X,1,PROP1,1,XIND,NOV,NRANGE)
      RETURN
   10 T(1)=X(1)
      T(3)=X(2)
      T(2)=PROP1(1)
      T(4)=PROP1(2)
      Y=GR1(T,XIND,2)
      RETURN
      END
```

```
      SUBROUTINE LAGRNE(L,M,N,PROP,X,PPL,PPM,PPN)
      DIMENSION PROP(1),X(1),Y(3),F(3)
C     THIS SUBROUTINE COMPUTES LAGRANGE DERIVATIVE
C     SET UP TABLE
      F(1)=PROP(L)
      F(2)=PROP(M)
      F(3)=PROP(N)
      Y(1)=X(L)
      Y(2)=X(M)
      Y(3)=X(N)
C     FORWARD VALUE
      PPL=FLAGRE(F,Y,Y(1))
C     CENTRAL VALUE
      PPM=FLAGRE(F,Y,Y(2))
C     BACKWARD VALUE
      PPN=FLAGRE(F,Y,Y(3))
      RETURN
      END
```

```
      SUBROUTINE ODDPT(INDEX,M)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      DIMENSION IMSH(200),JMSH(200),DIST(20),ISTRE(20),KSTRE(20),
     .NEWORD(20),IFNL(200),KFNL(200),KVAL(200),PROP(10),X(10)
      PI=3.1415926535898
C     DETERMINE BOUNDARY POINTS FOR ODD POINTS
      GO TO(100,300,500),INDEX
  100 KOUNT=0
      DO 110 I=1,NOZ
      DO 110 J=1,NOR
      IF(NACTVE(I,J).EQ.0)GO TO 110
      IF(NODD(I,J).EQ.0)GO TO 110
      INTER=1
      KOUNT=KOUNT+1
      IMSH(KOUNT)=I
      JMSH(KOUNT)=J
      NAMNT=0
      NUM=J
  112 CONTINUE
C     ASSUME THAT THE COLUMN J HAS A VERTICAL INTERSECTION
      DO 120 L=1,NBP
      KSPCE=LBPT(L)
      DO 120 K=1,KSPCE
      IF((K.EQ.1).OR.(K.EQ.KSPCE))GO TO 118
      IF(INTER.NE.ITYPE(L,K,M))GO TO 120
      IF(NUM.NE.INUM(L,K,M))GO TO 120
  118 CONTINUE
      NAMNT=NAMNT+1
      ISTRE(NAMNT)=L
      KSTRE(NAMNT)=K
      DIST(NAMNT)=SQRT((RM(I,J)-RBM(L,K,M))**2+
     .(ZM(I,J)-ZBM(L,K,M))**2)
  120 CONTINUE
      GO TO(122,124),INTER
C     ASSUME HORIZONTAL INTERSECTION
  122 INTER=2
      NUM=I
      GO TO 112
  124 IF(NAMNT.EQ.0)GO TO 126
C     FIND MINIMUM DISTANCE
      CALL SORT(DIST,NAMNT,NEWORD,0,-1)
C     STORE RESULTS
      NFINL=NEWORD(1)
      IFNL(KOUNT)=ISTRE(NFINL)
      KFNL(KOUNT)=KSTRE(NFINL)
      KVAL(KOUNT)=0
      L=ISTRE(NFINL)
      K=KSTRE(NFINL)
      KTYPE=ITYPE(L,K,M)
      IF((K.EQ.1).OR.(K.EQ.LBPT(L))) GO TO 140
      GO TO(128,130),KTYPE
  128 KVAL(KOUNT)=1
```

```
      IF((THETA(L,K,M).GT.(.75*PI)).AND.
     .(THETA(L,K,M).LE.(1.25*PI)))KVAL(KOUNT)=2
      GO TO 132
  130 IF((THETA(L,K,M).GT.(.25*PI)).AND.
     .(THETA(L,K,M).LE.(.75*PI)))KVAL(KOUNT)=3
      IF((THETA(L,K,M).GT.(1.25*PI)).AND.
     .(THETA(L,K,M).LE.(1.75*PI)))KVAL(KOUNT)=4
  132 IF(KVAL(KOUNT).NE.0)GO TO 110
  126 WRITE(6,8000)I,J
      CALL EXIT
  140 IF(K.EQ.1) GO TO 142
      KVAL(KOUNT)=6
      GO TO 110
  142 KVAL(KOUNT)=5
  110 CONTINUE
  300 CONTINUE
      IF(KOUNT.EQ.0)RETURN
C     INTERPOLATE ODD POINTS
      DO 310 I=1,KOUNT
      L=IMSH(I)
      K=JMSH(I)
      LB=IENL(I)
      KB=KENL(I)
      NK=KVAL(I)
      CALL PREPOD(NK,L,K,LB,KB,M,X,PROP,URB,UR,XIND)
      CALL INTERP(PROP,X,UR(L,K,M),XIND,3,3)
      CALL PREPOD(NK,L,K,LB,KB,M,X,PROP,UZB,UZ,XIND)
      CALL INTERP(PROP,X,UZ(L,K,M),XIND,3,3)
      CALL PREPOD(NK,L,K,LB,KB,M,X,PROP,RHOB,RHO,XIND)
      CALL INTERP(PROP,X,RHO(L,K,M),XIND,3,3)
      CALL PREPOD(NK,L,K,LB,KB,M,X,PROP,PB,P,XIND)
      CALL INTERP(PROP,X,P(L,K,M),XIND,3,3)
      A(L,K,M)=SOUND(RHO(L,K,M),P(L,K,M))
  310 CONTINUE
      RETURN
  500 CONTINUE
C     INTERPOLATE FOR ODD BOUNDARY POINTS
      DO 510 I=1,NBP
      KSPCE=LBOT(I)-1
      DO 520 K=2,KSPCE
      IF(NBACT(I,K,M).EQ.0)GO TO 520
      IF(NBODD(I,K,M).EQ.0)GO TO 520
      NART=1
      IF(K.EQ.KSPCE)NART=2
      GO TO(530,540),NART
  530 IK=K-1
      JK=K+1
      LK=K+2
      MK=K
      IF(NBACT(I,LK,M).EQ.0)GO TO 540
      GO TO 550
  540 IK=K-2
      JK=K-1
      LK=K+1
      MK=K
      IF(NBACT(I,IK,M).EQ.0)GO TO 530
  550 CONTINUE
C     INTERPOLATE FOR ODD BOUNDARY POINT
```

```
      CALL BDPROP(IK,JK,LK,MK,I,M,PROP,X,URB,XIND)
      CALL INTERP(PROP,X,URB(I,K,M),XIND,3,3)
      CALL BDPROP(IK,JK,LK,MK,I,M,PROP,X,UZB,XIND)
      CALL INTERP(PROP,X,UZB(I,K,M),XIND,3,3)
      CALL BDPROP(IK,JK,LK,MK,I,M,PROP,X,PB ,XIND)
      CALL INTERP(PROP,X,PB(I,K,M),XIND,3,3)
      CALL BDPROP(IK,JK,LK,MK,I,M,PROP,X,RHOB,XIND)
      CALL INTERP(PROP,X,RHOB(I,K,M),XIND,3,3)
      AB(I,K,M)=SOUND(RHOB(I,K,M),PB(I,K,M))
  520 CONTINUE
  510 CONTINUE
      RETURN
 8000 FORMAT(1H ,15HERROR SUB ODDPT,I3,1H,,I2)
 8010 FORMAT(1H ,2013)
 8020 FORMAT(1H ,10F10.3)
      END
```

```
      SUBROUTINE PHYCHR(IBOUND,DELT)
C     THIS SUBROUTINE PROJECTS THE PHYSICAL CHAR. AND
C     PARTICLE PATH BACK TO ORIGINAL TIME PLANE
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      PI=3.141592653589793/2.
      RAP=0.
      ZAP=0.
C     COMPUTE BI-CHARACTERISTICS
      DO 100 I=1,4
      AK=FLOAT(I)-1.
      IF(IBOUND.EQ.0)GO TO 110
      IF(I.EQ.2)GO TO 100
      IF(((IBOUND.EQ.2).OR.(IBOUND.EQ.5)).AND.(I.EQ.3))GO TO 100
      IF(((IBOUND.EQ.3).OR.(IBOUND.EQ.6)).AND.(I.EQ.1))GO TO 100
  110 RGA=RAP-.5*DELT*(URNEW+URG(I)+(ANEW+AG(I))*
     .COS(AK*PI))
      ZGA=ZAP-.5*DELT*(UZNEW+UZG(I)+(ANEW+AG(I))*
     .SIN(AK*PI))
      RG(I)=RGA*COS(ANGLE)-ZGA*SIN(ANGLE)+RNEW
      ZG(I)=RGA*SIN(ANGLE)+ZGA*COS(ANGLE)+ZNEW
  100 CONTINUE
C     COMPUTE PARTICLE PATH
      RGA=RAP-.5*DELT*(URNEW+URG(5))
      ZGA=ZAP-.5*DELT*(UZNEW+UZG(5))
      RG(5)=RGA*COS(ANGLE)-ZGA*SIN(ANGLE)+RNEW
      ZG(5)=RGA*SIN(ANGLE)+ZGA*COS(ANGLE)+ZNEW
      IF(IBOUND.LT.4) RETURN
      RG(5)=RNEW
      ZG(5)=ZNEW
      RETURN
      END
```

```
      SUBROUTINE PREPB(I1,I2,I3,I4,PROP,Y,PROPA,X,IART)
C     THIS SUBROUTINE PREPARES THE DOMAIN DATA FOR
C     INTERPOLATION
      DIMENSION PROP(1),Y(1),PROPA(1),X(1)
      X(1)=Y(I1)
      PROPA(1)=PROP(I1)
      X(2)=Y(I2)
      PROPA(2)=PROP(I2)
      X(3)=Y(I3)
      PROPA(3)=PROP(I3)
      IF(IART.GT.0)RETURN
      X(4)=Y(I4)
      PROPA(4)=PROP(I4)
      RETURN
      END
```

```
      SUBROUTINE PREPDO(I1,I2,I3,PROP,KOUNT,RPD,ZPD,PROP1,X)
C     THIS SUBROUTINE PREPARES THE DOMAIN DATA FOR INTERPOLATION
      DIMENSION PROP1(10),X(10),PROP(1),RPD(1),ZPD(1)
      GO TO(100,110),KOUNT
  100 X(1)=RPD(I1)
      PROP1(1)=PROP(I1)
      X(2)=RPD(I2)
      PROP1(2)=PROP(I2)
      X(3)=RPD(I3)
      PROP1(3)=PROP(I3)
      RETURN
  110 X(1)=ZPD(I1)
      PROP1(1)=PROP(I1)
      X(2)=ZPD(I2)
      PROP1(2)=PROP(I2)
      X(3)=ZPD(I3)
      PROP1(3)=PROP(I3)
      RETURN
      END
```

```
      SUBROUTINE PREPOD(NK,IR,IC,LB,KB,M,X,PROP,PROPB,PROPG,XIND)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .PB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      DIMENSION X(1),PROP(1),PROPG(25,25,2),PROPB(10,30,2)
C     THIS SUBROUTINE PREPARES DATA FOR ODD POINT INTERPOLATION
      GO TO(100,200,300,400,500,600),NK
C     NK=1,VERTICAL INTERSECTION AND ADD ROW INDEX
  100 CONTINUE
      X(1)=ZBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=ZM(IR+1,IC)
      PROP(2)=PROPG(IR+1,IC,M)
      X(3)=ZM(IR+2,IC)
      PROP(3)=PROPG(IR+2,IC,M)
      XIND=ZM(IR,IC)
      RETURN
C     NK=2,VERTICAL INTERSECTION AND SUBTRACT ROW INDEX
  200 CONTINUE
      X(1)=ZBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=ZM(IR-1,IC)
      PROP(2)=PROPG(IR-1,IC,M)
      X(3)=ZM(IR-2,IC)
      PROP(3)=PROPG(IR-2,IC,M)
      XIND=ZM(IR,IC)
      RETURN
C     NK=3,HORIZONTAL INTERSECTION AND SUBTRACT COLUMN INDEX
  300 CONTINUE
      X(1)=RBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=RM(IR,IC-1)
      PROP(2)=PROPG(IR,IC-1,M)
      X(3)=RM(IR,IC-2)
      PROP(3)=PROPG(IR,IC-2,M)
      XIND=RM(IR,IC)
      RETURN
C     NK=4,HORIZONTAL INTERSECTION AND ADD COLUMN INDEX
  400 CONTINUE
      X(1)=RBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=RM(IR,IC+1)
      PROP(2)=PROPG(IR,IC+1,M)
      X(3)=RM(IR,IC+2)
      PROP(3)=PROPG(IR,IC+2,M)
      XIND=RM(IR,IC)
      RETURN
C     NK=5,FIRST CORNER POINT INTERPOLATION
  500 CONTINUE
      NART=ITYPE(LB,KB,M)
      GO TO (510,520),NART
  510 X(1)=RBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=RBM(LB,KB+1,M)
      PROP(2)=PROPB(LB,KB+1,M)
```

```
      X(3)=RBM(LB,KB+2,M)
      PROP(3)=PROPB(LB,KB+2,M)
      XIND=RM(IR,IC)
      RETURN
  520 X(1)=ZBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=ZBM(LB,KB+1,M)
      PROP(2)=PROPB(LB,KB+1,M)
      X(3)=ZBM(LB,KB+2,M)
      PROP(3)=PROPB(LB,KB+2,M)
      XIND=ZM(IR,IC)
      RETURN
C     NK=6,SECOND CORNER POINT INTERPOLATION
  600 CONTINUE
      NART=ITYPE(LB,KB,M)
      GO TO(610,620),NART
  610 X(1)=RBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=RBM(LB,KB-1,M)
      PROP(2)=PROPB(LB,KB-1,M)
      X(3)=RBM(LB,KB-2,M)
      PROP(3)=PROPB(LB,KB-2,M)
      XIND=RM(IR,IC)
      RETURN
  620 X(1)=ZBM(LB,KB,M)
      PROP(1)=PROPB(LB,KB,M)
      X(2)=ZBM(LB,KB-1,M)
      PROP(2)=PROPB(LB,KB-1,M)
      X(3)=ZBM(LB,KB-2,M)
      PROP(3)=ZBM(LB,KB-2,M)
      XIND=ZM(IR,IC)
      RETURN
      END
```

```
      SUBROUTINE PREPRS(INDEX,IB,KVALUE,ETA,BETA,KNT)
      COMMON/BUFFER/PURR(15),PURZ(15),PUZR(15),PUZZ(15),
     .PAR(15),PAZ(15)
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),
     .DAR(15),DAZ(15)
C     THE PURPOSE OF THIS SUBROUTINE IS TO PROCESS THE
C     DERIVATIVES FOR THE ABNORMAL GENERAL POINT AND
C     THE CORNER POINT
      GO TO(100,300),INDEX
C     COMPUTE GENERAL POINT DERIVATIVES
  100 CONTINUE
      CALL ABDIF(IB,ETA,KVALUE)
      DO 110 KL=1,9
      DURR(KL)=PURR(KL)
      DURZ(KL)=PURZ(KL)
      DUZR(KL)=PUZR(KL)
      DUZZ(KL)=PUZZ(KL)
      DAR(KL)=PAR(KL)
      DAZ(KL)=PAZ(KL)
  110 CONTINUE
      RETURN
C     COMPUTE CORNER POINT DERIVATIVES
  300 CONTINUE
      GO TO(310,320),KNT
C     COMPUTE FIRST CORNER POINT
C     I BOUNDARY IS 7,4,1 AND K BOUNDARY IS 7,8,9
  310 CONTINUE
      CALL ABDIF(1,ETA,1)
      DO 312 KL=1,9
      DURR(KL)=PURR(KL)
      DURZ(KL)=PURZ(KL)
      DUZR(KL)=PUZR(KL)
      DUZZ(KL)=PUZZ(KL)
      DAR(KL)=PAR(KL)
      DAZ(KL)=PAZ(KL)
  312 CONTINUE
C     COMPLETE ADJOINING BOUNDARY
      CALL ABDIF(2,BETA,3)
      DO 314 KL=7,9
      DURR(KL)=PURR(KL)
      DURZ(KL)=PURZ(KL)
      DUZR(KL)=PUZR(KL)
      DUZZ(KL)=PUZZ(KL)
      DAZ(KL)=PAZ(KL)
      DAR(KL)=PAR(KL)
  314 CONTINUE
      RETURN
C     DO SECOND CORNER POINT
  320 CONTINUE
      CALL ABDIF(1,ETA,1)
      DO 322 KL=1,9
      DURR(KL)=PURR(KL)
      DURZ(KL)=PURZ(KL)
      DUZR(KL)=PUZR(KL)
      DUZZ(KL)=PUZZ(KL)
      DAR(KL)=PAR(KL)
      DAZ(KL)=PAZ(KL)
  322 CONTINUE
```

```
C       COMPLETE ADJOINING BOUNDARY
        CALL ABDIF(2,BETA,1)
        DO 324 KL=1,3
        DURR(KL)=PURR(KL)
        DURZ(KL)=PURZ(KL)
        DUZR(KL)=PUZR(KL)
        DUZZ(KL)=PUZZ(KL)
        DAR(KL)=PAR(KL)
        DAZ(KL)=PAZ(KL)
324     CONTINUE
        RETURN
        END
```

```
      SUBROUTINE PRINT
C     WRITE OUT INTERMEDIATE CUTPUT
      COMMON/PLANE/URG(5),UZG(5),PG(5),RHOG(5),RG(5),ZG(5),AG(5),
     .PNEW,RHNEW,ANEW,URNEW,UZNEW,RAP,ZAP,RNEW,ZNEW
      COMMON/SETUP/RAS(4),RAC(4),R(5),DELP,DELRH,DELUR,
     .DELUZ,S(5)
      COMMON/PARTIAL/PURR(5),PURZ(5),PUZR(5),PUZZ(5),PAR(5),PAZ(5)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/DERIV/DURR(15),DURZ(15),DUZR(15),DUZZ(15),DAR(15),DAZ(15)
      WRITE(6,8010)
      WRITE(6,8000)DELP,DELRH,DELUR,DELUZ
      WRITE(6,8000)(RAC(I),I=1,4)
      WRITE(6,8000)(RAS(I),I=1,4)
      WRITE(6,8000)(R(I),I=1,5)
      WRITE(6,8000)(S(I),I=1,5)
      DO 151 I=1,5
  151 WRITE(6,8000)RG(I),ZG(I),PG(I),RHOG(I),AG(I),URG(I),UZG(I)
      DO 153 I=1,5
  153 WRITE(6,8000)PURR(I),PURZ(I),PUZR(I),PUZZ(I),PAR(I),PAZ(I)
      DO 160 I=1,12
  160 WRITE(6,8000)RPD(I),ZPD(I),PD(I),RHOD(I),AD(I),URD(I),UZD(I)
      DO 162 I=1,12
  162 WRITE(6,8000)DURR(I),DURZ(I),DUZR(I),DUZZ(I),DAR(I),DAZ(I)
      WRITE(6,8020)PNEW,RHNEW,URNEW,UZNEW,ANEW,RNEW,ZNEW
 8000 FORMAT(1H ,10E10.3)
 8010 FORMAT(10H SUB PRINT)
 8020 FORMAT(1H ,7E14.7)
      RETURN
      END
```

```
      SUBROUTINE REFLCT(M)
C     THIS ROUTINE SETS NEW DOMAIN POINTS FOR CORNER POINT
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/XCHNG/IS,KS,KS1,KS2,ID,ID1,ID2
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),
     .RHOD(15),URD(15),UZD(15),LSUB(15)
      RPD(ID)=RBM(IS,KS,M)
      ZPD(ID)=ZBM(IS,KS,M)
      PD(ID)=PB(IS,KS,M)
      RHOD(ID)=RHOB(IS,KS,M)
      URD(ID)=URB(IS,KS,M)
      UZD(ID)=UZB(IS,KS,M)
      AD(ID)=AB(IS,KS,M)
      RPD(ID1)=RBM(IS,KS1,M)
      ZPD(ID1)=ZBM(IS,KS1,M)
      PD(ID1)=PB(IS,KS1,M)
      RHOD(ID1)=RHOB(IS,KS1,M)
      URD(ID1)=URB(IS,KS1,M)
      UZD(ID1)=UZB(IS,KS1,M)
      AD(ID1)=AB(IS,KS1,M)
      RPD(ID2)=RBM(IS,KS2,M)
      ZPD(ID2)=ZBM(IS,KS2,M)
      PD(ID2)=PB(IS,KS2,M)
      RHOD(ID2)=RHOB(IS,KS2,M)
      URD(ID2)=URB(IS,KS2,M)
      UZD(ID2)=UZB(IS,KS2,M)
      AD(ID2)=AB(IS,KS2,M)
      RETURN
      END
```

```
      SUBROUTINE RPRIME(DIST,RPD,ZPD,PURR,PUZR,PAR,
     .URD,UZD,AD)
      DIMENSION DIST(1),RPD(1),ZPD(1),PURR(1),PUZR(1),PAR(1),URD(1),
     .UZD(1),AD(1)
C     THIS SUBROUTINE COMPUTES THE R PRIME DERIVATIVES
      DIST(1)=0.
      DIST(2)=SQRT((RPD(2)-RPD(1))**2+(ZPD(2)-ZPD(1))**2)
      DIST(3)=SQRT((RPD(3)-RPD(1))**2+(ZPD(3)-ZPD(1))**2)
      CALL LAGRNE(1,2,3,URD,DIST,PURR(1),PURR(2),PURR(3))
      CALL LAGRNE(1,2,3,UZD,DIST,PUZR(1),PUZR(2),PUZR(3))
      CALL LAGRNE(1,2,3,AD,DIST,PAR(1),PAR(2),PAR(3))
      RETURN
      END
```

```
      SUBROUTINE SCAN(ZVAR,ZA,UZA,AA,UZVAR,AVAR,DELTA,
     .DMIN)
C     COMPUTE TIME INCREMENT
      XNUMER=(ZVAR-ZA)/(UZA+AA)
      DENON=1.-(UZVAR-AVAR)/(UZA+AA)
      DELTIM=XNUMER/DENON
      DMIN1=DELTIM*(UZA+AA)
      IF(DELTIM.GE.DELTA)RETURN
      DELTA=DELTIM
      DMIN=DMIN1
      RETURN
      END
```

```
      SUBROUTINE SEARCH(DELR,DELZ,RINIT,ZINIT,M)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/BOUND/JCODE(10),KCODE(10,3),LCODE(10),MCODE(10),
     .BTIME(10,20,3),BCON(10,20,20),BSPCE(10,20),ACON(10,20,20),
     .CCON(10,20,20)
      COMMON/DATA/ITEST,NBP,WIDTH,ANGLE
      PI=3.1415926535898
C     DETERMINE IF GENERAL POINT IS ACTIVE AND/OR ODD
C     DETERMINE BOUNDARY ANGLE
      VALUE=AMAX1(DELR,DELZ)*.001
      DMEAS=AMIN1(DELR,DELZ)*.9
      DO 10C I=1,NOZ
      DO 100 J=1,NOR
      NACTVE(I,J)=1
      NODD(I,J)=0
  100 CONTINUE
      DO 420 I=1,NOZ
      DO 420 J=1,NOR
      RM(I,J)=RINIT+DELR*FLOAT(J-1)
      ZM(I,J)=ZINIT+DELZ*FLOAT(I-1)
      DO 420 K=1,NBP
      KP=LCODE(K)
      KSPCE=LCODE(K)-1
      DO 432 L=1,KSPCE
      ANGLE=ATAN2((ZB(K,L+1,M)-ZB(K,L,M)),
     .(RB(K,L+1,M)-RB(K,L,M)))
      ZPR=-(RM(I,J)-RB(K,L,M))*SIN(ANGLE)+
     .(ZM(I,J)-ZB(K,L,M))*COS(ANGLE)
      RPR=(RM(I,J)-RB(K,L,M))*COS(ANGLE)+
     .(ZM(I,J)-ZB(K,L,M))*SIN(ANGLE)
      PPR=SQRT((ZPR**2)+(RPR**2))
      IF(ZPR.GE.(-VALUE)) GO TO 422
      NACTVE(I,J)=0
      GO TO 432
  422 IF(ZPR.LT.DMEAS)NODD(I,J)=1
  432 CONTINUE
  420 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SET(ANGLE,NUMBR,NCODE,LCODE,IC1,IC2)
C     THIS SUBROUTINE SETS THE ROW OR COLUMN NUMBERS
C     NCODE=1,VERTICAL INTERSECTION,NCODE=2,HORIZONTAL
C     INTERSECTION
      PI=3.141592653589
      GO TO(100,200),NCODE
  100 GO TO(110,130),LCODE
C     LCODE=1,POSITIVE R INTERSECTION
C     LCODE=2,NEGATIVE R INTERSECTION
  110 IF((ANGLE.GT.PI).AND.(ANGLE.LE.1.25*PI))GO TO 112
      IC1=NUMBR-1
      IC2=NUMBR+1
      RETURN
  112 IC1=NUMBR+1
      IC2=NUMBR-1
      RETURN
C     NEGATIVE R INTERSECTION
  130 CONTINUE
      IF((ANGLE.GT..75*PI).AND.(ANGLE.LE.PI))GO TO 132
      IC1=NUMBR-1
      IC2=NUMBR+1
      RETURN
  132 IC1=NUMBR+1
      IC2=NUMBR-1
      RETURN
C     HORIZONTAL INTERSECTION
  200 GO TO(210,230),LCODE
C     LCODE=1,POSITIVE R INTERSECTION
C     LCODE=2,NEGATIVE R INTERSECTION
  210 IF((ANGLE.GT.1.5*PI).AND.(ANGLE.LE.1.75*PI))GO TO 212
      IC1=NUMBR-1
      IC2=NUMBR+1
      RETURN
  212 IC1=NUMBR+1
      IC2=NUMBR-1
      RETURN
C     NEGATIVE R INTERSECTION
  230 CONTINUE
      IF((ANGLE.GT.1.25*PI).AND.(ANGLE.LE.1.5*PI))GO TO 232
      IC1=NUMBR-1
      IC2=NUMBR+1
      RETURN
  232 IC1=NUMBR+1
      IC2=NUMBR-1
      RETURN
      END
```

```
      SUBROUTINE SOLCT(I,K,M,J,IR1,IR2,IPT1,IPT2,KPT1,KPT2,
     .PHI,BETA,KTYPE,LTYPE)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      DATA PI/3.1415926/
C     THE PURPOSE OF THIS ROUTINE IS TO SELECT THE PROPER
C     ROW OR COLUMN FOR THE CORNER POINT
      GO TO(100,300),KTYPE
  100 CONTINUE
C     VERTICAL INTERSECTION,FIND ROW
C     IART=1,ROWS INCREASE,IART=2,ROWS DECREASE
      IART=1
      IF((PHI.GT..75*PI).AND.(PHI.LE.1.25*PI))IART=2
      DO 110 IL=1,NOZ
      JART=NOZ-IL+1
      GO TO(112,114),IART
  112 IF(ZM(IL,IPT1).GT.ZBM(I,J,M))GO TO 116
      GO TO 110
  114 IF(ZM(JART,IPT1).LT.ZBM(I,J,M))GO TO 116
  110 CONTINUE
C     ERROR IF DROP THROUGH
      WRITE(6,8000)I,K
      CALL EXIT
  116 GO TO(122,124),IART
  122 IR1=IL
      IR2=IL+1
      GO TO 500
  124 IR1=JART
      IR2=JART-1
      GO TO 500
  300 CONTINUE
C     HORIZONTAL INTERSECTION,FIND COLUMN
C     IART=1,COLUMNS INCREASE,IART=2,COLUMNS DECREASE
      IART=1
      IF((PHI.GT.1.25*PI).AND.(PHI.LE.1.75*PI))IART=2
      DO 310 IL=1,NOR
      JART=NOR-IL+1
      GO TO(312,314),IART
  312 IF(RM(IPT1,IL).GT.RBM(I,J,M))GO TO 316
      GO TO 310
  314 IF(RM(IPT1,JART).LT.RBM(I,J,M))GO TO 316
  310 CONTINUE
C     ERROR IF DROP THROUGH
      WRITE(6,8000)I,K
      CALL EXIT
  316 GO TO(322,324),IART
  322 IR1=IL
      IR2=IL+1
      GO TO 500
  324 IR1=JART
      IR2=JART-1
  500 CONTINUE
```

```
C     LOCATE THE BOUNDARY POINT
      KSPCE=LBPT(K)
      DO 510 IL=1,KSPCE
      LTYPE=ITYPE(K,IL,M)
      JART=INUM(K,IL,M)
      IF(JART.NE.IR1)GO TO 510
      IF(KTYPE.NE.LTYPE)GO TO 520
  510 CONTINUE
      WRITE(6,8010)
      CALL EXIT
  520 BETA=THETA(K,IL,M)
C     ASSUME BOUNDARY ANGLE GT 0 AND LE PI
      KART=1
      IF((BETA.GE.PI).AND.(BETA.LT.2.*PI))KART=2
      GO TO(522,524),IART
  522 GO TO(532,534),KART
  532 KPT1=IL
      KPT2=IL+1
      RETURN
  534 KPT1=IL
      KPT2=IL-1
      RETURN
  524 GO TO(534,532),KART
 8000 FORMAT(1H ,18HERROR IN SUB SOLCT,
     .32HCANNOT FIND PROPER ROW OR COLUMN)
 8010 FORMAT(1H ,18HERROR IN SUB SOLCT,
     .49HMUST DEVELOP CAPABILITY IN PROGRAM FOR SAME TYPE ,
     .12HINTERSECTION)
      END
```

```
      SUBROUTINE SORT(X,KK,JKL,LABEL,ISTYPE)
C     THIS ROUTINE SORTS DATA IN ASCENDING ORDER OR DESCENDING
C     ORDER
      DIMENSION X(1),JKL(1),IX(1)
      EQUIVALENCE(ZZ,IZZ),(Z,IZ)
      LOGICAL ASCEND,FLTSWCH,FIXSWCH,ALFSWCH
      DATA FLTSWCH,FIXSWCH,ASCEND/.FALSE.,.FALSE.,.FALSE./
      IF(LABEL.EQ.0)ASCEND=.TRUE.
      K=1
      N=KK
      DO 1 I=1,N
    1 JKL(I)=I
      IF(ISTYPE)110,120,130
  120 STOP
  130 FIXSWCH=.TRUE.
  110 FLTSWCH=.NOT.FIXSWCH
   10 CONTINUE
      KLM=IABS(MOD(IZZ,(N-K+1))+K)
      Z=X(KLM)
  200 FORMAT(1HA,A9)
      DO 100 I=K,N
      ZZ = X(I)
      IF(FIXSWCH) GO TO 60
      IF(ASCEND) GO TO 45
      IF(ZZ.LE.Z)GO TO 100
      GO TO 20
   45 IF(ZZ.GE.Z)GO TO 100
      GO TO 20
   60 IF(ASCEND) GO TO 65
      IF(IZZ  .LE.IZ) GO TO 100
      GO TO 20
   65 IF(IZZ  .GE.IZ) GO TO 100
   20 Z=X(I)
      KLM=I
  100 CONTINUE
      Y=X(K)
      X(K)=X(KLM)
      X(KLM)=Y
      KMN=JKL(K)
      JKL(K)=JKL(KLM)
      JKL(KLM)=KMN
      K=K+1
      IF(K.LT.N)GO TO 10
      RETURN
      END
```

```
FUNCTION SOUND(RHONEW,PNEW)
COMMON/STATE/PINIT,RHOINT,AINIT
SOUND=PNEW/RHONEW
SOUND=SQRT(SOUND)
RETURN
END
```

```
      SUBROUTINE TRNSFR(NCODE,M,JK)
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .PB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/EXCHNG/ID,IE,IF,IC1,IC2,NUMBR
      GO TO(100,200),NCODE
C     SET UP BOUNDARY DOMAIN  --  VERTICAL INTERSECTION
  100 RPD(ID)=RM(JK,IC1)
      ZPD(ID)=ZM(JK,IC1)
      PD(ID)=P(JK,IC1,M)
      RHOD(ID)=RHO(JK,IC1,M)
      URD(ID)=UR(JK,IC1,M)
      UZD(ID)=UZ(JK,IC1,M)
      AD(ID)=A(JK,IC1,M)
      RPD(IE)=RM(JK,NUMBR)
      ZPD(IE)=ZM(JK,NUMBR)
      PD(IE)=P(JK,NUMBR,M)
      RHOD(IE)=RHO(JK,NUMBR,M)
      URD(IE)=UR(JK,NUMBR,M)
      UZD(IE)=UZ(JK,NUMBR,M)
      AD(IE)=A(JK,NUMBR,M)
      RPD(IF)=RM(JK,IC2)
      ZPD(IF)=ZM(JK,IC2)
      PD(IF)=P(JK,IC2,M)
      RHOD(IF)=RHO(JK,IC2,M)
      URD(IF)=UR(JK,IC2,M)
      UZD(IF)=UZ(JK,IC2,M)
      AD(IF)=A(JK,IC2,M)
      RETURN
C     SET UP BOUNDARY DOMAIN  --  HORIZONTAL INTERSECTION
  200 RPD(ID)=RM(IC1,JK)
      ZPD(ID)=ZM(IC1,JK)
      PD(ID)=P(IC1,JK,M)
      RHOD(ID)=RHO(IC1,JK,M)
      URD(ID)=UR(IC1,JK,M)
      UZD(ID)=UZ(IC1,JK,M)
      AD(ID)=A(IC1,JK,M)
      RPD(IE)=RM(NUMBR,JK)
      ZPD(IE)=ZM(NUMBR,JK)
      PD(IE)=P(NUMBR,JK,M)
      RHOD(IE)=RHO(NUMBR,JK,M)
      URD(IE)=UR(NUMBR,JK,M)
      UZD(IE)=UZ(NUMBR,JK,M)
      AD(IE)=A(NUMBR,JK,M)
      RPD(IF)=RM(IC2,JK)
      ZPD(IF)=ZM(IC2,JK)
      PD(IF)=P(IC2,JK,M)
      RHOD(IF)=RHO(IC2,JK,M)
      URD(IF)=UR(IC2,JK,M)
      UZD(IF)=UZ(IC2,JK,M)
      AD(IF)=A(IC2,JK,M)
      RETURN
      END
```

```
      SUBROUTINE TRSCNR(M,KTYPE)
C     THIS ROUTINE SETS UP DOMAIN POINTS FOR CORNER POINT
      COMMON RM(25,25),ZM(25,25),P(25,25,2),A(25,25,2),RHO(25,25,2),
     .UR(25,25,2),UZ(25,25,2),NOR,NOZ,RBM(10,30,2),ZBM(10,30,2),
     .RB(10,21,2),ZB(10,21,2),THETA(10,30,2),PB(10,30,2),AB(10,30,2),
     .RHOB(10,30,2),URB(10,30,2),UZB(10,30,2),NACTVE(25,25),NODD(25,25),
     .NBACT(10,30,2),NBODD(10,30,2),ITYPE(10,30,2),INUM(10,30,2),
     .LBPT(10),IAXES(10,30)
      COMMON/DOMAIN/RPD(15),ZPD(15),PD(15),AD(15),RHOD(15),
     .URD(15),UZD(15),LSUB(15)
      COMMON/YCHNG/KB,KP,IR,IC1,IC2,ID,IE,IF
      GO TO(100,300),KTYPE
  100 RPD(ID)=RM(IR,IC1)
      ZPD(ID)=ZM(IR,IC1)
      PD(ID)=P(IR,IC1,M)
      RHOD(ID)=RHO(IR,IC1,M)
      URD(ID)=UR(IR,IC1,M)
      UZD(ID)=UZ(IR,IC1,M)
      AD(ID)=A(IR,IC1,M)
      RPD(IE)=RM(IR,IC2)
      ZPD(IE)=ZM(IR,IC2)
      PD(IE)=P(IR,IC2,M)
      RHOD(IF)=RHO(IR,IC2,M)
      URD(IE)=UR(IR,IC2,M)
      UZD(IE)=UZ(IR,IC2,M)
      AD(IE)=A(IR,IC2,M)
      RPD(IF)=RBM(KB,KP,M)
      ZPD(IF)=ZBM(KB,KP,M)
      PD(IF)=PB(KB,KP,M)
      RHOD(IF)=RHOB(KB,KP,M)
      URD(IF)=URB(KB,KP,M)
      UZD(IF)=UZB(KB,KP,M)
      AD(IF)=AB(KB,KP,M)
      RETURN
  300 RPD(ID)=RM(IC1,IR)
      ZPD(ID)=ZM(IC1,IR)
      PD(ID)=P(IC1,IR,M)
      RHOD(ID)=RHO(IC1,IR,M)
      URD(ID)=UR(IC1,IR,M)
      UZD(ID)=UZ(IC1,IR,M)
      AD(ID)=A(IC1,IR,M)
      RPD(IE)=RM(IC2,IR)
      ZPD(IE)=ZM(IC2,IR)
      PD(IE)=P(IC2,IR,M)
      RHOD(IE)=RHO(IC2,IR,M)
      URD(IE)=UR(IC2,IR,M)
      UZD(IE)=UZ(IC2,IR,M)
      AD(IE)=A(IC2,IR,M)
      RPD(IF)=RBM(KB,KP,M)
      ZPD(IF)=ZBM(KB,KP,M)
      PD(IF)=PB(KB,KP,M)
      RHOD(IF)=RHOB(KB,KP,M)
      URD(IF)=URB(KB,KP,M)
      UZD(IF)=UZB(KB,KP,M)
      AD(IF)=AB(KB,KP,M)
      RETURN
      END
```

Appendix F. Interpolation Schemes

For data points not falling on the grid points, a two-dimensional interpolation must be used to obtain the values of the variables. A review of current literature shows that there are several interpolation schemes available, including the least-squares method, the double-parabolic method, linear interpolation and spline interpolation. After trying the first two, we have adopted the double-parabolic interpolation method, for both interior and boundary points.

In this Appendix, a brief review of these schemes will be given. A numerical comparison of the accuracy of the least-squares and double-parabolic methods will also be shown.

1. Least-Squares Method - This method was used by Cline and Hoffman [F1]*, Ransom, Hoffman and Thompson [F2], and Richardson [F3], all in connection with the numerical method of characteristics. Referring to Fig. F1, properties at P(r,z) which is not a regular grid-point, are desired. In this method, a bivariate interpolating polynomial

$$f(r,z) = a_0 + a_1 r + a_2 r^2 + a_3 z + a_4 z^2 + a_5 rz \qquad \text{(F1)}$$

is written. The coefficients a_0 to a_5 are determined by a least-squares fit to the nine nearest grid-points where the values of the function are known. Then, values of the function and its derivatives at P can all be obtained from (F1).

The advantage of this method is its ease of application; once the polynomial (F1) is known, the derivatives can be obtained directly. Any number of neighboring grid-points can be used for least-squares fitting, making it easy to apply to curved boundaries. The disadvantage is that the degree of accuracy is not clearly known. The least-squares technique is most suitable for a set of data points which have random error with respect to the true function; the method minimizes the overall error. For our present purpose, the values of the function at the grid-points are to be used for subsequent calculations. The least-squares fitted curve usually does not pass through these original values at grid-points. For a point very close to a grid-point, a discontinuity may be created.

---

* The bracketed numbers designate the references of this appendix shown on page 150.

2. Straight-line Interpolation - Sauerwein [F4] used a linear interpolation method. Due to the geometry of his grid system, which involves three base-points for each characteristic cone, a one-step, two-point interpolation is sufficient; interpolation along two coordinates is not needed. The method is, of course, only of first order accuracy.

3. Double-Parabolic Interpolation - This is the method used in the present program. (For the blast-wave computer code, the one-dimensional parabolic interpolation was used). A second-order polynomial (parabolic) is written with r as the variable and the three coefficients are determined by matching the three grid-points on one horizontal line, such as points 1, 4, and 7 in Fig. F1. Once the values at points A, B, and C are known, a second-order polynomial is written in terms of z and used to interpolate for values at point P.

In order to demonstrate the error involved in using a quadratic approximation to the function f(x) between the points 0, 1 and 2, let us first define the second-degree,

Newton divided-difference formula [F5]; namely,

$$f(x) = f(x_o) + (x - x_o)\, f[x_o,x_1] + (x - x_o)(x - x_1)\, f[x_o,x_1,x_2] + (x - x_o)(x - x_1)(x - x_2)\, \frac{d^3 f(\eta)}{\partial x^3} \qquad \text{(F2)}$$

where

$$x_o \leq \eta \leq x_2$$

and the notation f[ ] is defined by

$$f[a_i,a_j] = \frac{f(a_i) - f(a_j)}{a_i - a_j}$$

$$f[a_i,a_j,a_k] = \frac{f[a_i,a_j] - f[a_j,a_k]}{a_i - a_k}$$

and

$$f[a_i,a_j,a_k,a_\ell] = \frac{f[a_i,a_j,a_k] - f[a_j,a_k,a_\ell]}{(a_i - a_\ell)}$$

Let y(x) represent the quadratic polynomial

$$y(x) = b_1 x^2 + b_2 x + b_3 \tag{F3}$$

then equation (F2) can be written as

$$f(x) = y(x) + (x - x_o)(x - x_1)(x - x_2) \frac{d^3 f(n)}{dx^3} \tag{F4}$$

The error E(x), in the quadratic approximation to f(x) is then given by

$$E(x) = (x - x_o)(x - x_1)(x - x_2) \frac{d^3 f(n)}{dx^3} \tag{F5}$$

From Eq. (F5) we can then see that

$$E(x) = 0(\Delta x^3)$$

or a quadratic approximation Eq.F2, to f(x) is second-order accurate.

4. Spline Interpolation - The spline interpolation method is more sophisticated, more accurate, and has many different versions. A typical one is given by Fowler and Wilson [F6]. The basic concept of spline interpolation may be explained by a simple example. Let the value of y be known at x_A, x_B, and x_C, as shown in Fig. F2.

A polynomial is written for the curve between A and B, and another for that between B and C. The six unknown coefficients are then determined by the four known positions, $y_1(x_A)$, $y_1(x_B)$, $y_2(x_B)$, and $y_2(x_C)$; one slope matching condition, $y_1'(x_B) = y_2'(x_B)$; and one curvature matching condition, $y_1''(x_B) = y_2''(x_B)$.

Although a higher degree of accuracy can be achieved by this interpolation method, it is not suitable for our present purpose because of its complexity, and also because our finite-difference scheme is only of second-order accuracy; any higher order of accuracy in interpolation is not required.

5. Numerical Comparison of Two Interpolation Schemes -

To compare the numerical accuracy of the least-squares method and the double-parabolic method, we have used as a standard the exact solution of Sedov's cylindrical-blast-wave problem, see Ref. F7. An initial time-plane is chosen at a time of 1.591×10^{-4} second, when the shock front is two feet (0.6096 m) from the origin. The blast has an initial energy of $1.5482649 \times 10^{+6}$ ft-lb/ft ($6.8870252 \times 10^{+6}$ joule/m); this gas has a specific heat ratio of 1.4. A typical point near the origin (3.75, 6) was chose, Fig. F3. The error in density as calculated by the least-squares method is 0.8%, and by the double-parabolic method 0.0033%. In this region of the blast wave, the density gradient is very steep, and the accuracy test is a quite severe one.

As shown by this numerical comparison, the double-parabolic method is much superior, and will be adopted for all our calculations.

6. References of Appendix F.

F1. Cline, M.D., and Hoffman, J.D., "The Analysis of Nonequilibrium, Chemically Reacting, Supersonic Flow in Three Dimensions Using a Bicharacteristics Method", Journal of Computationsl Physics, 12, 1-23 (1973).

F2. Ransom, V.H.,Hoffman, J.D., and Thompson, H.D., "Second-Order Bicharacteristics Method for Three-Dimensional, Steady, Supersonic Flow", AIAA Journal, Vol. 10, No. 12, December, 1972.

F3. Richardson, D.J., "The Solution of Two-Dimensional Hydrodynamic Equations by the Method of Characteristics", Methods in Computational Physics, Vol. 3.

F4. Sauerwein, H., "The Calculation of Two and Three-Dimensional Inviscid Unsteady Flow by the Method of Characteristics", MIT: Fluid Dynamics Research Lab. Report No. 64-4, June, 1964.

F5. Beckett, R., and Hurt, J., Numerical Calculations and Algorithms, McGraw-Hill, 1967, Chapter 5.

F6. Fowler, D.H., and Wilson, C.W., "Cubic Spline, A Curve Fitting Routine", Union Carbide Nuclear Co., Report Y-1400.

F7. Chou, P.C., and Karpp, R.R., "Solution of Blast Waves by the Method of Characteristics", DIT Report No. 125-7, Drexel Institute of Technology, September, 1965

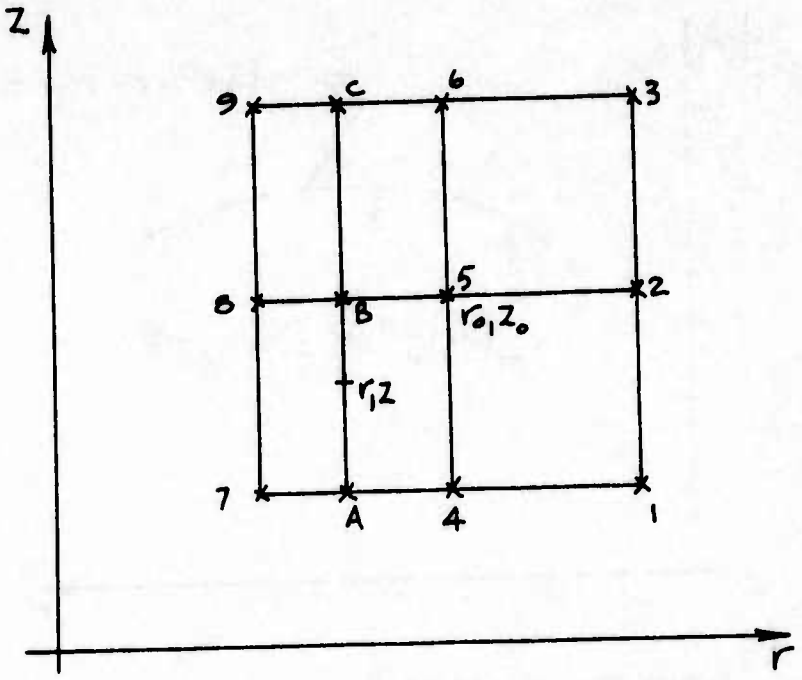


Figure F1. Schematic of data points.

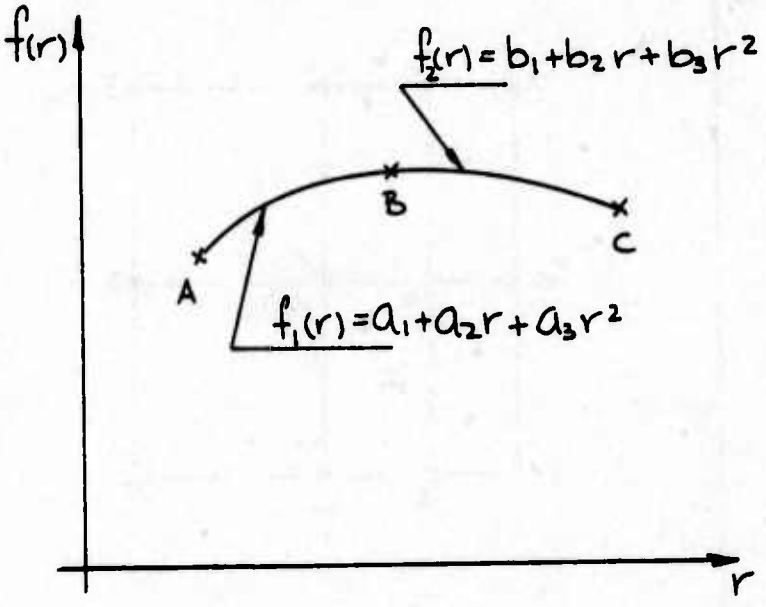


Figure F2. Schematic of parabolic spline.

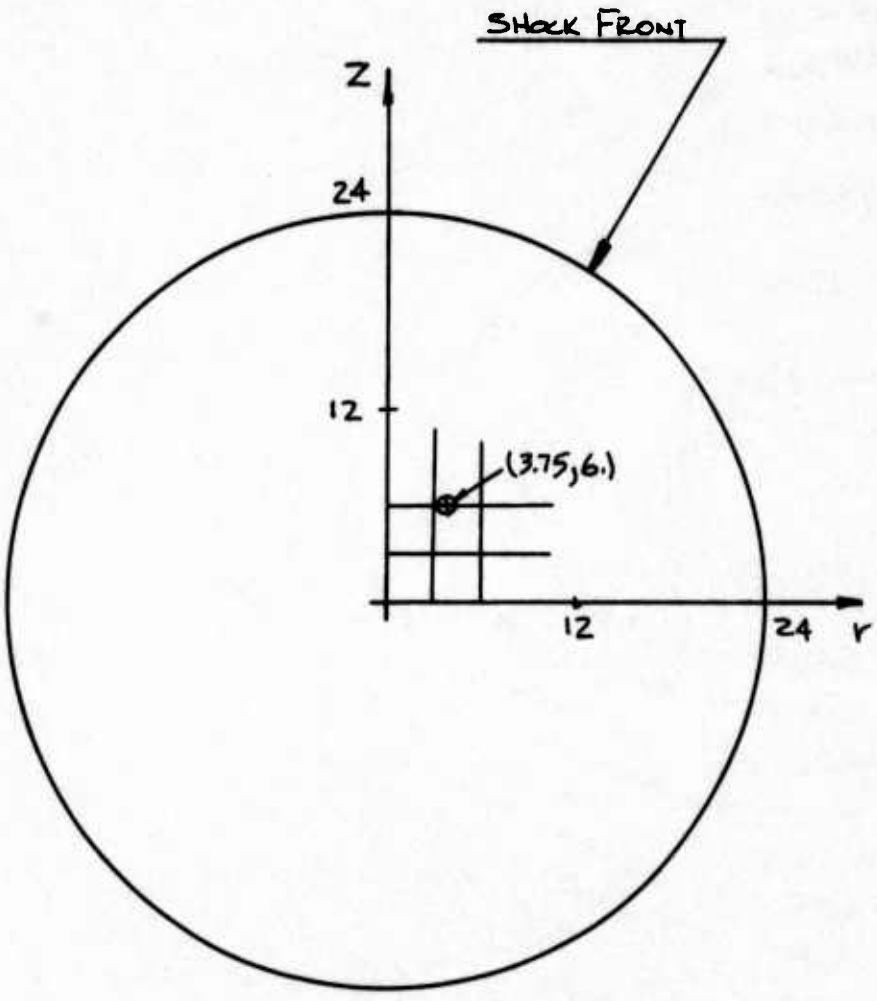


Figure F3. Sketch of Blast Wave Shock Front at 159.1 μsec.

DISTRIBUTION LIST

No. of Copies	Organization
12	Commander Defense Documentation Center ATTN: DDC-TCA Cameron Station Alexandria, VA 22314
2	Commander US Army Materiel Command ATTN: AMCDMA Mr. N. Klein Mr. J. Bender 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Command ATTN: AMCRD, BG H. A. Griffith 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Command ATTN: AMCRD-T 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Materiel Command ATTN: AMCRD-R 5001 Eisenhower Avenue Alexandria, VA 22333
1	Commander US Army Aviation Systems Command ATTN: AMSAV-E 12th and Spruce Streets St. Louis, MO 63166
1	Director US Army Air Mobility Research and Development Laboratory Ames Research Center Moffett Field, CA 94035
1	Commander US Army Electronics Command ATTN: AMSEL-RD Fort Monmouth, NJ 07703
1	Commander US Army Missile Command ATTN: AMSMI-R Redstone Arsenal, AL 35809
1	Commander US Army Tank Automotive Command ATTN: AMSTA-RHFL Warren, MI 48090
2	Commander US Army Mobility Equipment Research & Development Center ATTN: Tech Docu Cen, Bldg. 315 AMSME-RZT Fort Belvoir, VA 22060
1	Commander US Army Armament Command Rock Island, IL 61202
1	Commander US Army Harry Diamond Laboratories ATTN: AMXDO-TI 2800 Powder Mill Road Adelphi, MD 20783
3	Drexel University Mechanical Engineering & Mechanics Department ATTN: Dr. Pei Chi Chou 32nd & Chestnuts Streets Philadelphia, PA 19104

Aberdeen Proving Ground

Marine Corps Ln Ofc
Director, USAMSAA