A comprehensive computational procedure is presented for predicting the supersonic region of the flow field on advanced reentry vehicle shapes in steady flight at pitch and yaw. The procedure utilizes explicit second order accurate finite difference methods applied to the conservation law form of the steady inviscid flow equations. Improved numerical methods are used at the body surface and the bow shock wave. Provisions for treating body...
geometries with discontinuous slopes are also included. Either perfect gas or real gas equilibrium thermodynamic properties can be used.

The computational procedure is implemented as a fortran computer code which provides a practicable representation of the inviscid flow field and the resulting aerodynamic force and moment on the vehicle.

In this report (Vol. I) the analytical and numerical development of the procedure is presented and the associated computer code is described. A comparison report (Vol. II User's Manual) contains detailed instructions for operating the code and interpreting the output results.

This report describes the analytical, and computational aspects of a computer program for predicting inviscid flow fields and aerodynamics on realistic reentry configurations. This work was performed by members of the Mathematical and Engineering Analysis Branch of NSWC/WOL. The initial code development was supported by the Naval Sea Systems Command under the Aeroballistic Reentry Technology (ART) program with some of the fundamental analytical and numerical work supported by NSWC Independent Research Funds. Most of the final code development and documentation was supported by the Air Force Space and Missile System Organization under the technical management of the Aerospace Corporation.

The authors gratefully acknowledge the efforts of Mr. R. Feldhuhn, NSWC coordinator for the ART program, who was responsible for initiating the present work and whose continued interest and support throughout the investigation was invaluable. The authors are also indebted to Mr. M. Lyons and Dr. E. Ndefo of the Aerospace Corporation for several stimulating technical discussions which lead to important improvements in the final code.

C. A. FISHER
By direction

1
CONTENTS

0. INTRODUCTION ................................... 4

PART I: ANALYSIS

1. NOTATION (PART I) ................................ 8

2. GOVERNING EQUATIONS AND BOUNDARY CONDITIONS .......... 13
   2.1 Steady Flow Equations. ............................. 13
   2.2 Boundary Conditions. ................................. 15

3. BASIC COMPUTATIONAL ALGORITHM .......................... 16
   3.1 Computational Region and Transformed Equations ........ 16
   3.2 Interior Points. ...................................... 22
   3.3 Bow Shock Points ..................................... 25
   3.4 Body Surface Points ................................... 30
   3.5 Symmetry and Periodic Boundary Points ................. 37
   3.6 Step Size and Stability. .............................. 39

4. SPECIAL FEATURES ....................................... 41
   4.1 Discontinuities in Body Slope. ......................... 41
   4.2 Wall Entropy Reduction ................................. 53
   4.3 Mesh Clustering ....................................... 57

5. FORCE AND MOMENT CALCULATIONS .......................... 63

APPENDIX A - DIFFERENTIAL EQUATIONS FOR BOUNDARY POINTS .... 68

APPENDIX B - SYMMETRY CONDITIONS. .......................... 82

APPENDIX C - CFL CONDITIONS ............................... 85

6. REFERENCES ............................................. 98

PART II: PROGRAMMING

7. GENERAL REMARKS ...................................... 100

8. COMMON ................................................. 101

9. MAIN ................................................... 112
   9.1 Section 1 ............................................. 112
   9.2 Section 2 ............................................. 114
10. SUBROUTINES USED IN THE FLOW FIELD CALCULATION. .......... 118
  10.1 BODYP, ENTRY BODYP .................................. 118
  10.2 DECODE .............................................. 119
  10.3 EVAL, ENTRY EVALSY, ENTRY EVALPR ...................... 120
  10.4 JUMP ................................................. 122
  10.5 TRANF, ENTRY TRANFW .................................. 123
  10.6 TRANG, ENTRY TRANGW .................................. 124
  10.7 WALL ................................................. 125
  10.8 SHOCK ................................................. 128

11. AUXILIARY SUBROUTINES ................................... 129
  11.1 INTEG ................................................ 129
  11.2 INTRPL .............................................. 130
  11.3 REZONE .............................................. 130
  11.4 RGAS ............................................... 131
  11.5 HRGAS, ENTRY ARGAS .................................... 132
  11.6 SERCH, LOCATE ....................................... 133
  11.7 SHFAX, SHFAXD ...................................... 133

12. INPUT-OUTPUT ROUTINES .................................... 135
  12.1 BODY, ENTRY BODYN, ENTRY BODYR ......................... 135
  12.2 FIELD ............................................... 136
  12.3 OUT ............................................... 136
  12.4 RECOVR, SAVE ....................................... 136
  12.5 TRANFD ............................................. 138
  12.6 TRANGD ............................................. 138

APPENDIX D - LISTINGS ...................................... 139
0. INTRODUCTION

An important aspect of the design and evaluation of maneuverable and advanced ballistic reentry vehicles is the determination of the inviscid flow field surrounding the body. The inviscid flow field provides surface pressure distributions required for determining the aerodynamic loading on the vehicle and other surface information which is needed as input for determining surface heat transfer rates and other boundary layer effects. A cost effective method for obtaining this information is to use high-speed computer codes which numerically solve the steady, three-dimensional, inviscid flow equations associated with arbitrary shaped reentry vehicles flying at supersonic/hypersonic speeds.

The numerical calculation of the inviscid flow field over reentry vehicles is divided into two parts—the blunt body region calculation and the supersonic region calculation (see Fig. 1). The blunt body region calculation determines the transonic flow field near the stagnation point. The supersonic region calculation determines the flow field downstream of the blunt body region. The differing nature of the flow in these two regions requires significantly different computer codes for calculating each region. The blunt body region is computed first and is continued downstream until supersonic flow is established everywhere in the shock layer. The computed results from this calculation are used to establish an "initial" data plane which is used to start the supersonic region calculation. This latter portion represents the majority of the total flow field on maneuverable and high performance ballistic reentry vehicles.
Fig. 1. Computational regions and cylindrical coordinate system for reentry vehicle inviscid flow calculations.
In this report, a computer code for performing the supersonic region calculation is described. This code is based on the conservation law form of the steady, inviscid equations. Codes of this type are sometimes referred to as shock capturing techniques since internal (embedded) shock waves in the flow field are computed in an approximate manner without explicitly locating (or tracking) the shock surface. The present code incorporates improved numerical methods at the body surface and the bow shock wave which yield a wider applicability to missile design than existing codes of this type (e.g. ref. 1).

This report is divided into two parts. In Part I, the partial differential equations, boundary conditions, and finite difference equations which are the basis of the computer code are discussed. In Part II, the fortran computer code is described. A companion report*, the Users' Manual, contains detailed instructions for running the code on CDC 6000 series and 7600 machines and for interpreting the output results.

PART I: ANALYSIS
1. NOTATION (PART I)

Symbols

- $a$: sound speed
- $a_3, a_4, a_5, a_7$: body geometry parameters; see, (3.16a) and (3.16b)
- $A$: see (3.3d)
- $A_1, A_0$: see (3.9c)
- $A_{\text{ref}}$: reference area for force and moment coefficients nondimensionalized by $R_0^2$ (see Fig. 7)
- $A$: see (A-5a)
- $A^*$: characteristic matrix (Appendix A)
- $b = b(\phi, z)$: body shape function
- $B$: see (3.3d)
- $B$: see (A-5b)
- $c = c(\phi, z)$: shock shape function
- $C$: see Fig. 7
- $C_1, C_2$: see (3.9c)
- $C$: step size factor, see (3.25)
- $D$: see (3.22)
- $\hat{e}_z, \hat{e}_r, \hat{e}_\phi$: unit vectors in $z, r, \phi$ directions, respectively
- $\hat{e}_r, \hat{e}_n$: see Fig. 4 and (4.3)
- $e_1$: see (3.3e)
- $E$: see (3.3c)
- $E$: see (2.1b)
- $f = f(X, Y, Z)$: clustering transformation, see (3.2)
- $F$: see (3.3a)
- $F_a, F_n, F_y$: aerodynamic force components, see Fig. 7 and (5.1) - (5.3)
- $\gamma$: see (2.1b)
\[ g = g(Y,Z) \] clustering transformation, see (3.2)

\[ G \] see (3.3b)

\[ \mathcal{H} \] see (2.1b)

\[ h \] enthalpy,

\[ H_\infty \] total enthalpy, see (2.2)

\[ H_0 \] see (3.8b)

\[ j \] 0 or 1, determines scheme, see (3.6a) and (3.6b)

\[ k \] step number

\[ K_1, K_2 \] thermodynamic derivatives, see (3.16) and (A-4)

\[ \lambda, \lambda_1, \lambda_2, \lambda_o, \lambda_1^{(1)}, \lambda_2^{(2)} \] characteristic null vectors, see Appendix A

\[ m \] denotes mesh point \( Y = Y_m \) see (3.4)

\[ m_s \] limit mesh point for wall entropy reduction, see Sec. 4.2

\[ M \] value of \( m \) corresponding to \( Y = 1 \), see Figs. 2 and 3

\[ M_\infty \] free stream Mach number

\[ M^C_a, M^C_n, M^C_y \] aerodynamic moment components about \( C \), see Fig. 7 and (5.4) - (5.6)

\[ \mathcal{M} \] Mach number in front of oblique shock, see (4.9)

\[ n \] denotes mesh point \( X = X_n \), see (3.4)

\[ \mathbf{n} \] normal vector

\[ N \] value of \( n \) corresponding to \( X = 1 \) (bow shock), see Figs. 2 and 3

\[ \mathbf{O} \] matrix, see Appendix A

\[ p \] pressure

\[ P \] \( \log p \)

\[ \rho \] see (3.16a) and (3.16b)
\( \Phi \) see (3.19)

\( Q \) \((p,u,v,w)\)

\( r \) radial coordinate, nondimensionalized by \( R_0 \)

\( R_0 \) nondimensionalizing length

\( R_s, \tilde{R} \) see Appendix A

\( s \) entropy, see (3.9c)

\( T_{g_5}, T_{f_4}, T_{f_6}, T_f \) transformation quantities, see (3.3h) and (3.3i)

\( u \) velocity component in \( r \) direction

\( U = (U_1, U_2, U_3, U_4) \) conservation vector, see (2.1a)

\( \bar{U} \) averaged values of \( U \), see (4.11)

\( v \) velocity component in \( \phi \) direction

\( V = \sqrt{u^2 + v^2 + w^2} \) velocity vector

\( V_n \) normal component of velocity

\( V_\phi \) velocity component in \( \phi \) direction

\( V_2 \) \( v + (b_\phi/b)u \)

\( V \) see (3.17a) and (3.17b)

\( w \) velocity component in \( z \) direction

\( x, y \) see Fig. 1 or Fig. 7

\( \bar{x}, \bar{y}, \bar{z} \) see (3.1)
coordinates in computational space, see Figs. 2 & 3

\(X, Y, Z\)  
\(X_x, X_y, X_\phi\)  
\(Y_z, Y_\phi\)  
\(z(\zeta)\)  
\(z_c\)  
\(z_{\text{ref}}\)  
\(z_0\)  
\((z_{\text{c.p.}})_p\)  
\((z_{\text{c.p.}})_y\)  
\(\alpha\)  
\(\beta\)  
\(\beta_0\)  
\(\beta_1\)  
\(\gamma\)  
\(\Gamma\)  
\(\delta\)  
\(\Delta X/\Delta Y (j = 0), -\Delta X/\Delta Y (j = 1)\)  
\(\Delta Z\)  
\(\eta = (\eta_1, \eta_2, \eta_3, \eta_4)\)  
\(\theta\)  
\(\Theta = (\Theta_1, \Theta_2, \Theta_3, \Theta_4)\)  
\(\lambda, \lambda_0, \lambda_+\)  
\(\mu, \mu_1, \mu_2, \mu_3\)  

\(\text{stability parameters, see Sec. 3.6} \)
\[ v_w = \sqrt{1 + (b_\phi/b)^2 + b_z^2} \]

\[ v_\infty = \sqrt{1 + (c_\phi/c)^2 + c_z^2} \]

\[ \ell = (\ell_1, \ell_2, \ell_3, \ell_4) \] see (3.16a)

\( \rho \) density

\( \phi, \theta \) see Fig. 4

\( \phi \) azimuthal angle

\( \phi_0 \) \( \pi \) for the symmetric problem; 2\( \pi \) for the nonsymmetric problem

\( \phi \) see (3.8c)

**Subscripts**

\( n, m \) quantity evaluated at \( X = X_n, Y = Y_m \)

\( s \) quantity evaluated at bow shock wave

\( w \) quantity evaluated at wall

\( \mp \) quantities at upstream and downstream side of body

slope discontinuity in section 4.1 only

**Superscripts**

\( k \) quantity at step \( Z = Z^k \)

\( * \) predicted value (except in Appendix A)

**Other**

A dependent variable used as a subscript denotes partial differentiation with respect to that variable; e.g., if \( b = b(\phi, z) \) then

\[ b_z = \frac{\partial b}{\partial z} \]

\( |q| \) denotes the absolute value of \( q \) if \( q \) is a scalar and the modulus (or magnitude) of \( q \) if \( q \) is a vector.

\( \cdot \) and \( \times \) denote vector inner product and cross products, respectively.
2. GOVERNING EQUATIONS AND BOUNDARY CONDITIONS

2.1 Steady Flow Equations

Consider a body oriented cylindrical coordinate system $r, \phi, z$ illustrated in Figure 1. With respect to these coordinates, the conservation of mass and momentum equations for a steady, inviscid, flow can be written as a system of (weak) conservation laws; i.e.,

$$\frac{\partial U}{\partial z} + \frac{\partial \Psi}{\partial r} + \frac{\partial \Phi}{\partial \phi} + \mathcal{E} = 0$$

where

$$U = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} \rho w \\ p + \rho w^2 \\ \rho w u \\ \rho w v \end{pmatrix}$$

and

$$\Psi = \begin{pmatrix} \rho u \\ \rho w u \\ p + \rho u^2 \end{pmatrix}, \quad \Phi = \frac{1}{r} \begin{pmatrix} \rho v \\ \rho w v \\ \rho u v \end{pmatrix}, \quad \mathcal{E} = \frac{1}{r} \begin{pmatrix} \rho w u \\ \rho (u^2 - v^2) \end{pmatrix}.$$ (2.1a)

In the above, $p$ denotes pressure and $\rho$ denotes density and $u,v,w$ are the velocity components in the $r,\phi,z$ directions, respectively. The energy equation for a steady inviscid flow with an isoenergetic free stream can be reduced to an algebraic equation; viz.

$$h + \frac{1}{2}(u^2 + v^2 + w^2) = H_{\infty} = h_{\infty} + \frac{1}{2}(u_{\infty}^2 + v_{\infty}^2 + w_{\infty}^2).$$ (2.2)

Here $h$ is the enthalpy and $H_{\infty}$ is the total energy in the free stream (which is constant). In the above equations and throughout our discussion, all lengths are non-dimensionalized by $R_0$, some body length; all other quantities are dimensional.*

*It is understood that the dimensions are consistent, i.e., velocity has dimensions of (pressure/density)$^{1/2}$ and enthalpy has dimensions of pressure/density.
The flow is assumed to be in local thermodynamic equilibrium (or frozen-equilibrium) so that known relationships exist between the thermodynamic variables, $p$, $\rho$, and $h$. Two other thermodynamic variables will be introduced later; viz., the sound speed, $a$, and the entropy, $s$. We assume the standard form of the thermodynamic relations which expresses $h$, $\rho$, and $a$ as functions of $p$ and $s$. In addition, we assume that $s$ can be expressed as a function of $p$ and $h$ and also as a function of $p$ and $\rho$.

The computational algorithm which will be described in the next section specifically requires that $h = h(p, \rho)$, $a = a(p, \rho)$ and, at certain points, $\rho = \rho(p, s)$—these relations being given either in closed form or as curve fits. Note that the sound speed can be defined in terms of $h(p, \rho)$ by

$$a^2 \frac{\partial h}{\partial p} \frac{\rho}{P} + (\frac{\partial h}{\partial \rho} P) = \frac{a^2}{\rho}. \tag{2.3}$$

For the special case of a perfect gas* with ratio of specific heats, $\gamma$, we have

$$h = \frac{\gamma}{\gamma-1} \frac{P}{\rho}, \quad a^2 = \gamma \frac{P}{\rho} \tag{2.4} \quad \text{[perf]}$$

and

$$\frac{\rho}{\rho_m} = \exp \left\{ \frac{1}{\gamma} \left( \log \left( \frac{P}{\rho_m} \right) - (s - s_m)/c_v \right) \right\}. \tag{2.5} \quad \text{[perf]}$$

where $c_v$ is the specific heat at constant volume.

---

*The term perfect gas used herein refers to a gas satisfying the perfect gas law (thermodynamically perfect) and having constant specific heats (calorically perfect). Throughout this report equations which are valid only for perfect gases are followed by [perf].
2.2 Boundary Conditions

The shock layer is bounded by the given body surface, and the bow shock wave, which is an unknown to be determined. The body surface is assumed to be prescribed in the form \( r = b(\phi, z) \) where the function \( b(\phi, z) \) is continuous and piecewise twice continuously differentiable. On the body surface, the boundary condition for inviscid flow is that the normal component of velocity must vanish, i.e.,

\[
- b \frac{\partial w}{\partial z} - \left( \frac{\partial b}{\partial z} \right) v = 0
\]  

(2.6)

The bow shock surface is assumed to be in the form \( r = c(\phi, z) \), an unknown function, to be determined in the calculation. On the bow shock surface the Rankine-Hugoniot relations must hold. These relations are:

\[
\begin{align*}
\rho V_n &= \rho_\infty V_{n\infty}, \\
\vec{V} - \hat{n}V_n &= \vec{V}_\infty - \hat{n}V_{n\infty}, \\
p + \rho V_n^2 &= p_\infty + \rho_\infty V_{n\infty}^2, \\
h + \frac{1}{2}V_n^2 &= h_\infty + \frac{1}{2}V_{n\infty}^2.
\end{align*}
\]

(2.7)

In the above, \( V_n = \hat{t} \cdot \vec{V} \) is the component of velocity normal to the shock surface and, \( \vec{V}_t = \vec{V} - \hat{n}V_n \), where \( \hat{t} \) is the unit vector normal to the shock surface given by

\[
\hat{t} = \frac{1}{v_s} \left( c_z \hat{e}_z - \hat{e}_r + c_\phi \hat{e}_\phi \right),
\]

(2.7a)

\[
v_s = \sqrt{1 + c_z^2 + (c_\phi/c)^2}
\]

where \( \hat{e}_z, \hat{e}_r, \) and \( \hat{e}_\phi \) are the unit vectors in the \( z, r, \) and \( \phi \) directions, respectively. In (2.7), the subscript \( \infty \) refers to the free stream quantities (upstream of the shock); all other quantities are values immediately behind (downstream of) the shock.
1. BASIC COMPUTATIONAL ALGORITHM

The region of the shock layer in which the axial velocity component is supersonic (i.e., $w_{a}$) is referred to here as the supersonic region. It can be shown that in this region the system (2.1) is of hyperbolic type with the $z$ axis as the time-like direction. This implies among other things that the numerical solution of (2.1), (2.2) can be obtained by marching stepwise in the $z$ direction. The procedure adopted here employs an explicit finite difference method which advances (i.e., determines) the flow variables $c, p, w, u, v$ and the bow shock geometry $c_{b}, c_{w}$ to station $z + \Delta z$ using the known quantities at station $z$. The step size $\Delta z$ is chosen to satisfy a stability criterion.* The calculation begins at some plane $z = z_{o}$ in the supersonic region where all flow quantities and shock geometry are known. This plane will be referred to as the initial plane. The calculation is continued downstream to any desired axial station (or until the axial velocity becomes sonic) by repeated use of the procedure using the previous $z + \Delta z$ as the next known station.

3.1 Computational Region and Transformed Equations

In the following discussions two different problems will be considered: the symmetric and the non-symmetric problems. In the symmetric problem, the body is assumed to be symmetric about the pitch plane (i.e., in Fig. 1, the body is symmetric about the $x - z$ plane and $\phi = 0$). The calculation for this problem need only be performed for $0 \leq z \leq z_{o}$ since the flow field is symmetric about $\phi = 0$ and $\phi = \pi$ (wind and lee sides, respectively).

*See Section 3.6
The planes $\phi = 0$ and $\phi = \pi$ are symmetry boundaries where all variables are even functions of $\phi$ except $v$ which is an odd function of $\phi$. In the non-symmetric problem, the body need not be symmetric and/or $\beta \neq 0$. In this problem, the calculation must be performed for $0 \leq \phi \leq 2\pi$; and all variables are periodic functions of $\phi$ with period $2\pi$.

For both problems, the shock layer for $z \geq z_0$ is transformed into the computational region $Z \geq z_0$, $0 \leq X \leq 1$, $0 \leq Y \leq 1$. The transformation is conveniently expressed as a composite of two mappings. The first is given by the usual normalizing transformations;

$$
\begin{align*}
\bar{z} &= z \\
\bar{x} &= \frac{r-b(z,\phi)}{c(z,\phi) - b(z,\phi)} \\
\bar{y} &= \phi/\phi_o
\end{align*}
$$

where $\phi_o$ is $\pi$ for the symmetric problem and $2\pi$ for the non-symmetric problem. This mapping by itself transforms the shock layer into the region $\bar{z} \geq z_0$, $0 \leq \bar{x} \leq 1$, $0 \leq \bar{y} \leq 1$. The second mapping maps this region one-to-one onto itself. For computational purposes, it is convenient to express the second mapping in inverted form, i.e.,

$$
\begin{align*}
\bar{z} &= Z \\
\bar{x} &= f(X,Y,Z) \\
\bar{y} &= g(Y,Z)
\end{align*}
$$

where $f(0,Y,Z) = 0$, $f(1,Y,Z) = 1$; $g(0,Z) = 0$, $g(1,Z) = 1$. The primary purpose of the second mapping is to cluster computational points in the shock layer by choosing $f$ and $g$ appropriately (see sec. 4.3 for a discussion of this feature). The mapping functions $f(X,Y,Z)$ and $g(Y,Z)$ must be
twice continuously differentiable and be given so that the functions and their derivatives up to and including the second order are accordingly smoothly defined. Apart from this restriction (and certain ones which will be imposed in sec 3.5), the functions \( f(X,Y,Z) \) and \( g(Y,Z) \) can be arbitrary. When no transformation is desired, then one should set \( f = X \) and \( g = Y \).

The governing equations, (2.1), when transformed to the computational space \((X,Y,Z)\) by (3.1) - (3.2) become

\[
\frac{3U}{3Z} + \frac{3F}{3X} + \frac{3G}{3Y} + E = 0 \tag{3.3}
\]

where

\[
F = X_z U + X_r \phi + X_p \psi = \left( \begin{array}{c} \rho A \\ X_z p + \rho w A \\ X_r p + \rho u A \\ \frac{1}{r} X_p p + \rho v A \end{array} \right) \tag{3.3a}
\]

\[
G = Y_z U + Y_r \phi = \left( \begin{array}{c} \rho B \\ Y_z p + \rho w B \\ \rho u B \\ \frac{1}{r} Y_p p + \rho v B \end{array} \right) \tag{3.3b}
\]

\[
E = E_k - \left( \frac{3X}{3X} + \frac{3Y}{3Y} \right) U + \frac{3Y}{3X} \phi + \frac{3X}{3Y} \psi + \left( \frac{\partial \phi}{\partial X} + \frac{\partial \psi}{\partial Y} \right) \tag{3.3c}
\]

\[
E_k = W_1 + (Y_z T_{8_5} + T_{f_4} X_z + T_{f_6} p)
\]

\[
wE_1 + \frac{\partial u^2}{r} + X_r T_{f_6} p
\]

\[
W_1 + \frac{1}{r} [\rho v u + p (X \phi T_{f_4} + Y \phi T_{8_5} + T_{f_7})] \tag{3.3c}^*
\]

*The awkward appearing notation \((T_{f_4}, T_{f_6}, \text{ etc.})\) follows the FORTRAN formulation of our code.
In the above,

\[ A = X \frac{\partial}{\partial x} + X \frac{\partial u}{\partial \phi} + \frac{1}{r} \frac{\partial \phi}{\partial \phi}, \quad B = Y \frac{\partial}{\partial y} + \frac{1}{r} \frac{\partial \phi}{\partial \phi} \]  

(3.3d)

\[ e_1 = \rho \left( \frac{u}{r} + AT_{f_4} + BT_{f_5} + \frac{v}{r} T_{f_7} \right) \]  

(3.3e)

\[ Y_z = - \frac{\partial \phi}{\partial \phi}, \quad Y_\phi = \frac{1}{(\phi \partial \phi)} \]  

(3.3f)

\[ X_\phi = \frac{1}{[(\phi \partial \phi)(c-b)]} \]  

\[ X_z = - \frac{(f_Z + Y_z \phi)}{f_X} + X_\phi [(f-1)b_z - f_{c_z}] \]  

(3.3g)

\[ X_\phi = - \frac{Y_\phi \phi}{f_X} + X_\phi [(f-1)b_\phi - f_{c_\phi}] \]  

(3.3h)

\[ T_{g_5} = \frac{g_{Y_Y}}{g_Y}, \quad T_{g_6} = \frac{g_{Z_Y}}{g_Y} \]  

(3.3i)

\[ T_{f_4} = \frac{f_{XX}}{f_X}, \quad T_{f_7} = Y_\phi \frac{f_{YY}}{f_X} - \frac{(b_\phi - c_\phi)}{(c-b)} \]  

(3.3j)

\[ T_{f_6} = T_{g_6} + \frac{(f_{XX} + Y_z f_{YX})}{f_X} - \frac{(b_z - c_z)}{(c-b)} \]  

(3.3k)

Note that in the above, the body slopes, \( b_z \) and \( b_\phi \), are derived from the given body geometry function; the shock slopes, \( c_z \) and \( c_\phi \), are unknowns to be determined in the calculation.

The system of partial differential equations (3.3) is discretized and solved numerically in the computational space using a mesh defined by

\[ \{(X_n, Y_m) : X_n = (n-1)\Delta X (n=1, 2, \ldots, N), \quad Y_m = (m-1)\Delta Y (m=1, 2, \ldots, M)\} \]  

(3.4)

where \( \Delta X = 1/(N-1) \) and \( \Delta Y = 1/(M-1) \). In Figures 2 and 3, we depict typical discretized computational planes \( Z = z = \) constant and the corresponding
Fig. 2. Computational and corresponding physical plane for symmetric problem.
Fig. 3. Computational and corresponding physical plane for the non-symmetrical problem.
physical planes for the symmetric and non-symmetric problems, respectively.
As indicated in the figures, there are four types of points each requiring
differing numerical procedures; interior points corresponding to
n = 2, · · · , N - 1; m = 2, · · · , M - 1; points on the symmetry or periodic boundary planes
Y = 0 and Y = 1 (m = 1 and m = M, respectively); boundary points at the bow shock,
X = 1 (n = N); boundary points at the body surface, X = 0 (n = 1). For all points,
the solution is advanced using predictor-corrector finite difference methods,
i.e., the known solution at Z, say, is used to determine temporary (i.e.,
predicted) values at Z + ΔZ; then the predicted values are used to determine
the final solution (i.e., corrected) values at Z + ΔZ.

In section 3.2-3.5 below, the particular method used for each of
the above four types of points will be described. We shall assume that the
quantities ρ, u, v, w, p, c, cϕ, cz are known for Z = Zk on the mesh defined by
(3.4). The objective is to determine these quantities on this mesh for
Z = Zk+1 = Zk + ΔZ.

3.2 Interior Points

The numerical solution for all interior points is obtained by
approximating the system (3.1) by the second-order accurate finite difference
scheme of MacCormack (ref. 2). In the code, either of two MacCormack
schemes can be selected. These are given by (j = 0 or 1):

22
\[ u_{n,m}^* = u_{n,m}^k + \Lambda Z (\frac{\partial u}{\partial Z})_{n,m}^k \]

\[ \frac{\partial u_{n,m}^k}{\partial Z} = -\left( \frac{\partial F_{n,j-1,m}^k - F_{n,j+1,m}^k}{\Lambda X} \right) - \left( \frac{\partial G_{n,j,m+1}^k - G_{n,j,m}^k}{\Lambda Y} \right) - e_{n,m}^k \]

\[ u_{n,m}^{k+1} = \frac{1}{2} u_{n,m}^k + u_{n,m}^* + \Lambda Z (\frac{\partial u}{\partial Z})_{n,m}^* \]

\[ \frac{\partial u_{n,m}^*}{\partial Z} = -\left( \frac{\partial F_{n,j-1,m}^* - F_{n,j+1,m}^*}{\Lambda X} \right) - \left( \frac{\partial G_{n,j,m+1}^* - G_{n,j,m}^*}{\Lambda Y} \right) - e_{n,m}^* \]

In the above, \( u_{n,m}^* = U(Z^k, X_n, Y_m) \). The quantities \( F_{n,m}^k, G_{n,m}^k, E_{n,m}^k \) are evaluated from (3.3a) - (3.3c) at the point \( (Z^k, X_n, Y_m) \) using the known quantities \( \rho, u, v, w, p, c, \) and \( c_z \) with \( c_\phi \) given by (3.5). The quantities \( u_{n,m}^* \) are the predicted values of \( U \) at \( (Z^{k+1}, X_n, Y_m) \) from which predicted values of the flow quantities \( \rho, u, v, w, p \) can be determined using the definitions (2.1a), (2.2), and the thermodynamic relations. (This is described in the next paragraph). The quantities \( F_{n,m}^*, G_{n,m}^*, E_{n,m}^* \) are evaluated from (3.3a) - (3.3c) at the point \( (Z^{k+1}, X_n, Y_m) \) using: the predicted values of \( \rho, u, v, w, p, c_\phi, c_z \), and the corrected values of \( c \).

The predictor and corrector procedure for \( c, c_\phi, c_z \) will be described in the discussion of the bow shock points.

In the above, the flow variables \( \rho, u, v, w, p \) must be determined (or decoded) from the computed conservation variables \( U_1, U_2, U_3, U_4 \) after both the predictor and corrector steps. For decoding, it is convenient to introduce an effective gamma defined by

\[ \gamma = \frac{1}{1 - \frac{P_t}{\rho_0}}. \]
From the definitions (2.1a), we have

\[
\begin{align*}
    u &= U_3/U_1, \\
    v &= U_4/U_1, \\
    \rho &= U_2 - U_1 w, \\
    \phi &= U_1/w
\end{align*}
\] (3.8a)

Substitution of (3.8a) into (2.2) yields

\[
    h = \frac{1}{2} [H_o - w^2], \\
    H_o = 2H_w - \frac{U_3^2 + U_4^2}{U_2^2}
\] (3.8b)

Substituting \( \rho \) and \( p \) given by (3.8a), and \( h \) given by (3.8b) into (3.7), we obtain a quadratic equation for \( w \). The root of this equation corresponding to \( w^2 > \gamma p/\rho \) is

\[
w = \frac{U_2 [\Gamma + \sqrt{1 - \phi}]}{U_1 (\Gamma + 1)}, \quad \phi = (\Gamma^2 - 1)[H_o \left(\frac{U_4}{U_2}\right)^2 - 1]
\] (3.8c)

In the case of a perfect gas, \( \Gamma = \gamma \), which is known, and equations (3.8c) and (3.8a) provide the desired decoding formulas. For a more general gas where \( h = h(p, \rho) \) is given as a curve fit to thermodynamic data, \( \Gamma \) is not known a-priori and the decoding cannot be performed in closed form. In this case the decoding can be performed by solving the nonlinear equation

\[
h(p, \rho) - \frac{1}{2} (H_o - w^2) = 0
\]

for \( w \). In this equation \( h(p, \rho) \) is the value of \( h \) obtained from the curve fits with \( p \) and \( \rho \) defined by (3.8a). In the code, this equation is solved iteratively using the secant method. To start the iteration in the predictor cycle, the initial guess for \( w \) is obtained using (3.8c) with the value of \( \Gamma \) from the previous step; the second guess uses (3.8c) with the value of \( \Gamma \) defined from (3.7) which results from the initial trial.
For the corrector cycle, the iteration is started similarly except
the initial guess uses in (3.8c) the value of $\Gamma$ defined in the predictor
step.

3.3 Bow Shock Points

At the bow shock boundary, the Rankine-Hugoniot relations (2.7)
must be satisfied. These relations give the flow variables at the shock
in terms of the shock geometry $c$, $c_\phi$, and $c_z$ (which are unknowns) and the
free stream quantities. In the present procedure, a special system of
equations for $c$, $c_\phi$, and $c_z$ is numerically solved using a second-order
accurate predictor-corrector method to advance the shock geometry. This
procedure differs from the more common practice of determining $c_\phi$ using
central differences of $c$ (see, refs 1,3,6,7,8, and 9).

The system of equations used for advancing $c$, $c_\phi$, and $c_z$ is developed
in Appendix A (see, (A-20) - (A-22)). The resulting system is,

\[
\frac{3c}{3Z} = c_z - \left( \frac{Y}{3Y} \right) c_\phi 
\]

\[
\frac{3c_\phi}{3Z} = \frac{3c}{3Y} - \left( \frac{Y}{3Y} \right) \frac{3c_\phi}{3Y} \] (3.9a)

\[
\frac{3c_z}{3Z} = \frac{C_2}{C_1} \left( \frac{\mathcal{R}_s}{c} - \frac{c_\phi}{c} \left[ \left( \frac{Y}{3Y} \right) - \left( \frac{Y}{3Y} \right) \frac{3c_z}{3Y} \right] - \frac{c_\phi}{c} \left( c_z - \frac{Y}{3Y} \frac{3c_\phi}{3Y} \right) \right) 
\] (3.9b)

where

\[
C_1 = \left( \frac{\nu_s w_\infty - c_z V_{n_\infty}}{A_1} \right) - \left( p_p \right) [1 + (c_\phi/c)^2]/\nu_s^2 
\]

\[
C_2 = \left[ \frac{\nu_s w_\infty - (c_\phi/c) V_{n_\infty}}{A_1} \right] + c_z (c_\phi/c) (p_p \infty)/\nu_s^2 
\]
\[ A_1 = [\beta_0 \rho \infty \left( V_{n \infty} + p(V_s \omega - c_z V_{n \infty}) \right) A_0 + \beta_0 \rho \infty (V_{n \infty} - V_n) ] \]

\[ A_0 = \frac{\left( V_{n \infty} - V_n \right) \left[ a^2 + V_n^2 + \mathcal{K}_1 \left( \frac{a^2}{\rho} \right) V_n \left( V_{n \infty} - V_n \right) \right]}{v_{n \infty} \left( a^2 - V_n^2 \right)} \]

\[ \beta_0 = \frac{1}{a} \sqrt{(w^2 - a^2) \left[ 1 + \left( \frac{c_r}{c} \right)^2 \right] + \left[ u - \left( \frac{c_r}{c} \right) v \right]^2} \]

\[ \mathcal{K}_1 = \left( \frac{\partial \rho}{\partial \rho} \right)_p = \frac{1}{(2 \rho h)} \]

\[ \mathcal{R}_s = \tilde{y} \left( \frac{3F}{2X} + \frac{3G}{2Y} + E \right) - (A_1 - v_s \rho w) \left[ v_{n \infty} + \left( \frac{c_r}{c} \right) u \right] y_z / (y_s v_s) \]

\[ \tilde{y} = (\tilde{y}_1, \tilde{y}_2, \tilde{y}_3, \tilde{y}_4) \]

\[ \tilde{y}_1 = [2 - (\mathcal{K}_1 / \rho) v^2] \lambda_1, \tilde{y}_2 = [(u - \left( \frac{c_r}{c} \right) v) - \lambda_1] / \omega + \lambda_1 \mathcal{K}_1 w / \rho \]

\[ \tilde{y}_3 = \lambda_1 \mathcal{K}_1 u / \rho - 1, \tilde{y}_4 = c_r / c + \lambda_1 \mathcal{K}_1 v / \rho \]

\[ \lambda_1 = - a^2 \left( \beta_0 w + [u - \left( \frac{c_r}{c} \right) v] \right) / (w^2 - a^2) \]

In the above, the quantity \( \tilde{y} \left( \frac{3F}{2X} + \frac{3G}{2Y} + E \right) \) denotes the inner product of these vectors. Note that for a perfect gas

\[ \mathcal{K}_1 = - \rho / h \quad \text{[perf]} \]

In the case of a more general gas where \( h = h(p, \rho) \) is given as a curve fit to thermodynamic data, \( \mathcal{K}_1 \) is determined numerically using central differences.

The quantities \( c, c_r, \) and \( c_z \) are advanced using predictor-corrector methods in the form:
where in both the predictor and corrector steps the derivatives 
\( \frac{\partial c}{\partial Z} \), \( \frac{\partial \phi}{\partial Z} \), and \( \frac{\partial c}{\partial Z} \) are determined using eqs. (3.9a) - (3.9c). In the predictor step, (3.9a) - (3.9c) are evaluated at \((Z^k, 1, Y_m)\) with the Y-derivatives approximated by 

\[
\left( \frac{\partial C}{\partial Y} \right)_{N,m} \approx \frac{\left( c^k \right)_{N,m+1} - \left( c^k \right)_{N,m}}{\Delta Y}
\]

and the X derivative, \( \frac{\partial F}{\partial X} \), approximated by 

\[
\left( \frac{\partial F}{\partial X} \right)_{N,m} \approx \frac{F^k_{N,m} - F^k_{N-1,m}}{\Delta X}
\]

In the corrector step, (3.9a) - (3.9c) are evaluated at \((Z^k + \Delta Z, 1, Y_m)\) using the predicted values of the flow variables and shock geometry (corrected value of \( c \), see below) with the Y derivatives approximated by 

\[
\left( \frac{\partial C}{\partial Y} \right)_{N,m} \approx \frac{\left( c^r \right)_{N,m} - \left( c^r \right)_{N,m-1}}{\Delta Y}
\]
and \( \frac{\partial F}{\partial X} \) approximated by

\[
\frac{\partial F}{\partial X} \approx \frac{F_{N,m}^* - F_{N-1,m}^*}{\Delta X} + \frac{F_{N,m}^k - 2F_{N-1,m}^k + F_{N,m-2}^k}{\Delta X}
\]

The latter formula is used to achieve second order accuracy. The corrector for the shock shape \( c \) is performed before the corrector for \( c_\phi \), and \( c_Z \).

This allows the use of corrected values of \( c \) in the evaluation of \( \frac{\partial c_\phi}{\partial Z} \) and \( \frac{\partial c_Z}{\partial Z} \) in the corrector step and also in the Rankine-Hugoniot relations.

After advancing \( c, c_\phi, \) and \( c_Z \) in the predictor (or corrector), the predicted (or corrected) values of the flow variables at \( X = 1 \) are obtained from the Rankine-Hugoniot relations. These are rewritten for the purpose of the computation using the effective gamma defined by (3.7).

The result is

\[
p = \frac{1}{\Gamma + 1} \left[ p_\infty + c_\infty V_{n_\infty}^2 + \sqrt{(c_\infty V_{n_\infty}^2 - p_\infty \Gamma + \frac{2p_\infty V_{n_\infty}^2 (\Gamma - 1) (\Gamma - \Gamma')}{\Gamma - 1}} \right] \tag{3.12}
\]

\[
\rho = c_\infty^2 V_{n_\infty}^2 / (\rho_\infty V_{n_\infty}^2 + p_\infty - p)
\]

\[
u = u_\infty + \left( \frac{V_{n_\infty}}{\gamma} \right) \left( 1 - \frac{c_\phi}{\rho} \right), \quad w = w_\infty - (u - u_\infty) c_Z, \quad v = v_\infty - (u - u_\infty) (c_\phi/c)
\]

In the above, \( V_{n_\infty} \) is the free stream velocity component normal to the shock and \( w_\infty, v_\infty, u_\infty \) are the free stream velocity components given by \( V_\infty \cos \beta \cos \alpha, V_\infty (\cos \beta \sin \alpha + \sin \beta \cos \phi) \), and \( -V_\infty (\cos \beta \sin \alpha \cos \phi + \sin \beta \sin \phi) \), respectively. For the case of a perfect gas, \( \Gamma = \Gamma_{\infty} = \gamma \), and (3.12) give directly the flow quantities at \( X = 1 \) since \( V_{n_\infty} \) is known when \( c, c_\phi, \) and \( c_Z \) are determined. For a more general gas where \( h = h(p, \phi) \)}
is given as a curve fit, the Rankine-Hugoniot relation must be solved numerically. This is done by solving the nonlinear equation

\[
h(p, \rho) - h_\infty - \frac{1}{2} (V_n^2 - V_\infty^2) = 0
\]

for \( V_n \). Here \( h(p, \rho) \) is the value of \( h \) from the curve fits with \( p \) and \( \rho \) defined using the first and third equations in (2.7). This equation is solved iteratively using the secant method. To start the iteration, the required two estimates of \( V_n \) are obtained using

\[
V_n = \rho_\infty \frac{V_\infty}{\rho}.
\]

For the first estimate, \( \rho \) is the value obtained from (3.12) with \( \Gamma \) the previous value in the predictor step and the predicted value in the corrector step. For the second estimate, the value of \( \Gamma \) resulting from the first estimate is used.
3.4 Body Surface Points

At the body surface, \( X = 0 \) (\( n = 1 \)), the boundary condition (2.6) must be satisfied. Our computational method for the points on \( X = 0 \) utilizes a special system of equations which is valid only on the body surface. These auxiliary equations are developed in Appendix A (see (A-10), (A-11), and (A-13) of Appendix A). Before giving the completely expanded forms of these equations, we will discuss the basic computational method.

The basic variables on \( X = 0 \) are the natural log of the pressure, \( P = \log p \), the quantity \( V_2 = v + (b_0/b)u \), and the entropy, \( s \). These quantities are advanced on \( X = 0 \) using predictor-corrector methods in the form:

\[
\begin{align*}
p^*_{1,m} &= p^k_{1,m} + \Delta Z \left( \frac{1}{p} \frac{\partial p}{\partial Z} \right)^k_{1,m} \\
(V^2)^*_{1,m} &= (V^2)^k_{1,m} + \Delta Z \left( \frac{3}{2} \frac{\partial V_2}{\partial Z} \right)^k_{1,m} \\
s^*_{1,m} &= s^k_{1,m} + \Delta Z \left( \frac{\partial s}{\partial Z} \right)^k_{1,m}
\end{align*}
\] (predictor) \tag{3.13}

\[
\begin{align*}
p^{k+1}_{1,m} &= \frac{1}{2} \left[ p^k_{1,m} + p^*_{1,m} + \Delta Z \left( \frac{1}{p} \frac{\partial p}{\partial Z} \right)^*_{1,m} \right] \\
(V^{2})^{k+1}_{1,m} &= \frac{1}{2} \left[ (V^2)^k_{1,m} + (V^2)^*_{1,m} + \Delta Z \left( \frac{3}{2} \frac{\partial V_2}{\partial Z} \right)^*_{1,m} \right] \\
s^{k+1}_{1,m} &= \frac{1}{2} \left[ s^k_{1,m} + s^*_{1,m} + \Delta Z \left( \frac{\partial s}{\partial Z} \right)^*_{1,m} \right]
\end{align*}
\] (corrector) \tag{3.14}

where in both the predictor and corrector steps the derivatives \( \frac{\partial p}{\partial Z}, \frac{\partial V_2}{\partial Z}, \) and \( \frac{\partial s}{\partial Z} \) are determined from eqs. (A-10), (A-11), and (A-13).
(The completely expanded form of these equations will be presented below in this section). After both the predictor and corrector steps, the flow variables $u, v, w, p$ are determined as follows. From the computed value of $p$, we obtain the pressure; i.e., $p = \exp(P)$. Using thermodynamic relations, we obtain $\rho$ and $h$ from $p$ and $s$. It follows from (2.2) that

$$v^2 = u^2 + w^2 + v^2 = 2(H - h)$$ and from (2.6), we have

$$w = \sqrt{[1+(b_y/b)^2]v^2 - V_2^2} / v_w,$$

$$v_w^2 = 1 + b_z^2 + (h_y/h)^2$$

$$v = [V_2 - (b_y/b)h_w]/[1 + (b_y/b)^2],$$

$$u = h_z w + (b_y/b)v$$

Note, the boundary condition (2.6) is satisfied by both predicted and corrected values of the flow variables.

The above formulation, of identifying the variables $P$, $V_2$, and $s$ as the variables to be advanced at the boundary, provides a form of the equations which allows one to maintain an accurate approximation through regions where the flow nonuniformity develops and remains adjacent to the body surface (e.g., entropy layers on blunted slender bodies). Other formulations which utilize finite differencing in the vicinity of such flow non-uniformities characteristically are differencing nonsmooth flow quantities and hence require some local modifications to maintain the calculation. However, our formulation allows one to advance the
solution by the differencing of only smooth quantities. In order to see why this is so, first observe that, since such nonuniformities lie along the body surface which is a stream surface, they have the character of a contact discontinuity. It is known (c.f., ref. 4, p. 317-18) that across such a discontinuity only the pressure and the normal component of velocity can never experience jumps. Now consider the present wall point computation. The only X differences required are in (A-10), the equation for pressure. The quantities $V_z$ and $s$ are advanced using only quantities defined on the wall. Further, the only quantities differenced in the X direction are $p$ and $A$. But, $p$ is smooth across the nonuniformity and $A$ at the wall is, except for a factor involving geometry, the normal component of velocity which is also smooth across the nonuniformity.

Hence, the wall point calculation remains well behaved. Consider now the calculation for the adjacent interior point, $X = \Delta X$. In either the predictor or corrector (depending on $j=0$ or 1) an X difference of $F$ must be taken across the nonuniformity. But at $X = 0$, $F$ involves only the pressure and geometry (since $A = 0$ at the wall in both the predictor and the corrector). Thus, in the calculation at $X = \Delta X$, the X difference of $F$ is unaffected by the nonuniformity.

The formulas used for determining $\frac{3p}{3Z}$, $\frac{3V_z}{3Z}$, and $\frac{3s}{3Z}$ in (3.13) and (3.14) are (A-10), (A-11), and (A-13), respectively. The actual formulas used in the code are obtained by expanding these equations using the boundary condition (2.6) and noting that at $X = 0$, $X_z = -b_z X_\phi$ and $X_\phi = -b_\phi X_z$.

From (A-10), we obtain

$$\frac{3p}{3Z} = X_\lambda \frac{2p}{3X} - \frac{1}{b_1} \left\{ \omega \left( \lambda \frac{3A}{3X} - (a_\lambda w + a_\lambda v) \right) + \mathcal{P} \right\} (3.16)$$
where

\[ \mathcal{P} = \frac{\partial \mathbf{V}}{\partial b} + w_{\Phi} + \varepsilon_1 + \xi \frac{\partial \mathcal{G}}{\partial Y} + p \left[ \xi_2 (T \Phi, Y \Phi, T \Phi, Y \Phi) + \frac{1}{b^3} + \xi_4 (T \Phi, Y \Phi, b \Phi / b) \right] \]

\[ \lambda_+ = \frac{a^2 (\beta - \frac{b}{z})}{\frac{w^2}{a^2} - a^2}, \quad \beta_1 = \sqrt{\frac{w^2}{a^2} + \left[ 1 + \left( \frac{b \Phi}{b} \right)^2 \right]} \]

\[ a_7 = \frac{\partial b}{\partial z} = b_{zz} - b_{z \Phi} z / Y \Phi \]

\[ a_4 = \frac{\partial (b \Phi / b)}{\partial z} = \frac{1}{b} \left[ b_{z \Phi} - \frac{b z \Phi}{b} - (b \Phi - b^2 / b) Y z / Y \Phi \right] \]

\[ \xi = (\xi_1, \xi_2, \xi_3, \xi_4) \]

\[ \xi_1 = w \lambda_+ (2 - \frac{2}{\kappa_1}) \]

\[ \xi_2 = b_{z \Phi} - \lambda_+ + \frac{w^2}{\kappa_1} \lambda_+ / \rho \]

\[ \xi_3 = w \lambda_+ \lambda_+ / \rho - 1 \]

\[ \xi_4 = w \lambda_+ \lambda_+ / \rho + b_{\Phi} / b \]

\[ \kappa_1 = \frac{\partial \rho}{\partial h} \]

In the above, the quantity \( \xi \cdot \frac{\partial \mathcal{G}}{\partial Y} \) denotes the inner product of these vectors.

Note that for a perfect gas \( \kappa_1 = - \rho / h \); for a more general gas \( \kappa_1 \) is determined numerically using central differences in the curve fits.

From (A-11), we obtain

\[ \frac{\partial \mathbf{V}}{\partial Z} = a_4 u + \mathbf{V} / \rho \mathbf{w} \] (3.17)
where

$$
\mathbf{V} = \eta \frac{3G}{\partial Y} - \rho v w z / b - \frac{D}{b} (T, Y \frac{\partial \phi}{\partial Y} + b \frac{\partial \phi}{\partial Y})
$$

(3.17a)

$$
\eta = (\eta_1, \eta_2, \eta_3, \eta_4)
$$

$$
\eta_1 = V_z, \eta_2 = 0, \eta_3 = -b / b, \eta_4 = -1
$$

Equation (A-13) remains

$$
\frac{\partial s}{\partial Z} = -B \frac{\partial s}{\partial \phi}.
$$

(3.18)

In the predictor step (i.e., (3.13)), the quantities $\frac{\partial p}{\partial Z}_{1,m}$, $\frac{\partial V_{z}}{\partial Z}_{1,m}$, and $\frac{\partial s}{\partial Z}_{1,m}$ are obtained by evaluating (3.16) - (3.18) at $(Z^k, 0, Y_m)$ with the X partial derivatives replaced by

$$
\left( \frac{\partial}{\partial X} \right)^k_{1,m} = \left( \frac{\partial}{\partial X} \right)_{1,m}^{2,m} - \left( \frac{\partial}{\partial X} \right)_{1,m}^2
$$

and the Y partial derivatives replaced by

$$
\left( \frac{\partial}{\partial Y} \right)^k_{1,m} = \left( \frac{\partial}{\partial Y} \right)_{1,m}^{1,m + 1} - \left( \frac{\partial}{\partial Y} \right)_{1,m}^{1,m}.
$$

For the corrector step (3.16) - (3.18) are evaluated at $(Z^k + \Delta Z, 0, Y_m)$ using predicted values of the flow variables with the X partial derivatives replaced by

$$
\left( \frac{\partial}{\partial X} \right)^*_{1,m} = \left( \frac{\partial}{\partial X} \right)_{1,m}^{2,m} - \left( \frac{\partial}{\partial X} \right)_{1,m}^2
$$

and the Y partial derivatives replaced by

$$
\left( \frac{\partial}{\partial Y} \right)^*_{1,m} = \left( \frac{\partial}{\partial Y} \right)_{1,m}^{1,m + 1} - \left( \frac{\partial}{\partial Y} \right)_{1,m}^{1,m - 1}.
$$

34
An alternative procedure for the wall point calculation can be obtained by rewriting (3.16a) and (3.17a) using the definitions of $G$ and $e_1$, (2.6) and (2.2), and the thermodynamic relation (A-12), i.e.,

$$
\mathcal{P} = \frac{\rho V}{b} V_2 + \left[ b_z \phi_z + \frac{1}{b_z} b_\phi \phi_\phi + \left( \frac{\rho w}{a} - Y_z \right) \lambda + \frac{\partial P}{\partial Y} \right]
$$

$$
- \rho B \left( a_5 \omega + a_3 \nu \right)
$$

(3.16b)

$$
+ \rho \omega \left( (T_f - T_b) \nu + \frac{V}{b} T_f \nu + \frac{w}{b} \left( \frac{Y_\phi}{b_\phi} \frac{\partial (v/w)}{\partial Y} + b_z \right) \right)
$$

where

$$
a_5 = \frac{\partial b_z}{\partial Y} = b_\phi / Y_\phi
$$

$$
a_3 = \frac{(b_\phi/b)}{\partial Y} = [b_\phi/b] - (b_\phi/b)^2 / Y_\phi
$$

and

$$
\nu = \rho B (a_5 \omega - \frac{\partial V_2}{\partial Y}) - \frac{1}{b} \frac{\partial P}{\partial \phi_\phi} - \frac{\partial w w}{b} b_z
$$

(3.17b)

We have found that the use of (3.16b) and (3.17b) instead of (3.16a) and (3.17a) can in certain instances yield significantly different numerical results. The use of (3.16a) and (3.17a) appears to provide better results on the lee-side of bodies at large angle of attack; in fact, in sphere-cone calculations the use of (3.16a) and (3.17a) yields numerical results at large angles of attack where the calculation using (3.16b) and (3.17b) fails. On the other hand, immediately downstream of discontinuities in body slope which produce large expansions, the use of (3.16b) and (3.17b) produces numerical results where the calculation using (3.16a) and (3.17a) fails. The reason for these differing numerical behaviors, using essentially
equivalent analytical formulas, is not understood at the present time. In the present code, both formulations are available; either form can be selected by the user. The scheme using (3.16b) and (3.17b) is automatically used by the code after certain discontinuities in body slope are encountered (see section 4.1 for details).

In both formulations, the X partial derivatives appearing in (3.16) are forward differenced in both the predictor and corrector steps. This makes the numerical solution at the wall formally first order accurate. Both formulations can be made formally second order accurate (Ref. 1) by adding a correction to the corrector formula for the pressure; viz.,

\[ p_{1,m}^{k+1} = \frac{1}{2} p_{1,m}^k + p_{1,m}^* + \Delta Z \left( \frac{1}{p} \frac{\partial^2 p}{\partial z^2} \right)_{1,m}^k \]

(3.19)

where

\[ \bar{p}_{1,m}^k = \left( \lambda_{1,m}^k \right) \left( X_{1,m}^k \right) \left( \frac{2p_{2,m}^k - p_{3,m}^k - p_{1,m}^k}{\Delta X} \right) \]

\[ - \left( \frac{\rho_w}{\beta_{1,m}^k} \right) \left( \frac{2A_{2,m}^k - A_{3,m}^k}{\Delta X} \right) \]

The term \( \bar{p}_{1,m}^k \) is computed in the predictor step, using values at \( (Z_{1,m}^k, 0, Y_{1,m}) \), and applied in the corrector step.

In the present code, second order accuracy is an option which can be selected by the user. However, care must be exercised in the use of this option. The higher accuracy can be achieved only when the computed flow field near the wall is sufficiently smooth. In certain instances,
specifically downstream of discontinuities in body slope and/or body curvature, numerical oscillations will occur in the computed wall pressure when (3.19) is used. The magnitude of these oscillations depends on the size of the discontinuity; the case of body slope discontinuities is more serious—many times causing the calculation to fail. The code automatically switches the second order accurate scheme off (if it was originally selected) when discontinuities in body slope are encountered (see section 4.1 for details). Note that when the only flow nonuniformity present is of the type lying along the wall (e.g., an entropy layer), the second order scheme can still be used. This follows because the only quantities which are differenced in the X direction are p and A which are, as we have previously indicated, smooth quantities up to the wall.

3.5 Symmetry and Periodic Boundary Points

The procedures given above for wall, interior, and shock points must be slightly modified at the boundary planes Y=0 and Y=1 (corresponding to m=1 and M, respectively) since some of the differences in the Y direction require quantities defined at Y = -\Delta Y and Y = 1 + \Delta Y. The required quantities are obtained using either symmetry or periodic conditions depending on whether the symmetric or the non-symmetric problem is being considered. Additional considerations come into play when clustering transformations are being used. For computational simplicity certain restrictions are placed on the mapping functions f(X,Y,Z) and g(Y,Z). These restrictions vary depending on whether the symmetric or the non-symmetric problem is being considered.
Consider first the symmetric problem. Here the planes $\phi=0$ and $\phi=\pi$ are boundaries where all flow variables are even functions of $\phi$ except $v$ which is an odd function of $\phi$. Since these conditions must be applied in the computational space $(X,Y,Z)$, it is natural to impose on the mapping function $f(X,Y,Z)$ the condition

$$f_y(X,0,Z) = f_y(X,1,Z) = 0 \quad (3.20)$$

Now, given $q_e$, an even function of $\phi$ about $\phi=0$ and $\pi$ in the physical plane, it can be shown using (3.1), (3.2), when (3.20) is satisfied, that in the computational plane

$$q_e(X,-\Delta Y,Z) = q_e(X,\Delta Y,Z) + O(\Delta Y^3)$$

$$q_e(X,1+\Delta Y,Z) = q_e(X,1-\Delta Y,Z) + O(\Delta Y^3) \quad (3.21)$$

further, if $q_o$ is an odd function of $\phi$ about 0 and $\pi$ then

$$q_o(X,-\Delta Y,Z) = -q_o(X,\Delta Y,Z)D(0,Z) + O(\Delta Y^4)$$

$$q_o(X,1+\Delta Y,Z) = -q_o(X,1-\Delta Y,Z)/D(1,Z) + O(\Delta Y^4) \quad (3.22)$$

where

$$D = \frac{2-(g_{YY}/g_Y)\Delta Y}{2+(g_{YY}/g_Y)\Delta Y}.$$

In the computation for the symmetric problem, the points on $Y=0$ and $Y = 1 + \Delta Y$ are now treated in essentially the same manner as discussed in the previous sections. Except for the quantities needed on the "fringe" planes $Y = - \Delta Y$ and $Y = 1 + \Delta Y$. These are determined using either (3.21) or (3.22). The derivatives of $g$ are determined by direct evaluation of their analytical definitions at $Y = - \Delta Y$ and $1 + \Delta Y$. Thus, it is required

*The derivation of these expressions is carried out in Appendix B.
that \( g(Y,Z) \) be defined (and twice continuously differentiable) for

\(-\Delta Y \leq Y \leq 1+\Delta Y\). In both the predictor and corrector steps the boundary

conditions

\[ v(X,0,Z) = v(X,1,Z) = v_2(0,Z) = v_2(1,Z) = 0 \] (3.23)

are imposed.

Consider now the non-symmetric problem; i.e., all quantities are periodic functions of \( \phi \) with period \( 2\pi \) in the physical plane. It is convenient for computational purposes not to destroy this periodicity in the computational space; i.e., to have all quantities periodic functions of \( Y \) with period 1. This can be achieved by requiring that the mapping function \( f(X,Y,Z) \) be periodic in \( Y \) with period 1 and that the mapping function \( g(Y,Z) \) be such that \([g(Y,Z)-Y]\) is a periodic function of \( Y \) with period 1. With these conditions on \( f(X,Y,Z) \) and \( g(Y,Z) \), all quantities are periodic in the computational space. The calculation is now performed for only \( M-1 \) planes \( (m=1,2,\cdots,M-1) \) corresponding to \( 0 \leq Y \leq 1-\Delta Y \). For \( Y=0 \) \((m=1)\) and \( Y=1-\Delta Y \) \((m=M-1)\) it will be required, in either the predictor or corrector, to provide quantities on \( Y=-\Delta Y \) and \( Y=1 \), respectively. These are determined by the periodicity conditions

\[
q(X,-\Delta Y,Z) = q(X,1-\Delta Y,Z)
\]

and

\[
q(X,1,Z) = q(X,0,Z)
\] (3.24)

where \( q \) is any quantity.

3.6 **Step Size and Stability**

The step size \( \Delta Z \) is chosen in accordance with the CFL condition (which is a necessary condition for numerical stability). This condition
is that the domain of dependence of the partial differential equations must be contained in the domain of dependence of the finite difference equations at all points. The computation of $\Delta Z$ based on this condition is carried out in Appendix C. We give only the final results here. Note that two finite difference schemes are available in the code (corresponding to $j=0$ or $1$ in (3.6a) and (3.6b)) and the CFL condition for each scheme is different. The CFL conditions for the present code are given by

$$
\Delta Z \leq \Delta X \min \left\{ \frac{\sqrt{\frac{v^2 - a^2}{\mu}}} \right\}
$$

where the minimum is taken over all computational points. At each point, $\mu$ is defined by

$$
\mu = \max(\mu_1, \mu_2, \mu_3)
$$

where

$$
\mu_1 = |wA - a^2X_z| + a \sqrt{(w^2 - a^2)(X_r^2 + X^2/\tau^2) + (A - wX_z)^2}
$$

$$
\mu_2 = |\delta_p (wB - a^2Y_z)| + a \sqrt{(w^2 + v^2 - a^2)(Y^2_\phi/\tau^2)}
$$

$$
\mu_3 = |wA - a^2X_z - \delta (wB - a^2Y_z)| + a \sqrt{(w^2 - a^2)[X^2_r + \frac{1}{\tau^2}(X^2 - \delta Y^2_\phi)] + (wX_z - a^2Y_z/\tau^2)}
$$

and

$$
\delta = \begin{cases} 
\Delta X/\Delta Y, & \text{if } j=0 \\
-\Delta X/\Delta Y, & \text{if } j=1
\end{cases}
$$

In the code, $\Delta Z$ is computed using

$$
\Delta Z = C \Delta X \min \left\{ \frac{\sqrt{\frac{v^2 - a^2}{\mu}}} \right\}
$$

(3.25)

where $0 < C \leq 1$ is a numerical constant which can be selected by the user. Certain adjustments in $C$ can be made when discontinuities in body slope are encountered (see section 4.1 for details).
4. SPECIAL FEATURES

4.1 Discontinuities in Body Slope

Many important flow field applications involve body shapes given by functions \( b(\xi, \zeta) \) that contain discontinuous slopes \( b_\xi \) and \( b_\zeta \), e.g., bodies which have slices and/or flaps and biconics or other segmented body shapes. It is well known that when such discontinuities appear certain discontinuities in the flow variables at the surface are produced. Furthermore, these flow discontinuities generally propagate into the flow field as either shock waves or expansion fans. Numerically, such discontinuities violate the assumptions which underlie the basic numerical algorithms. Computing the flow field by marching through these discontinuities without any special provisions produces numerical oscillations downstream of the discontinuity in both the flow variables on the body surface and in the interior of the flow. When the discontinuities are large, these oscillations can result in a program halt or, at the very least, a substantial region of unrealistic results.

The present code incorporates special provisions which are used when certain discontinuities in body slope are encountered. The code approximately locates these body discontinuities and then computes the associated flow discontinuities in the surface flow variables based on a local analysis of the discontinuity. We have found that this procedure gives, in most cases, far better results than would be obtained by simply marching through the discontinuity. The present procedure, however, is not by itself a complete solution to all the problems associated with body slope discontinuities. Even though the scheme for interior points is of the "shock capturing" type, numerical oscillations in the interior, downstream of a body
discontinuity, can develop when the discontinuity produces large expansion or compression discontinuities. This is a defect in the MacCormack scheme. In certain instances some numerical oscillations also occur along the wall immediately downstream of the body slope discontinuity. A consistent method for computing the body surface points immediately downstream of discontinuities of this type is not known at the present time. At the end of this section, after the discussion of the procedure for obtaining discontinuities in the surface flow variables, various techniques available in the code for improving the downstream calculation will be discussed.

For complicated geometries (e.g., bodies with slices), the precise location of body slope discontinuities would require a considerable amount of additional logic and computations. For this reason, in the present code the discontinuities are only located approximately; i.e., within at most ΔZ. The procedure is as follows. At every step \( Z^{k+1} = Z^k + ΔZ \), the values of \( b_z \) and \( b_ϕ \) are compared to their previous values (at \( Z^k \)) using

\[
|\langle b_z \rangle_m^k - \langle b_z \rangle_m^{k+1}| - ΔZ \max \{|\langle b_{zz} \rangle_m^k|, |\langle b_z \rangle_m^{k+1}|\} > \varepsilon
\]

(4.1)

\[
|\langle b_ϕ \rangle_m^k - \langle b_ϕ \rangle_m^{k+1}| - ΔZ \max \{|\langle b_{zz} \rangle_m^k|, |\langle b_z \rangle_m^{k+1}|\} > \varepsilon
\]

(4.2)

where \( \varepsilon \) is a small positive number (we use \( \varepsilon = 10^{-6} \)).

When either of the above inequalities is satisfied, the code assumes that a discontinuity in \( b_z \) and/or \( b_ϕ \) exists between \( Z^k \) and \( Z^{k+1} \) at \( Y = Y_m \).

At each such value of \( Y \), the body geometry is temporarily modified* at \( Z^{k+1} \) by putting the body shape derivatives \( b_z, b_ϕ, b_{zz}, b_{zϕ}, \) and \( b_{ϕϕ} \)

*The body that is in effect created at \( Z = Z^{k+1} \) by this convention will be referred to as the "modified body."
equal to their values at $Z^k$. This temporarily removes the discontinuities in the body slopes and the predictor-corrector sequence for the "modified body" is performed for the step $Z^{k+1}$ using the basic scheme as described in section 3. After the corrector step, the body derivatives which were modified are put equal to their true values at $Z^{k+1}$. Also, for each value of $Y$ where discontinuities were sensed, the flow quantities at the body surface are changed by applying a local analysis to determine the appropriate flow discontinuities as if the body slope discontinuity were located at $Z = Z^{k+1}$. Note that this special procedure is only applied to discontinuities of $b_z$ and $b_\phi$ in the $Z$ direction. No special consideration is given to other discontinuities in body shape (e.g. discontinuities in $b_\phi$ for a fixed $Z$).

Let us now turn to the procedure for determining the appropriate flow discontinuities at the wall points $(0, Y, Z^{k+1})$ where according to the above procedure discontinuities in body slope have been effectively placed. The analysis is conveniently carried out in the physical space $(r, \phi, z)$. Let $(r_o, \phi_o, z_o)^+$ where $r_o = b(\phi_o, z_o)$ be the body surface point corresponding to $(0, Y, Z^{k+1})$. We will denote the values of quantities associated with the "modified body" geometry by the subscript $-$ and the values of quantities associated with the true body geometry by the subscript $+$. The modified body geometry will be referred to as the upstream side of $(r_o, \phi_o, z_o)$; the true body geometry will be referred to as the downstream side of $(r_o, \phi_o, z_o)^+$. All flow quantities on the upstream side of $(r_o, \phi_o, z_o)$ are the computed

*This terminology is motivated by the fact that the flow on the surface of the modified body is in the direction of $(r_o, \phi_o, z_o)^+$.

†The symbol $\phi_o$ used in this section bears no connection to its use in other sections of the report.
values at this point using the modified geometry and thus are known. The
problem is therefore to determine the flow quantities on the downstream side
of \((r_o, \phi_o, z_o)\) given the upstream quantities and the body slopes on both sides.

Because \(b_z\) and/or \(b_\phi\) have different values on the upstream and downstream
sides of \((r_o, \phi_o, z_o)\), there are two distinct body normal vectors at this point;
i.e.,

\[
\hat{n}_+ = -(b_\phi/b_r) \hat{e}_r + \hat{e}_\phi - (b_z) \hat{e}_z
\]  

The situation is depicted in Fig. 4. The vector \(\hat{\tau} = \hat{n}_+ \times \hat{n}_-\) is tangent
to the edge of the discontinuity at \((r_o, \phi_o, z_o)\); c.f., Fig. 4. We
introduce the vectors \(\hat{\sigma}_+ = \hat{n}_+ \times \hat{\tau}\) and consider the two sets of mutually per-
pendicular unit vectors, one for each side of \((r_o, \phi_o, z_o)\), given by

\[
(\hat{e}_n)_+ = \hat{n}_+ / |\hat{n}_+|, \quad \hat{e}_\tau = \hat{\tau} / |\hat{\tau}|, \quad (\hat{e}_\sigma)_+ = \hat{\sigma}_+ / |\hat{\sigma}_+|.
\]  

With respect to the above notation, the basic conditions used to determine the
flow quantities on the downstream side of \((r_o, \phi_o, z_o)\) are:

i.) \(\hat{V}_- \cdot \hat{n}_- = 0\); i.e., there is no change in the velocity component
tangent to the edge.

ii.) The quantities \(p_+, \rho_+, \) and \([\hat{V}_+ - (\hat{V}_+ \cdot \hat{e}_\tau) \hat{e}_\tau]\) are to be determined
so that the inviscid boundary condition \(\hat{V}_+ \cdot \hat{n}_+ = 0\) is satisfied.

Note that, by computation, the upstream flow satisfies the boundary condition
\(\hat{V}_- \cdot \hat{n}_- = 0\). Also, using i.) and applying the condition \(\hat{V}_+ \cdot \hat{n}_+ = 0\), we obtain

\[
\hat{V}_+ = (\hat{V}_- \cdot \hat{e}_\tau) \hat{e}_\tau + V_{\sigma_+} \hat{e}_{\sigma_+}
\]  

where \(V_{\sigma_+} = (\hat{V}_+ \cdot \hat{e}_{\sigma_+})\) is the only unknown quantity.

The actual procedure for carrying out ii.) depends on the value of
\(V_{\sigma_-} = \hat{V}_- \cdot \hat{e}_{\sigma_-}\), the known upstream component of the surface velocity normal.
Fig. 4. Orthogonal directions used in the analysis of body slope discontinuities.
to the edge. From standard vector identities and $\vec{V} \cdot \hat{n}_- = 0$, it follows that

$$V_{\sigma_-} = \frac{\hat{n}_- \cdot (\vec{V}_- \cdot \hat{n}_+)}{\sqrt{|\hat{n}_-|^2 |\hat{n}_+|^2 - (\hat{n}_- \cdot \hat{n}_+)^2}}$$

(4.5)

When $|V_{\sigma_-}| < a_-$, the flow across the edge is subsonic. In this case, a heuristic procedure is used; namely, the velocity vector is rotated to satisfy $\vec{V}_+ \cdot \hat{n}_+ = 0$ assuming there are no discontinuities in $|\vec{V}|$, $p$, and $\rho$; i.e.,

$$p_+ = p_-, \quad \rho_+ = \rho_-, \quad \text{and} \quad V_{\sigma_+} = V_{\sigma_-}.$$  

(4.6)

When $|V_{\sigma_-}| \geq a_-$, the flow across the edge is supersonic. This case has been analyzed by Prandtl and Meyer in a classical investigation (see, e.g., ref. 4, chapter XI). This analysis indicates that the quantities $p_+, \rho_+$, and $V_{\sigma_+}$ can be determined using classical two-dimensional supersonic turning relations in the plane determined by $\hat{n}_+$ and $\hat{n}_-$. The angle through which the flow is turned is given by

$$\theta = \cos^{-1}\left(\frac{(\hat{n}_+ \cdot \hat{n}_-)}{|\hat{n}_+||\hat{n}_-|}\right)$$

(4.7)

There are two cases; viz., $V_{\sigma_-} > 0$ and $V_{\sigma_-} < 0$. When $V_{\sigma_-} > 0$, it follows from (4.5) that $(\vec{V}_- \cdot \hat{n}_+) > 0$ and thus the flow is turned by a centered expansion fan attached to the edge (see, Fig. 5a). When $V_{\sigma_-} < 0$, it follows that $(\vec{V}_- \cdot \hat{n}_+) < 0$ and the flow is turned by an oblique shock wave attached to the edge (see, Fig. 5b). In the latter case, it can happen that conditions are such that no oblique shock wave exists; e.g., this can occur when $\theta$ is large and/or $(V_{\sigma_-}/a_-)^2$ is near one. In this situation, the Prandtl-Meyer analysis breaks down and a heuristic procedure is used. This will be discussed later in this section.
Fig. 5. Flow discontinuities in the case \( V_0^2 > a^2 \).
In the case when $V_{\sigma-} > 0$, $p_+$, $\rho_+$, and $V_{\sigma+}$ are obtained from the equations for a centered expansion written in the following differential form:

\[
\frac{dp}{da} = -\rho q^2 / \sqrt{q^2/a^2 - 1}
\]

where

\[
q^2 = 2H_{\infty} - (\frac{V_{\sigma+}}{a})^2 - 2h
\]

\[
p = \rho(a_-, p), h = h(p, \rho), a^2 = a^2(p, \rho)
\]

\[
s_+ = s(p_-, \rho_+)
\]

In the code (4.8) is numerically integrated on the interval $0 \leq a \leq \theta$
(where $\theta$ is given by (4.7)) subject to the initial condition $p = p_-$ at $a = 0$ using the second order Improved-Euler method with 18 subdivisions.

The final results at $a = \theta$ give $p_+ = p_{\alpha=\theta}$, $V_{\sigma+} = q_{\alpha=\theta}$, $p_+ = \rho_{\alpha=\theta}$.

In the case when $V_{\sigma-} < 0$, $p_+$, $\rho_+$, and $V_{\sigma+}$ are obtained using the oblique shock relations. Consider first the case of a perfect gas. In this case, the shock relations are given in ref. 5 (pp. 9-10); they are reproduced here for the sake of completeness. Let $\mathbf{\mathbf{\mathcal{M}}}^2 = (V_{\sigma-}/a_-)^2$ and solve

\[
\chi^3 + c_1 \chi^2 + c_2 \chi + c_3 = 0
\]

\[
c_1 = -(1 + 2/m^2) - \gamma \sin^2 \theta
\]

\[
c_2 = (2\mathbf{\mathbf{\mathcal{M}}}^2 + 1)/\mathbf{\mathbf{\mathcal{M}}} + [(\gamma + 1)/4 + (\gamma - 1)/\mathbf{\mathbf{\mathcal{M}}}^2] \sin^2 \theta
\]

\[
c_3 = \cos^2 \theta / \mathbf{\mathbf{\mathcal{M}}}^4
\]

for the middle root (in the above, $\theta$ is the turning angle, see (4.7)).

This value of $\chi$ is the sine-squared of the shock angle. We then have

\[
\text{(4.9)}
\]
\[ p_+ / p_- = \frac{2\gamma \bar{m}^2 X - (\gamma - 1)}{(\gamma + 1)} \]
\[ \rho_+ / \rho_- = \frac{\gamma + 1}{(\gamma - 1) \bar{m}^2 X + 2} \]
\[ V_{\alpha_+} / V_{\alpha_-} = \sqrt{1 - 4(\gamma \bar{m}^2 X - 1)(\gamma \bar{m}^2 X + 1)} / [(\gamma + 1) \bar{m}^4 X] \]

As we pointed out earlier, (4.9) may not yield a root which is physically meaningful (i.e., one that increases entropy) for certain values of \( \bar{m} \) and \( \theta \).

This occurs when
\[ (2c_1^3 - 9c_2 c_3 + 27c_3^2)^2 + 12(3c_2^2 - c_2^3) > 0. \]

In this special case, a heuristic procedure is used; namely, a normal shock wave (i.e., normal to \( \bar{\alpha} \)) is assumed to exist on the upstream side in front of \((r_0, \phi_0, z_0)\). The flow quantities behind this normal shock are obtained using (4.10) with \( X = 1 \). The final results for \( p_+ \), \( \rho_+ \), and \( V_{\alpha+} \) are obtained by performing an isentropic expansion from the normal shock values to the turning angle \( \theta \) using a modified Newtonian pressure law; i.e.,

\[ p_+ = \max \{ p_-, p_{ns} \sin^2 \theta \} \]
\[ s_+ = s(p_{ns}, \rho_{ns}), h_{ns} = h(p_{ns}, \rho_{ns}) \]
\[ \rho_+ = \rho(s_+, p_+), h_+ = h(p_+, p_+ \rho_+) \]
\[ V_{\alpha+} = -\sqrt{(V_{\alpha_+})^2_{ns} + 2(h_{ns} - h_+)} \]

where the subscript ns indicates the values obtained from (4.10) with \( X = 1 \).
In the case $V_{\sigma-} < 0$ for a real gas, the oblique shock must be determined by iteration. The procedure used is to satisfy the energy equation

$$u_-^2 + 2h(p_-, p_-) = u_+^2 + 2h(p_+, p_+)$$

where

$$u_- = \sin \theta V_{\sigma-}, \quad u_+ = \cos \theta V_{\sigma-} \frac{(1 - \tan \theta \cot \theta)}{(\tan \theta + \cot \theta)}$$

$$\rho_+ = \rho_- \frac{u_-}{u_+}, \quad p_+ = \rho_- \frac{u_-}{u_+} (u_- - u_+) + p_-$$

and $\theta$ is the shock angle (to be determined). Here $\hat{u}_-$ and $\hat{u}_+$ are the velocity components normal to the shock. The algorithm is initiated by taking $\sin \theta > \max(\sin \theta, 1/\mu)$; the value of $\sin \theta$ is then increased until the above equation is satisfied (if an increasing entropy solution is available). If such a solution does not exist the heuristic procedure described above for the perfect gas case is used with $\gamma = \Gamma_-$. 

As we have already pointed out, the calculation downstream of body slope discontinuities is adversely affected by the resulting discontinuities in the surface flow variables. In order to minimize, these effects certain optional numerical procedures should not be used downstream of the discontinuity. These are:
1.) the second order accuracy option for the wall point calculation (see, sec. 3.4)

ii.) the wall entropy reduction option (see, sec. 4.2) when the discontinuity produces an oblique shock wave

Furthermore, we have found from computational experience that the version of the body surface point calculation using (3.16b) and (3.17b) gives the best results downstream of a discontinuity. Therefore, after a discontinuity in body slope has been encountered at say, $Y_m$, the code automatically makes the above modifications for the remainder of the calculation on $Y = Y_m$.

When discontinuities are encountered which produce large expansion discontinuities at the surface, the calculation for interior points downstream of the discontinuity becomes ill-behaved. Numerical oscillations appear in the conservation vector $U$ which can produce negative pressure at isolated points in the interior flow field. For such situations, the code has incorporated a selective (local) smoothing scheme*. If the pressure in either the predictor or corrector step is negative at an interior point $(X_n^k, Y_m^k, Z^k)$ then the conservation vector is redefined using

$$
\tilde{U}_{n,m}^k = (U_{n+1,m}^k + \theta U_{n,m}^k + U_{n-1,m}^k)/(\theta + 2) \quad (4.10.1)
$$

where $\theta$ is a non-negative integer which is chosen by the user. In the calculation, $U_{n,m}^k$ is replaced by $\tilde{U}_{n,m}^k$ and the flow variables at $(X_n^k, Y_m^k, Z^k)$ are redefined accordingly. In some instances, the computed results are improved after large expansion discontinuities by reducing the step size. The code includes an option which automatically does this after an expansion discontinuity.

*The application of such smoothing techniques can be shown to yield first order accurate approximations for smooth flow problems.
by changing the step size to ΔZ/I where I is an integer greater than or equal to one which can be chosen by the user. The calculation for interior points downstream of compression discontinuities is better behaved than when expansion discontinuities are present and no special techniques need be applied at interior points in the case of compressions.

Immediately downstream of large compression or expansion discontinuities the wall point calculation is ill-behaved. We have found in computations for planar compression and expansion ramps and axially-symmetric bodies that the region of poor surface results extends from the discontinuity downstream to a point on the body where the shock wave propagates into the interior flow past the first interior point (corresponding to n = 2). Based on these observations, the difficulty appears to be that in the wall point calculation the differences in the X direction must be taken across the flow discontinuity until it propagates across the first interior point. At the present time no rational approach for computing the wall points in this region which is valid for general three dimensional flows is known.

We have found that the following heuristic procedure improves the numerical results at the wall in this region. After a discontinuity in body slope has been encountered, which produces a pressure discontinuity at say, \( Y_m' \), the wall point calculation at \( Y_m \) for subsequent steps is performed while the terms in (3.16) which contain X-derivatives are multiplied by a factor. This factor increases smoothly from zero at the edge to unity after a fixed number of downstream marching steps.
The user can select the number of marching steps for the code to use this procedure. (When this number is put equal to zero the procedure is not used.)

In the calculation of complex body shapes (e.g., bodies with slices and finite span flaps), there can be planes \( Z = \text{constant} \) where some wall points have expansion discontinuities, some have compression discontinuities, and some have no discontinuities. The code logic has been written to account for such situations with regard to both the determination of the discontinuities at the body surface and the special numerical procedures used downstream of the discontinuities.

4.2 Wall Entropy Reduction

In the calculation of blunted smooth body shapes, the basic scheme will maintain the wall entropy at the stagnation point value (see, (3.13), (3.14) and (3.18)). This is the correct value of wall entropy corresponding to inviscid flow. In the case of a slender blunted body, the flow variables downstream of the nose develop large radial gradients at the wall (except for pressure and the normal component of velocity). The region containing these gradients decreases in thickness as the flow develops downstream. This inviscid phenomenon is known as the entropy (or vortical) layer. As we have previously pointed out (in section 3.4) the basic computational scheme is inherently capable of maintaining the correct inviscid value of wall entropy and performing the calculation when strong entropy layers are present without using special numerical techniques. In practical calculations of interest, the entropy layer cannot actually be resolved.
numerically because ultimately the layer's thickness will become smaller than the radial mesh spacing that one can afford to employ on any computer. When this happens, the present code computes a flow discontinuity at the body surface which corresponds to the variation of the flow variables across the entropy layer. This numerical discontinuity has the character of a contact discontinuity in that the pressure and normal component of velocity are continuous across it.

The development of the entropy layer and the corresponding numerical discontinuity produces at the body surface a flow which has a higher entropy and lower Mach number than the adjacent interior flow. This has at least two important effects on the subsequent calculation for the downstream flow field. First, the low axial Mach number at the surface causes a decrease in the step size $\Delta Z$ (see, (3.25). Thus, more computational steps are required and, hence, more computational time is required than if the wall entropy were at a level corresponding to the adjacent interior flow. Second, the lower speed flow on the wall can, in the presence of internal shock waves near the wall, become locally subsonic in the axial direction even though the adjacent interior flow remains supersonic. The appearance of axially subsonic flow at any point in the computation immediately halts the calculation.

It is therefore expedient in some cases to reduce the wall entropy and thereby reduce or eliminate the entropy layer. The procedure for performing this given by Kyriss and Harris (ref. 6) is incorporated as an option in the present code. The method is basically to define the wall entropy by linear extrapolation from the interior points rather than using (3.13), (3.14)
and (3.18). This procedure gradually reduces the wall entropy as the entropy layer develops. The actual algorithm is given by

\[
S^{k+1}_{1,m} = \begin{cases} 
2 \frac{S^k_{2,m} - S^k_{3,m}}{2(S^k_{2,m} + S^k_{3,m})} & \text{if } S^k_{2,m} \geq S^k_{3,m} \\
\frac{1}{2}(S^k_{2,m} + S^k_{3,m}) & \text{if } S^k_{2,m} < S^k_{3,m}
\end{cases}
\]

(4.11)

\[
S^{k+1}_{1,m} = \begin{cases} 
2 \frac{S^*_{2,m} - S^*_{3,m}}{2(S^*_{2,m} + S^*_{3,m})} & \text{if } S^*_{2,m} \geq S^*_{3,m} \\
\frac{1}{2}(S^*_{2,m} + S^*_{3,m}) & \text{if } S^*_{2,m} < S^*_{3,m}
\end{cases}
\]

(4.12)

The wall entropy is defined by (4.11) or (4.12) for \(1 \leq m \leq ms\) and by (3.13) or (3.14) and (3.18) for \(ms < m \leq M\) where \(ms\) is an integer \(0 \leq ms \leq M\) which is selected by the user. Note that when \(ms = 0\), the option is not used and when \(ms = M\), the option is used for all wall points. Kyriss and Harris use essentially this procedure only on the windward plane (\(ms = 1\), for the symmetric problem) and allow the lower entropy to be convected around the body using (3.18). We have found that \(ms = 2\) works better in the present code. Note that unless \(ms = M\), there will be no reduction of the wall entropy on the leeside in a symmetric problem (c.f., (3.18)).

Reducing the wall entropy by the above procedure substantially increases the step size and thus decreases computer run time without changing the surface pressure results for smooth body geometries. The other surface variables, however, are changed considerably. When discontinuities in body slope are encountered, the associated jumps in the surface flow variables (see, sec. 4.1) are significantly different depending on whether or not the wall entropy has been reduced. Further differences in the computed
surface pressure distribution develop for a region immediately downstream of the body slope discontinuity even though the option is turned off after a discontinuity in body slope is encountered. We have also found that in some calculations involving windward flaps that the use of this option resulted in successful calculations, whereas, the calculation not using the option failed due to the appearance of subsonic axial flow at some point on the flap.

It should be pointed out here that the above option is a purely heuristic procedure and there are many open questions associated with its use. No rational interpretation of the numerical results obtained using the option has been given—the results do not correspond to inviscid flow in the strict sense. At the present time it is not known a-priori when (or when not) to use this option in general circumstances. When new flow configurations are to be computed, we strongly recommend that where possible a few test cases should be computed both with and without the option and that the decisions on the use of the option should be based on a close examination of the results for physical consistency.
4.3 **Mesh Clustering**

In this section the use of the mesh clustering transformations in our code is discussed. As indicated in Section 3.1, clustering the mesh in the physical plane in either \( r \) or \( \phi \), or both directions, can be accomplished by appropriate choices of the mappings \( f(X,Y,Z) \) and \( g(Y,Z) \).

The primary purpose of mesh clustering is to distribute points in the physical plane \((r, \phi), Z = \text{const.}\) so that a region of large flow field gradients will contain relatively more computational points than a region where there are more gradual changes. The use of such techniques is fairly common, see refs. 3,7, but very rarely is it carefully described in terms of its overall efficiency and cost effectiveness.

Our preliminary experimentation with these techniques has revealed that a straightforward attempt at incorporating such mappings into flow field calculations may produce unexpected difficulties. For example, attempts at clustering mesh points in the \( r \) direction near the body surface to resolve the entropy layer were not successful. It is clear that if one attempts to fully resolve the types of entropy layers encountered on missile configurations that the transformations themselves would have to incorporate such large gradients that computationally the transformations could appear to be "discontinuous".

To illustrate some of the code requirements that must be satisfied an example is included. Consider the set of transformation functions for (3.2) given by

\[
\begin{align*}
\bar{z} &= Z \\
\bar{x} &= f(X,Y,Z) = \frac{(Z - z_0)x^1 + X}{(Z - z_0) + 1}, \quad Z \geq z_0 \\
\bar{y} &= g(Y,Z) = \frac{(Z - z_0)y^2 + Y}{(Z - z_0) + 1}, \quad Z \geq z_0
\end{align*}
\]

57
where \( j, l \) are integer exponents chosen suitably and \( z_0 \) represents the
Z location where the calculation is first started. The above transformation
has the effect that a uniform mesh at \( Z = z_0 \) is gradually, as \( Z \) increases,
evolved into a mesh with more radial points clustered about the body surface
and more planes clustered about the wind plane (\( \psi = 0 \)) in the physical
plane (see Fig. 6).

For \( f(X,Y,Z) \) and \( g(Y,Z) \) to be an admissible coordinate transformation
they must be one-to-one (invertible) mappings and twice continuously
differentiable with respect to \( X,Y,Z \) and satisfy
\[
\begin{align*}
  f(0,Y,Z) &= g(0,Z) = 0 \\
  f(1,Y,Z) &= g(1,Z) = 1
\end{align*}
\]  
(4.13)

For our particular example, the definition immediately reveals
that these conditions are all satisfied. Note in addition to the piecewise
analytical definition of \( f \) and \( g \) our code requires the following partial
derivatives be similarly given:
\[
\begin{align*}
  \{f_X, f_Y, f_Z, f_{XX}, f_{YY}, f_{ZZ}\} \\
  \{g_Y, g_Z, g_{YY}, g_{ZZ}\}
\end{align*}
\]

Returning to our above example, note that these partial derivatives
are easily determined. It is also easy to verify that our example is
one-to-one since one easily checks that \( f_X(X,Y,Z) > 0 \) and \( g_Y(Y,Z) > 0 \)
are satisfied. Furthermore, we observe that
\[
  f_X(0,Y,Z) = g_Y(0,Z) = \frac{1}{Z+1}
\]
indicating that for \( Z \gg 1 \) more points are clustered about the
surface and wind plane (see Fig. 6).
Fig. 6. The clustering function $f$ given in (4.13) for $j = 2$, $z_0 = 1$, and $Z = 10$. 
Finally an additional condition is imposed at the boundary. To apply the transformation to a symmetric problem it is required that

\[ f_y(X,0,Z) = f_y(X,1,Z) = 0 \]

For the above example this is clearly satisfied. On the other hand, the above transformation would not be suitable for a non-symmetric problem. For in this case it is required that

\[ f(X,Y+1,Z) = f(X,Y,Z) \]

Whereas the former condition is clearly satisfied, it is easy to see that no choice of exponents \( \ell \) in our above example will make \( g(Y,Z) - Y \) satisfy the periodicity condition.

In our early studies we attempted to systematically develop transformations from simple piecewise polynomial or rational functions. The use of polynomials would result in a saving on the functional evaluations required compared to using transcendental mapping functions (refs. 3,7). Our experiences with implementing a variety of polynomial interpolatory methods revealed that it was difficult to predict the parameters for a particular computation. Moreover, with the various analytical conditions to be satisfied (such as (4.14)) and the further desirability of having specific points included in the mesh, the analytical approach becomes overly burdened. For these, and other reasons stated below, we have adopted the approach of defining mesh transformations by a user-given set of discrete spatial points. The user must provide a set of points \( \{ \tilde{x}_n \} \) and/or \( \{ \phi_m = \tilde{y}_m \} \) and the code (see sections 12.5, 12.6) will assign the relationship

\[ \tilde{x}_n = f(X_n), \ n = 1,2,\ldots,N; \quad \tilde{y}_m = g(X_m), \ m = 1,2,\ldots,M. \]
Clearly one must select the points to satisfy (4.13) and whatever appropriate conditions such (4.14) may be required.

Note well, this implicit definition of the underlying transformation functions will be assumed to be smooth. If the data chosen does not arise from a context where smoothness is guaranteed, it is best to modify the choice to enforce some degree of smoothness. Our strategy has been to implement a "mesh pre-processor". All this entailed was a simple smoothing of the data points using equation (4.10.1) with $\Theta = 2$. This is repeated several times until the resulting differences appear smooth.

To approximate derivatives we use standard finite difference approximations from the given (or smoothed) data. For our purposes it is sufficient to approximate these derivatives to second order accuracy. For interior points $n = 2, \cdots, N-1, m = 2, \cdots, M-1$ we use standard centered differences

\[
(f_X)_n = \frac{f(X_{n+1}) - f(X_{n-1})}{2\Delta X}
\]

\[
(f_{XX})_n = \frac{f(X_{n+1}) - 2f(X_n) + f(X_{n-1})}{\Delta X^2}
\]

with similar expressions for $(g_Y)_m$ and $(g_{YY})_m$.

The only points requiring any explanation are the end points of the intervals. By second order extrapolation

\[
(f_{XX})_{n=1} = 2(f_{XX})_{n=2} - (f_{XX})_{n=3}
\]

\[
(f_{XX})_{n=N} = 2(f_{XX})_{n=N-1} - (f_{XX})_{n=N-2}
\]
A second order approximation to \((f_X^i)_{i=0}\) can be derived by assuming that \((f_{XX})\) is essentially linear on the intervals \((0, 2\Delta X)\) and \((1 - 2\Delta X, 1)\) then

\[
(f_X^i)_{n=1} = (f_X^i)_{n=2} - \frac{\Delta X}{2} ((f_{XX}^i)_{n=1} + (f_{XX}^i)_{n=2})
\]

(4.16a)

and

\[
(f_X^i)_{n=N} = (f_X^i)_{n=N-1} - \frac{\Delta X}{2} ((f_{XX}^i)_{n=N} + (f_{XX}^i)_{n=N-1})
\]

(4.16b)

For \(g(Y)\) a similar treatment is employed, for the symmetric case. Indeed at \(m = 1, M\) the approximations identical to (4.15) and (4.16) are made to \(g_{YY}\) and \(g_Y\), respectively. In the symmetric case it is further required to compute fringe planes \(Y = -\Delta Y\) and \(Y = 1 + \Delta Y\). Again the same approximations are used except now we assume \(g_{YY}\) is linear in the extended interval \((-\Delta Y, 2\Delta Y)\) and \((1-2\Delta Y, 1+\Delta Y)\). In the non-symmetric case, the data defining \(g(Y)\) is extended beyond \(0 \leq Y \leq 1\) using the periodicity conditions. The first and second derivatives at \(Y = 0, 1\) are determined in the same manner as at interior points.

Our experience has convinced us that the use of the above type of discrete definition of mesh transformations in conjunction with a preprocessor mesh smoother offers the widest flexibility for the user. One can easily create a local refinement of the mesh without much analytical care if one is prepared to allow a mesh smoother to alter your initial mesh array. Our experience has shown that a reasonable eyeball choice followed by several smoothings will only slightly alter the mesh and will retain the concentrations in the regions where originally desired.
5. FORCE AND MOMENT CALCULATIONS

The aerodynamic forces and moments acting on the vehicle are obtained by numerically integrating the computed surface pressure distributions. In the code, these calculations are performed in auxiliary subroutines so that the user may conveniently substitute alternative definitions and/or numerical integration procedures. In this section, the procedures and definitions currently available in the code are described.

The sign conventions for the components of the aerodynamic force and moment are illustrated in Fig. 7. The force and moment are computed assuming that the base pressure is $p_\infty$. Both the force and moment have dimensions of pressure. The center for the moments, $C$, is any point on the $z$-axis where, say, $z=z_c$ (see, Fig. 7). The derivative with respect to $z$ of the force components are given by

$$\frac{3F_a}{z} = \int\int (p_w - p_\infty) b \cdot \sin\phi \, d\phi,$$  

$$\frac{3F_n}{z} = \int\int (p_w - p_\infty) (b \cdot \cos\phi + b_z \cdot \sin\phi) \, d\phi,$$  

$$\frac{3F_v}{z} = \int\int (p_w - p_\infty) (b_z \cdot \cos\phi - b \cdot \sin\phi) \, d\phi.$$

The derivatives with respect to $z$ of the moment components taken about $C(z=z_c)$ are given by

$$\frac{3M_C}{z} = \int\int (p_w - p_\infty) b \cdot \sin\phi \, d\phi,$$

$$\frac{3M_n}{z} = \int\int (z_c - z) \frac{3F_v}{z} + \int\int (p_w - p_\infty) b_z^2 \cdot b \cdot \sin\phi \, d\phi.$$
Fig. 7. Definitions of aerodynamic force and moment components
\[
\frac{\partial^2 F}{\partial z^2} = (z-z_c) - n + \int_0^{2\pi} (p_w - p_\infty) b z^2 \cos \phi \, d\phi
\]  

(5.6)

In (5.1) - (5.6), \( p_w \) denotes the surface pressure. Note that for the symmetric problem

\[
\frac{\partial F}{\partial z} = \frac{\partial M^C}{\partial z} = \frac{\partial n}{\partial z} = 0.
\]

The integrals appearing in the above are computed numerically at each computational step, \( z = Z^k \), using Simpson's rule in the computational plane.

For the purpose of discussion, consider

\[
\psi = \int_0^{2\pi} \psi(\phi) \, d\phi
\]

which is, for a fixed value of \( z \), the general form of all the above integrals.

For the symmetric problem, the non-zero integrals are written in the computational plane as

\[
\psi = 2\int_0^1 (\psi/\psi_\phi) \, dY = 2\int_0^1 \psi(Y) \, dY.
\]

This is numerically integrated on the uniform computational mesh using Simpson's rule in the form

\[
\psi = \frac{2\Delta Y}{3} \left[ \psi_1 + 4\psi_2 + 2\psi_3 + 4\psi_4 + 2\psi_5 + \cdots \right]
\]

\[
\cdots + 2\psi_{M-2} + 4\psi_{M-1} + \psi_M, \text{ if } M \text{ is odd}
\]

and

\[
\psi = \frac{2\Delta Y}{3} \left[ \psi_1 + 4\psi_2 + 2\psi_3 + 4\psi_4 + 2\psi_5 + \cdots \right]
\]

\[
\cdots + 2\psi_{M-3} + 4\psi_{M-2} + \frac{1}{2} (5\psi_{M-1} + 3\psi_M), \text{ if } M \text{ is even}
\]

where \( \psi_m = \psi(Y_m) \), \( Y_M = 1 \). In the last expression, the trapezoidal rule is used for the subinterval \([Y_{M-1}, Y_M]\). In the nonsymmetric problem \( \psi \) is written as

65
\[ \psi = \int_0^1 (\psi/Y_\phi) dY = \int_0^1 \psi(Y) dY. \]

In this case, the integrands are periodic functions of Y with period 1 (i.e., \( \psi(Y_1) = \psi(Y_M) \)) and Simpson's rule becomes

\[ \psi = \frac{\Delta Y}{3} [2\psi_1 + 4\psi_2 + 2\psi_3 + 4\psi_4 + \cdots] \]

\[ \cdots + 2\psi_{M-2} + 4\psi_{M-1}], \text{ if } M \text{ is odd} \]

and

\[ \psi = \frac{\Delta Y}{3} [6\psi_1 + 3\psi_2 + 2\psi_3 + 4\psi_4 + 2\psi_5 + \cdots] \]

\[ \cdots + 2\psi_{M-3} + 4\psi_{M-2}], \text{ if } M \text{ is even}. \]

The code also computes, at each computational step, \( Z^k \), the force and moment vectors acting on the body truncated at \( z = Z^k \). These quantities are defined, for example, by

\[ F_a(Z^k) = \int_0^{Z^k} \frac{\partial F}{\partial z} dz \quad (5.7) \]

with similar expressions for the other truncated force and moment components. The integrals of the type (5.7) are evaluated numerically using the trapezoidal rule; i.e.,

\[ F_a(Z^{k+1}) = F_a(Z^k) + (\frac{Z^{k+1} - Z^k}{2}) \left[ \frac{\partial F}{\partial z} \right]_{z = Z^k} + (\frac{\partial F}{\partial z}) \left|_{z = Z^{k+1}} \right] \quad (5.8) \]

with similar expressions for the other force and moment coefficients.

Note that this calculation requires the force and moment on the body truncated at the initial plane \( z = Z_0 \). These quantities must be given along with the initial flow field data.

The final results are presented in coefficient form by dividing the force components and their derivatives by \( \frac{\rho_\infty V^2}{2} A_{ref} \) and the moment components and their derivatives by \( \frac{\rho_\infty V^2}{2} (A_{ref}) (z_{ref}) \) where \( A_{ref} \) is a reference area.
(non-dimensionalized by $R_o^2$). The parameters $z_c$, $z_{\text{ref}}$, and $A_{\text{ref}}$ can be selected by the user; however, if no selection is made, the code assumes that $z_c = 0$, $z_{\text{ref}}$ is the body length, and $A_{\text{ref}}$ is the base area of the baseline body (see, Fig. 6). The code also computes two centers of pressure. The center of pressure in pitch defined by

$$
(Z_{\text{c.p.}})_{p} = [z_c + M_c^p / F_n] \quad (5.9)
$$

for $F_n \neq 0$ and the center of pressure in yaw defined by

$$
(Z_{\text{c.p.}})_{y} = [z_c - M_c^y / F_n] \quad (5.10)
$$

for $F_y \neq 0$. 
DIFFERENTIAL EQUATIONS FOR BOUNDARY POINTS ON THE
BODY SURFACE AND THE BOW SHOCK WAVE

In this appendix differential equations are derived which are used for the computation at points on the boundaries \( X = 0 \) and \( 1 \). These equations are obtained from certain characteristic compatibility conditions evaluated on \( X = 0 \) and \( 1 \). Such equations have been used in various forms in many non-conservation codes (e.g., see refs. (6, 8, 9)). The equations used in the present code differ from those previously used. The advantages of the present equations are discussed in Secs. 3.3 and 3.4.

Since we are concerned with the boundary conditions on the planes \( X = 0 \) and \( 1 \), it is sufficient to consider the system (3.3) in the form

\[
\frac{3U}{3Z} + \frac{3F}{3X} = -\frac{3G}{3Y} - E = \mathcal{R}.
\]  

(A-1)

In this equation, the right hand side, \( \mathcal{R} \), will be treated as an inhomogeneous term since the Y derivative, being interior to \( X = 0 \) and \( 1 \), depends only on quantities on the boundary planes. From this point of view (A-1) is treated as a hyperbolic system in one space dimension \( X \) with \( Z \) time-like.

We have on the boundary plane, \( X = 0 \), the condition that \( u = b_z w + (b_y / b)v \).

On the boundary \( X = 1 \), the Rankine-Hugoniot conditions (2.7) are satisfied.

To motivate our treatment of the boundary points, consider as a prototype problem the mixed initial-boundary value problem on the strip \( Z \geq 0, \ 0 \leq X \leq 1 \) where initial data is given on \( Z = 0 \) and boundary conditions are given on \( X = 0 \) and \( 1 \). It is known (see, for example, ref. 10, pp. 471-475) from the theory of hyperbolic systems in one space dimension that, for a well posed mixed initial-boundary value problem, of this form, certain information concerning the solution at boundary points, say, \((0, Z^*)\) and \((1, Z^*)\) can be determined from the characteristic compatibility conditions.
corresponding to the characteristics through \((0, Z^*)\) and \((1, Z^*)\) which enter the region \(0 \leq X \leq 1, Z < Z^*\) (e.g., in Fig. A-1, the curves \(C_1\) and \(C_3\) through \((0, Z^*)\) and the curves \(C_2\) and \(C_4\) through \((1, Z^*)\)). The remaining information required to completely determine all the unknowns at \((0, Z^*)\) and \((1, Z^*)\) must be provided by the boundary conditions specified on \(X = 0\) and \(X = 1\). These boundary conditions are needed to replace the characteristics through \((0, Z^*)\) and \((1, Z^*)\) which do not enter the region \(0 \leq X \leq 1, Z < Z^*\), (e.g., \(C_2\) through \((0, Z^*)\) and \(C_1\) through \((1, Z^*)\) in Figure A-1). These characteristics are inadmissible since there is no information available for \(X < 0\) and \(X > 1\).

As we shall see in the following, at the boundaries \(X = 0\) and \(1\) the system \((A-1)\) has four independent characteristic compatibility equations. At \(X = 0\), there is one \(C_1\)-type characteristic, one \(C_2\)-type characteristic, and two \(C_3\)-type characteristics (see Fig. A-1). Consistent with the above remarks, we will find that the three compatibility conditions corresponding to the \(C_1\) and \(C_3\) characteristics will provide differential equations for advancing three unknown quantities on \(X = 0\). These quantities together with the boundary condition \(u = b\omega + (b/\phi)v\) determine all flow variables on \(X = 0\).

On the boundary \(X = 1\), there are three \(C_1\)-type characteristics and one \(C_2\)-type characteristic (there are no \(C_3\)-types). We will find that the compatibility condition corresponding to the \(C_2\)-type characteristic will provide a differential equation for determining the shock shape function \(c(\phi, z)\). The shock geometry together with the Rankine-Hugoniot relations \((2.7)\) determine all flow quantities on \(X = 1\).

In order to derive the necessary characteristic conditions, the system \((A-1)\) is rewritten in an equivalent quasi-linear form. This can be done by changing the dependent variables in \((A-1)\) from \(U\) to

\[
Q = \begin{pmatrix} p \\ u \\ v \\ w \end{pmatrix},
\]
Fig. A-1. Types of characteristic curves through boundary points on \( X = 0 \) and \( 1 \)
We assume here that \( p = p(p,h) \) where \( h \) is given by (2.2). Using the chain rule, we have
\[
\frac{\partial U}{\partial z} = \left[ \frac{\partial U}{\partial Q} \right] \frac{\partial Q}{\partial z} \tag{A-2}
\]
and, from (3.3a),
\[
\frac{\partial F}{\partial x} = \left( \frac{\partial}{\partial x} \left[ \frac{\partial U}{\partial Q} \right] + x \left[ \frac{\partial^2 F}{\partial Q \partial x} \right] + x \left[ \frac{\partial^2 F}{\partial Q \partial \phi} \right] \right) \frac{\partial Q}{\partial x}
+ \frac{\partial x}{\partial x} U + \frac{\partial x}{\partial x} \phi + \frac{\partial x}{\partial \phi} \psi - \frac{1}{r} \frac{\partial}{\partial \phi} \frac{\partial x}{\partial \phi} \psi \tag{A-3}
\]

where \( \left[ \frac{\partial U}{\partial Q} \right] \), \( \left[ \frac{\partial^2 F}{\partial Q \partial x} \right] \), and \( \left[ \frac{\partial^2 F}{\partial Q \partial \phi} \right] \) are Jacobian matrices.

These can be obtained by taking partial derivatives with respect to the components of \( Q \) of (2.1a) and (2.1b) using (2.2); for example,
\[
\left[ \frac{\partial U}{\partial Q} \right] = \begin{bmatrix}
\kappa_2 w & -\kappa_1 w u & -\kappa_1 w v & \rho - \kappa_1 w^2 \\
1 + \kappa_2 w^2 & -\kappa_1 w^2 u & -\kappa_1 w^2 v & 2 \rho w - \kappa_1 w^3 \\
\kappa_2 u v & \rho w - \kappa_1 w u^2 & -\kappa_1 u w v & \rho u - \kappa_1 w^2 u \\
\kappa_2 v w & -\kappa_1 w v u & \rho w - \kappa_1 w v^2 & \rho v - \kappa_1 w^2 v
\end{bmatrix}
\]

where \( \kappa_1 = \frac{\partial h}{\partial \rho} \) and \( \kappa_2 = \frac{\partial h}{\partial \rho} \) are thermodynamic quantities which are related to the sound speed by
\[
\frac{\rho}{\alpha^2} = \rho \kappa_2^2 \tag{A-4}
\]
Substitution of (A-2) and (A-3) into (A-1) yields the desired quasi-linear system. This system is simplified by multiplying each term on the left by the non-singular matrix
\[
\mathcal{D} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
-w & 1 & 0 & 0 \\
-u & 0 & 1 & 0 \\
-v & 0 & 0 & 1
\end{bmatrix}
\]
The resulting system of equations is

\[ A \frac{\partial^2 Q}{\partial z^2} + B \frac{\partial^3 Q}{\partial x \partial z^2} = \vec{R} \]  

(A-5)

where

\[
\begin{bmatrix}
\chi_2 \rho -1 & -\chi_1 \omega & -\chi_1 \omega & \rho -\chi_1 \omega^2 \\
1 & 0 & 0 & \rho \omega \\
0 & \rho \omega & 0 & 0 \\
0 & 0 & \rho \omega & 0 \\
\end{bmatrix}
\]

(A-5a)

\[ A = \begin{bmatrix}
\chi_2 \rho \omega - u \chi_1 \omega \\
0 \\
0 \\
0 \\
\end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix}
\chi_2 \rho \omega - u \chi_1 \omega \\
X_z \\
X_r \\
\frac{1}{r} X_\phi \\
\end{bmatrix} \]

(A-5b)

The system (A-5) can be reduced to characteristic form by standard methods (see, e.g., ref. 10, pp. 424-427) although the required computations are lengthy. We summarize the procedure here. Since the matrix is non-singular for \( \omega > a \), we can consider the characteristic matrix \( A^* \) associated with (A-5) in the form \( A^* = \lambda A + B \) where \( \lambda \) is a scalar. For every real value of \( \lambda \) which is a root of
there corresponds a real left null vector, \( \xi_\lambda \), of \( A^\ast \) (i.e., \( \xi_\lambda A^\ast = 0 \)).

Hence,

\[
\xi_\lambda \mathbb{B} = -\lambda \xi_\lambda A .
\]  

(A-7)

Multiplying (A-5) on the left by \( \xi_\lambda \) and using (A-7), we obtain the scalar equation

\[
\xi_\lambda A (\frac{\partial \rho}{\partial \rho} - \frac{\partial \rho}{\partial \lambda}) = \xi_\lambda \tilde{A}.
\]  

(A-8)

This equation is referred to as the characteristic compatibility condition along the characteristic curve defined by \( \frac{dX}{dZ} = -\lambda \) since the quantity

\[
\frac{\partial \rho}{\partial \rho} - \frac{\partial \rho}{\partial \lambda}
\]  

is the directional derivative tangent to this curve. The roots of (A-6) are

\[
\lambda_0 = -\frac{A}{\rho} \text{ (multiplicity 2)}
\]

and

\[
\lambda_{\pm} = \frac{-\rho A + a^2 X_Z \pm a^2 \beta_1}{\rho^2 - a^2}, \quad \beta_1 = \sqrt{\frac{(\rho^2 - a^2) (X_X^2 + \frac{1}{2} X_\phi^2) + (A - X_X \rho)^2}{a^2}} .
\]  

(A-9a)

Two independent null vectors correspond to the multiple root \( \lambda_0 \); viz.,

\[
\xi_{\lambda_0}^{(1)} = (0, 0, \frac{1}{r} X_\phi, -X_r), \quad \xi_{\lambda_0}^{(2)} = (0, \rho, u, v) .
\]  

(A-9b)

A left null vector corresponding to \( \lambda_\pm \) is

\[
\xi_{\lambda_\pm} = (\lambda_\pm \rho + A)(2\rho - \mathcal{K}_1 v^2), -\rho(\lambda_\pm + X_Z) + \mathcal{K}_1 w(\lambda_\pm \rho + A),
\]

\[
-\rho X_r + \mathcal{K}_1 u(\lambda_\pm \rho + A), -\rho X_\phi + \mathcal{K}_1 v(\lambda_\pm \rho + A) .
\]  

(A-9c)

73
The standard form for the compatibility equations (A-8) is to introduce
the associated directional derivatives. Used in this form the boundary
point computations would be essentially as in a reference plane method of
characteristics. Such a procedure is complicated since it requires a
considerable amount of interpolation. The method used herein eliminates
this difficulty by considering the partial differential equations (A-8)
directly. These equations are rewritten on the surfaces X = 0 and 1 using
the boundary conditions.* The resulting equations, valid only on X = 0
and 1, are solved numerically using predictor-corrector methods similar to
those used for interior points.

Wall Boundary Points (X = 0)

On the boundary X = 0, u = bzw + (b/b)v which implies that A = 0 on
X = 0 (c.f., eqs. (3.3d) and (3.3g)). Thus on X = 0, $\delta_1 > |X_z|$ (since w > a)
which implies that $\lambda_+ > 0$ and $\lambda_- < 0$. Also on X = 0, $\lambda_0 = 0$. Note that on
X = 0, $\lambda_0$ corresponds to a streamline since X = 0 is a stream surface.

Referring to Fig. (A-1), $\lambda_+$ corresponds to the characteristic curve $C_1$,
$\lambda_-$ corresponds to $C_2$, and $\lambda_0$ corresponds to $C_3$. According to our earlier
remarks, we exclude from further consideration the root $\lambda_-$. We consider first the compatability condition (A-8) with $\lambda_0 = \lambda_+$. The left-hand side of (A-8) is, using (A-5a) and (A-9),

$$\rho(\psi_1 \left( \frac{3p}{\rho z} - \lambda \right) + \frac{3p}{\rho z}) - \rho w \left( X \frac{3w}{\rho z} + X \frac{3u}{\rho z} \right) + \rho w \lambda \left( X \frac{3w}{\rho z} + X \frac{3u}{\rho z} \right)$$

Note that at the boundary X = 0, we have using the boundary condition and

(3.3g) that

$$X \frac{3w}{\rho z} + X \frac{3u}{\rho z} = X \left( -b \frac{3w}{\rho z} - \frac{3u}{b \rho z} \right) = X \left( w \frac{3b}{\rho z} + v \frac{3b}{\rho z} \right)$$

where $\frac{3b}{\rho z}$ and $\frac{3b}{\rho z}$ depend only on the given body geometry. Using the above,
we can write the compatability condition corresponding to $\lambda_+$ at the

*This approach was originally suggested by Kentzer (ref. 11).
The completely expanded form of (A-10) given by (3.16) is used to advance the pressure on the wall.

The left-hand side of (A-8) corresponding to $\lambda = \lambda_{\phi}^{(1)}$ is given by

$$\rho \omega \left( \frac{1}{r} \frac{\partial u}{\partial Z} - X \frac{\partial v}{\partial Z} \right) - \lambda_{\phi}^{(1)} \left( \frac{1}{r} \frac{\partial u}{\partial Z} \right) - \lambda_{\phi}^{(1)} \left( \frac{1}{r} \frac{\partial v}{\partial Z} \right)$$

On $X=0$, $\lambda_{\phi}^{(1)} = 0$ and $\frac{1}{r} \frac{\partial u}{\partial Z} = - X \left( \frac{b_{\phi}}{b} \right)$. Hence, the compatibility condition corresponding to $\lambda = \lambda_{\phi}^{(1)}$ at the wall can be written as

$$\rho \omega X_{r} \left( \frac{\partial v}{\partial Z} - u \frac{\partial v}{\partial Z} \right) = \lambda_{\phi}^{(1)} \mathbf{\kappa}$$

where

$$V_{2} = v + \left( \frac{b_{\phi}}{b} \right) u$$

The completely expanded form of this equation is given in (3.18). Equation (A-11) is used to advance the quantity $V_{2}$ on the wall. Note that $V_{2}$ is, except for a factor involving only body geometry, a component of velocity tangent to the wall.

The compatibility equation corresponding to $\lambda = \lambda_{\phi}^{(2)}$ can be written, by direct expansion of (A-8), as

$$\nu \left( \frac{\partial p}{\partial Z} + \rho \frac{\partial v^{2}}{\partial Z} \right) + \lambda \left( \frac{\partial p}{\partial Z} + \rho \frac{\partial v^{2}}{\partial Z} \right) = - B \left( \frac{\partial p}{\partial Y} + \rho \frac{\partial v^{2}}{\partial Y} \right)$$

where $v^{2} = w^{2} + u^{2} + v^{2}$. The above equation involves only derivatives of thermodynamic quantities. Indeed, using (2.2) and (A-4), we have (since $\rho = \rho(p,h)$)
Introducing the entropy, $s$, we have

$$\left(\frac{4}{3s}\right)_p ds = \frac{1}{a} dp - \dot{p}_d$$

which with (A-12) implies that the compatibility equation can be written as

$$w \frac{\partial s}{\partial X} + A \frac{\partial s}{\partial X} = -B \frac{\partial s}{\partial Y}.$$ 

This equation expresses, in the computational space, that the rate of change of entropy on a streamline is zero. On the boundary $X=0$, we have (since $A=0$)

$$w \frac{\partial s}{\partial Z} = -B \frac{\partial s}{\partial Y} \tag{A-13}$$

Note that eqs. (A-10), (A-11) and (A-13) can be used to advance $p$, $V_2$, and $s$ at the wall points. From the values of $p$ and $s$, the quantities $\rho$ and $h$ can be determined. The magnitude of the velocity vector can then be determined using (2.2). The quantity $V_2$ and the boundary condition then give the velocity components. This procedure is described in detail in section 3.4.

**Shock Boundary Points ($X = 1$)**

On the boundary $X = 1$,

$$A = (-c_2 w + u - c_\phi v/c)X_r = -V_n v_s X_r$$

(c.f., (3.3d), (3.3g)). Since $V_n > 0$, it follows from (4.9a) that $\lambda_o > 0$. On $X = 1$, the roots $\lambda_\pm$ of (A-6) are (see (A-9a))

$$\lambda_\pm = \frac{(wV_n v_s - a_2 c_\phi)X_r \pm a^2 \beta_1}{w^2 - a^2}$$

76
where
\[ \beta_1^2 = \frac{X^2}{a^4} \left( \left( \omega_n \nu_s - a^2 c_z \right)^2 + \nu_s^2 \left( \omega^2 - a^2 \right) \left( a^2 - \nu_n^2 \right) \right). \]

Since \( w > a \) and shock theory gives \( \nu_n \leq a \), it follows that
\[ a^2 \beta_1 \geq X \left( \omega_n \nu_s - a^2 c_z \right) > 0. \] Hence \( \lambda_+ > 0 \) and \( \lambda_- \leq 0 \) (equality when \( \nu_n = a \) which corresponds to zero shock strength). Referring to Fig. (A-1), \( \lambda_0 \) and \( \lambda_+ \) correspond to characteristic curves of the type \( C_1 \), and thus, by our earlier remarks, we exclude these roots from further consideration.

The compatibility condition (A-8) with \( \kappa_\lambda = \kappa_\lambda^- \) can be written using (A-5a), (A-7) and (A-9b) as
\[
\beta_1 \frac{\partial p}{\partial Z} + \rho \left[ X \left( \omega \frac{\partial u}{\partial Z} - u \frac{\partial \omega}{\partial Z} \right) + \frac{1}{r} X \left( \omega \frac{\partial v}{\partial Z} - v \frac{\partial \omega}{\partial Z} \right) \right] = -\frac{1}{\rho} \kappa_\lambda^- \left( \tilde{R} - B \right) \tag{A-14}
\]

Note that (A-3), (A-5a), and (3.3c) imply that
\[ \tilde{R} - B = -\left( \frac{\partial F}{\partial X} + \frac{\partial G}{\partial Y} + E \right). \]

Consider now the left hand side of (A-14) evaluated on the surface \( X = 1 \). Since the Rankine-Hugoniot conditions are satisfied identically on \( X = 1 \), these relations can be differentiated with respect to \( Z \) in order to obtain expressions for \( \frac{\partial p}{\partial Z}, \frac{\partial u}{\partial Z}, \frac{\partial v}{\partial Z}, \) and \( \frac{\partial \omega}{\partial Z} \) on \( X = 1 \). Substitution of these expressions into (A-14) yields a partial differential equation for the shock slopes \( c_z \) and \( c_\phi \). We summarize the development of this equation here. From the shock relations (3.9), we have on \( X = 1 \) that
\[ w = w_\infty - (V_n - V_n) e_1, \quad u = u_\infty + (V_n - V_n) e_2, \quad \text{and} \quad v = v_\infty - (V_n - V_n) e_3 \]
where \( e_1 = c_z / \nu_s, \quad e_2 = 1 / \nu_s, \) and \( e_3 = c_\phi / (c v_s) \). Differentiating these with respect to \( Z \), we obtain that on \( X = 1 \)

77
\[
\begin{align*}
\frac{\partial \mathbf{u}}{\partial z} &= \frac{3\mathbf{V}_\infty}{\partial z} - \frac{3\mathbf{V}}{\partial z} e_1 - (\mathbf{V}_\infty - \mathbf{V}) \frac{\partial \mathbf{e}_1}{\partial z} \\
\frac{3\mathbf{V}}{\partial z} &= \frac{3\mathbf{u}_\infty}{\partial z} + (\frac{\partial \mathbf{v}_\infty}{\partial z} - \frac{3\mathbf{V}}{\partial z}) e_2 + (\mathbf{V}_\infty - \mathbf{V}) \frac{\partial \mathbf{e}_2}{\partial z} \\
\frac{3\mathbf{V}}{\partial z} &= \frac{3\mathbf{u}_\infty}{\partial z} - (\frac{\partial \mathbf{v}_\infty}{\partial z} - \frac{3\mathbf{V}}{\partial z}) e_3 - (\mathbf{V}_\infty - \mathbf{V}) \frac{\partial \mathbf{e}_3}{\partial z}
\end{align*}
\] (A-15a)

Note that in the above
\[
\frac{3\mathbf{u}_\infty}{\partial z} = \mathbf{v}_\infty \frac{\partial \mathbf{v}}{\partial z} \quad \text{and} \quad \frac{3\mathbf{V}_\infty}{\partial z} = - \mathbf{u}_\infty \frac{\partial \mathbf{v}}{\partial z}.
\] (A-15b)

Since \( \mathbf{V}_\infty = \mathbf{w} - \mathbf{u}_\infty \mathbf{e}_1 + \mathbf{v}_\infty \mathbf{e}_2 + \mathbf{v}_\infty \mathbf{e}_3 \), we have that
\[
\frac{3\mathbf{V}_\infty}{\partial z} = \mathbf{w}_\infty \frac{\partial \mathbf{e}_1}{\partial z} - \mathbf{u}_\infty \frac{\partial \mathbf{e}_2}{\partial z} + \mathbf{v}_\infty \frac{\partial \mathbf{e}_3}{\partial z} - (\mathbf{v}_\infty \mathbf{e}_2 + \mathbf{u}_\infty \mathbf{e}_3) \frac{3\mathbf{V}}{\partial z}.
\] (A-15c)

From the shock relations (2.7), we have that \( p = \rho \rho_\infty (\mathbf{V}_\infty - \mathbf{V}) \)
which implies that on \( X = 1 \)
\[
\frac{3p}{\partial z} = \rho_\infty \left[ \frac{3\mathbf{V}_\infty}{\partial z} + \mathbf{V}_\infty \left( \frac{\partial \rho_\infty}{\partial z} - \frac{3\mathbf{V}}{\partial z} \right) \right]
\] (A-16)

In order to obtain an expression for \( \mathbf{n} \), we differentiate the equation
of state \( \rho = \rho(h,p) \) to obtain
\[
\frac{3\rho}{\partial z} = \gamma_1 \frac{3h}{\partial z} + \gamma_2 \frac{3p}{\partial z}
\] (A-17)

Differentiating the first and last equations in (2.7), we obtain for
points on \( X = 1 \) that
\[
\frac{3\rho}{\partial z} = \rho_\infty \left[ \frac{3\mathbf{V}_\infty}{\partial z} - (\mathbf{V}_\infty / \mathbf{V}) \frac{3\mathbf{V}}{\partial z} \right] \mathbf{V}_n
\]
\[
\frac{3h}{\partial z} = \mathbf{V}_\infty \frac{3\mathbf{V}}{\partial z} - \mathbf{V}_n \frac{3\mathbf{V}}{\partial z}
\]
Substituting these and (A-16) into (A-17) and using (A-4), we obtain

\[ \frac{3V}{\partial z} \left( \frac{\partial n}{\partial z} \right) - \frac{3V}{\partial z} \frac{n}{\rho} = \left( \frac{1}{\rho_{\infty}} \left( \frac{V_{n}}{V_{n_{\infty}}} - \frac{V_{n}}{V_{n_{\infty}}} \right) \left[ \frac{1}{V_{n}^{2}} + \frac{1}{a_{\infty}^{2}} + \frac{1}{\rho_{\infty}} \left( \frac{1}{\rho_{\infty}} - \frac{1}{\rho} \right) \right] \right) \frac{3V}{\partial z} \right) \frac{n}{\rho_{\infty}} \]  

(A-18)

Finally, we have that

\[ \frac{\partial e_{1}}{\partial z} = \left( 1 - \frac{e_{2}}{e_{2}} \right) \frac{3c}{\partial z} - e_{1} \frac{3(c_{\phi}/c)}{\partial z} \]  

(A-19a)

\[ \frac{\partial e_{2}}{\partial z} = - \left[ e_{1} \frac{3c}{\partial z} + e_{3} \frac{3(c_{\phi}/c)}{\partial z} \right] e_{2} \]  

(A-19b)

\[ \frac{\partial e_{3}}{\partial z} = - \left[ e_{1} e_{3} \frac{3c}{\partial z} + (1 - e_{2}) \frac{3(c_{\phi}/c)}{\partial z} \right] e_{2} \]  

(A-19c)

Substituting (A-15a) and (A-16) into (A-14) and using (A-15c), (A-18) and (A-19) to eliminate the terms \( \frac{\partial n}{\partial z}, \frac{3V}{\partial z}, \frac{\partial n_{\infty}}{\partial z}, \frac{\partial e_{1}}{\partial z} (i=1,2,3) \), we obtain from the compatability condition (A-14) a partial differential equation of the form

\[ C_{1} \frac{3c}{\partial z} + C_{2} \frac{3(c_{\phi}/c)}{\partial z} = \phi_{s} \]  

(A-20)

The quantities \( C_{1}, C_{2}, \) and \( \phi_{s} \) appearing in (A-20) are given in section 3.3.

In addition to (A-20), we also have

\[ \frac{3c}{\partial z} = c_{z} + c_{\phi} \frac{3\phi}{\partial z} \]  

(A-21)

and

\[ \frac{3c_{\phi}}{\partial z} - \frac{3c_{\phi}}{\partial z} + \frac{3c_{\phi}}{\partial z} = 0 \]  

(A-22)
The first equation (A-21) is the chain rule for the $Z$ derivative of the shock shape function $c(t,z)$. The second equation (A-22) expresses through the chain rule that $\frac{\partial^2 c}{\partial x \partial z} = \frac{\partial^2 c}{\partial \phi \partial Z}$. When $C_1$ is nonzero the equations (A-20) - (A-22) are formally a hyperbolic system with $Z$ the time-like direction and, therefore, can be numerically solved using a predictor-corrector method to determine $c$, $c_z$, and $c_{\phi}$. With the shock geometry determined, the Rankine-Hugoniot conditions determine all the flow variables on $X = 1$. The procedure is described in detail in section 3.3.

The above procedure requires that the coefficient $C_1$ does not change sign in the calculation. We now show that for a perfect gas $C_1 > 0$ when $\mathbf{w} \geq \mathbf{a}$. The proof uses the definitions following (3.9c) (see sec. 3.3) and the shock relations (2.7). Consider first the case of a finite strength shock; i.e., $V_{n_s} > V_n, a > V_n, p > p_\infty, \rho > \rho_\infty$. Substituting (2.7) and $P_1 = -\rho/h$ into the definition of $A_0$, we obtain

$$A_0 = \frac{(p - p_\infty)(V_n V_{n_s} + a^2 p_\infty/p)}{\rho_\infty V_{n_s}^2 (a^2 - V_n^2/n^2)}.$$  \hspace{1cm} \text{(A-23)}

This implies that $A_0 > 0$ and it follows from the definition of $A_1$ that

$$A_1 > \rho(\nu_s \mathbf{w} - c_z V_{n_{s,\infty}})A_0.$$  \hspace{1cm} \text{(A-24)}

From the shock relations and $\mathbf{w} \geq \mathbf{a}$, we have that $\nu_s \mathbf{w} - c_z V_{n_{s,\infty}} = \nu_s \mathbf{w} - c_z V_n \geq \nu_s a - c_z V_n > 0$.  \hspace{1cm} \text{(A-25)}

Using (A-24) and (A-25) in the definition of $C_1$ we obtain

$$C_1 > \frac{(\nu_s \mathbf{w} - c_z V_{n_{s,\infty}})^2 A_0 - (p - p_\infty)(\nu_s^2 - c_z^2)}{\nu_s^2}.$$  \hspace{1cm} \text{(A-26)}

Note that the significance of $C_1 > 0$ for the real gas case is the same as for a perfect gas but this fact has not been established at this time. In practical calculations, the value of $C_1$ is monitored at all points and, if a change of sign is detected the calculation is stopped.
Note that since \( u - c_v/c = c_w - V_n v_s \),

\[
\beta_0^2 a^2 = (v_s w - c_z v_n)^2 - (v_s^2 - c_z^2)(a^2 - V_n^2)
\]

which implies that

\[
(v_s w - c_z v_n)^2 = (v_s w - c_z v_n)^2 \geq (v_s^2 - c_z^2)(a^2 - V_n^2).
\]

Using (A-27), (A-23), (A-26), and \( \rho = \frac{V_n}{V_n} \), we obtain

\[
C_1 > \frac{(v_s^2 - c_z^2)(p - p_\infty)}{v_s^2 \rho_\infty V_n V_n} \left\{ V_n V_n + \frac{2 p_\infty}{p - V_n V_n} \right\} > 0
\]

which is the desired result. Consider now the case of a weak shock; i.e.,

\[
V_n = V_{n_\infty} = a = a_\infty, \quad p = p_\infty, \quad \rho = \rho_\infty.
\]

The quantity \( A_0 \) can be rewritten using (A-23) and the first two equations in (3.12) (with \( \gamma = \Gamma = \Gamma_\infty \)) as

\[
A_0 = \frac{2(v_{n_\infty}^2 + a_\infty)}{(\gamma+1) V_{n_\infty}^2} \quad [\text{perf}]
\]

which implies that \( A_0 = 4/\gamma+1 \) for \( V_{n_\infty} = a_\infty \). Since \( A_0 > 0 \) and

\[
(v_s w - c_z V_{n_\infty}) > 0,
\]

it follows directly from the definitions that \( A_1 > 0 \) and hence \( C_1 > 0 \).
APPENDIX B
SYMMETRY CONDITIONS

In this appendix, the symmetry conditions (3.21) and (3.22) used in the symmetric problem are derived. Recall that in this problem all flow variables are even functions of $\phi$ about $\phi = 0$ and $\pi$ in the physical space except $v$ which is an odd function of $\phi$ about $\phi = 0$ and $\pi$. Also, the body geometry and shock geometry functions are even functions of $\phi$ about 0 and $\pi$. In this appendix, $\phi^*$ will denote either 0 or $\pi$. To avoid any confusion we specify that: a function $u(r,\phi,z)$ is even about $\phi^*$ if and only if $u(r,\phi^*+\tau,z) = u(r,\phi^*+\tau,z)$ for all $\tau$ such that $\phi^* + \tau$ is in the domain of definition of $u$; a function $u(r,\phi,z)$ is odd about $\phi^*$ if and only if $u(r,\phi^*+\tau,z) = -u(r,\phi^*+\tau,z)$.

Consider first the effect of the mapping (3.1) on a quantity $u$ which in the physical plane is given by an even (or odd) function of $\phi$ about $\phi = \phi^*$. Let $u$ be given by $u(r,\phi,z)$ in the physical space and $\bar{u}(\bar{x},\bar{y},\bar{z})$ in the $(\bar{x},\bar{y},\bar{z})$ space; further, let $\bar{y}^* = 4^*/\phi_0$ (i.e., $\bar{y}^* = 0$ or 1). Note that from (3.1), $\bar{x}$ is an even function of $\phi$ about $\phi = \phi^*$ for fixed $r$ and $z$. Also, inverting (3.1), we have

$$r(\bar{x},\bar{y},\bar{z}) = \bar{x}[c(\bar{z},\phi_0\bar{y}) - b(\bar{z},\phi_0\bar{y})] + b(\bar{z},\phi_0\bar{y})$$

which is an even function of $\bar{y}$ about $\bar{y}^*$ for fixed $\bar{x},\bar{y}$. It therefore follows from

$$u(r,\phi,z) = \bar{u}(\bar{x}(r,\phi,z),\phi/\phi_0,z)$$

and

$$\bar{u}(\bar{x},\bar{y},\bar{z}) = u(r(\bar{x},\bar{y},\bar{z}),\bar{y}\phi_0,z)$$

that $u(r,\phi,z)$ is an even (odd) function of $\phi$ about $\phi = \phi^*$ for fixed $r,z$ if and only if $\bar{u}(\bar{x},\bar{y},\bar{z})$ is an even (odd) function of $\bar{y}$ about $\bar{y}^*$ for fixed $\bar{x},\bar{z}$. 

82
Consider now the quantity $u$ given in the computational space $(X,Y,Z)$ by the function $U(X,Y,Z)$. Note that $U(X,Y,Z) = \bar{u}(f(X,Y,Z), g(Y,Z), Z)$. For any fixed $Z$, let $Y^*$ denote the value of $Y$ such that $\bar{y}^* = g(Y^*, Z)$ (i.e., $Y^*$ is 0 or 1). The Taylor development of $U(X,Y,Z)$ about $(X,Y^*,Z)$ where $X$ and $Z$ are fixed gives

$$U(X,Y^* \pm \Delta Y, Z) = U(X,Y^*, Z) \pm \Delta Y U_y(X,Y^*, Z)$$

$$+ \frac{(\Delta Y)^2}{2} U_{yy}(X,Y^*, Z) + \frac{(\Delta Y)^3}{3!} U_{yyy}(X,Y^*, Z) + O(\Delta Y)^4.$$  

Adding and subtracting the above expressions, we obtain

$$U(X,Y^* \pm \Delta Y, Z) - U(X,Y^* + \Delta Y, Z) = \pm 2 \Delta Y U_y(X,Y^*, Z) + O(\Delta Y)^3 \quad \text{(B.1)}$$

and

$$U(X,Y^* \pm \Delta Y, Z) + U(X,Y^* + \Delta Y, Z) = 2 U(X,Y^*, Z) + \Delta Y^2 U_{yy}(X,Y^*, Z)$$

$$+ O(\Delta Y)^4. \quad \text{(B.2)}$$

For use in the above we have, by the chain rule, that

$$U_y = \bar{u}_f f_Y + \bar{u}_g g_Y \quad \text{(B.3)}$$

and

$$U_{yy} = \bar{u}_f f_Y f_Y + 2 \bar{u}_f f_Y g_Y + \bar{u}_g g_Y g_Y$$

$$+ \bar{u}_f g_Y f_Y + \bar{u}_g g_Y g_Y \quad \text{(B.4)}$$

Suppose that $u(r,\psi, z)$ is an even function of $\psi$ about $\psi^*$. By our previous remarks, it follows that $\bar{u}(x, y, z)$ is an even function of $y$ about $\bar{y}^*$. Since $\bar{u}$ is even about $\bar{y}^*$, we have that $\bar{u}_y(x, \bar{y}^*, z) = 0$. Also, if we assume that $f_Y(X,Y^*,Z) = 0$, it follows from (B.3) that $U_y(X,Y^*,Z) = 0$. This implies, using (B.1), that

$$U(X,Y^* \pm \Delta Y, Z) = U(X,Y^* + \Delta Y, Z) + O(\Delta Y)^3$$

which is the same as (3.21).
Suppose now that \( u(r, \phi, z) \) is an odd function of \( \phi \) about \( \phi^* \). By our previous remarks, \( u(x, y, z) \) is an odd function of \( y \) about \( y^* \). Hence, it follows that

\[
\overline{u(x, y^*, z)} = \overline{y} \overline{u(x, y^*, z)} = 0.
\]

Since the above holds for all \( x \), we also have that \( \overline{u(x, y^*, z)} = \overline{y} \overline{u(x, y^*, z)} = 0 \).

If \( f_y(X, Y^*, Z) = 0 \), it follows from (B.4) that

\[
U_{yy}(X, Y^*, Z) = \overline{u_y(x, y^*, z)} \overline{y_y(Y^*, Z)}
\]

and from (B.3) that

\[
\overline{y_y(x, y^*, z)} = U_y(x, Y^*, Z) \overline{y_y(Y^*, Z)}
\]

since \( U(X, Y^*, Z) = 0 \), it follows using (B.2) that

\[
U(X, Y^* \pm \Delta Y, Z) + U(X, Y^* + \Delta Y, Z) =
\]

\[
\Delta Y^2 U_y(x, Y^*, Z) \overline{y_y(Y^*, Z)} / \overline{y_y(Y^*, Z)} + 0(\Delta Y)^4
\]

Using (B.1) to evaluate \( U_y \) in the above, we obtain

\[
U(X, Y^* \pm \Delta Y, Z)[2 \Delta Y \overline{y_y(Y^*, Z)} / \overline{y_y(Y^*, Z)}] =
\]

\[
- U(X, Y^* \pm \Delta Y, Z)[2 \Delta Y \overline{y_y(Y^*, Z)} / \overline{y_y(Y^*, Z)}] + 0(\Delta Y)^4
\]

which is the same as (3.22).
In this appendix, we derive a necessary stability condition used for determining the step size $\Delta Z$ associated with a given computational mesh $\Delta X, \Delta Y$. The derivation is in the geometric spirit of CFL and consists of comparing the domain of dependence* of the difference equations, $\mathcal{D}_{\text{d.e.}}$, to the domain of dependence of the partial differential equations, $\mathcal{D}_{\text{p.d.e.}}$. Following CFL, it is necessary to restrict the largest value of $\Delta Z$ so that

$$\mathcal{D}_{\text{p.d.e.}} \subseteq \mathcal{D}_{\text{d.e.}}$$

for all computational points. For the quasi-linear system of hyperbolic equations considered here this condition is generally regarded as a necessary (but not always sufficient) condition for numerical stability. Indeed, it has been observed that the MacCormack scheme exhibits numerical instability for particular first order systems even when a geometric CFL stability condition is obeyed (ref. 12). For our system no such instabilities have ever been demonstrated nor observed in calculations.

*In this appendix, the domain of dependence of a point, $0$, with $Z$ coordinate $Z + \Delta Z$, is understood to be the smallest closed region of the plane $Z \leq Z_0, \mathcal{F}$, such that the initial data outside of $\mathcal{F}$ do not influence the solution at $0$. 

85
Consider an arbitrary interior computational point, 0, with coordinates \((Z_0 + \Delta Z, X_0, Y_0)\). The domain of dependence of 0 for the finite difference schemes given by (3.6a) and (3.6b) depends on the parameter \(j\) (= 0 or 1). These are depicted in Fig. C-1. The dots in the figures represent the only computational points which can influence the numerical solution at 0 for one complete (predictor and corrector) step. In stability considerations, it is necessary to consider the limit for successive mesh requirements with \(\Delta X/\Delta Z\) and \(\Delta Y/\Delta Z\) held fixed. In this limit, the points in and on the boundary of the shaded regions are the only ones that can influence the numerical solution at 0.

Consider now the domain of dependence of 0 for the system of partial differential equations in the computational space, see (3.3). For the purposes of stability analysis it is sufficient to consider a quasi-linear system equivalent to (3.3) given by

\[
\mathcal{A} \frac{\partial Q}{\partial Z} + \mathcal{B} \frac{\partial Q}{\partial X} + \mathcal{C} \frac{\partial Q}{\partial Y} = \mathcal{R}
\]  

(C.2)

where

\[
Q = \begin{pmatrix} p \\ u \\ v \\ w \end{pmatrix}
\]

In the above,

\[
\mathcal{A} = \mathcal{O} \left\{ \frac{\partial U}{\partial Q} \right\}, \quad \mathcal{B} = \mathcal{O} \left\{ X_x \frac{\partial U}{\partial Q} + X_r \frac{\partial V}{\partial Q} + X_\phi \frac{\partial W}{\partial Q} \right\},
\]

\[
\mathcal{C} = \mathcal{O} \left\{ Y_z \frac{\partial U}{\partial Q} + Y_\phi \frac{\partial W}{\partial Q} \right\}
\]

86
Fig. C-1. Domain of dependence for the interior finite difference equations
where $[\frac{\partial u}{\partial \psi}]$, $[\frac{\partial f}{\partial \psi}]$, and $[\frac{\partial y}{\partial \psi}]$ are Jacobian matrices and $O$ is the non-singular matrix given in Appendix A. The matrices $\mathcal{A}$ and $\mathcal{B}$ are given by (A-5a) and (A-5b) respectively; the matrix $\mathcal{C}$ is given by

$$
\mathcal{C} = \begin{bmatrix}
\mathcal{K}^2 \mathcal{B} & -\mathcal{K}^1 \mathcal{u} \mathcal{B} & \frac{\partial}{\partial \psi} \mathcal{Y} - v \mathcal{B} \mathcal{K}^1 & \mathcal{Y}_\mathcal{Z} - w \mathcal{K}^1 \mathcal{B} \\
\mathcal{Y}_Z & 0 & 0 & \rho \mathcal{B} \\
0 & \rho \mathcal{B} & 0 & 0 \\
\frac{1}{\rho} \mathcal{Y}_\phi & 0 & \rho \mathcal{B} & 0
\end{bmatrix}
$$

The inhomogeneous term $\mathcal{R}$ is not material to the present analysis and will not be given. The domain of dependence of the system (3.3) is the same as that of (C.2). The domain of dependence of $O$ for the system (C.2) is contained in the closed region formed by the intersection of the characteristic conoid with vertex at $O$ and the plane $Z = Z_0$ (c.f.; ref. 10, pp. 649-652).

The pertinent facts concerning the geometry of characteristic conoids associated with systems of the type (C.2) will be briefly reviewed here; for a more detailed explanation see ref. 10 pp. 577-599. The characteristic conoid with vertex $O$ is the envelope of all characteristic surfaces through $O$. A surface $\psi(Z,X,Y) = 0$ is characteristic at a point if its normal at the point satisfies the characteristic condition

$$
\mathcal{H}(\lambda_1, \lambda_2, \lambda_3; O; X,Y,Z) = \det(\lambda_1 \mathcal{A} + \lambda_2 \mathcal{B} + \lambda_3 \mathcal{C}) = 0 \quad \text{(C.3)}
$$
where \( \lambda_1 = \frac{\partial \phi}{\partial z}, \lambda_2 = \frac{\partial \phi}{\partial x}, \) and \( \lambda_3 = \frac{\partial \phi}{\partial y}. \) The surface of the characteristic conoid is generated by curves, called rays or bicharacteristics which are the lines of contact between the characteristic surfaces and the conoid they envelope. These curves, or rays, are given by the ordinary differential equations

\[
\frac{dz}{ds} = \frac{\partial \phi}{\partial \lambda_1}, \quad \frac{dx}{ds} = \frac{\partial \phi}{\partial \lambda_2}, \quad \frac{dy}{ds} = \frac{\partial \phi}{\partial \lambda_3} \quad (C.4)
\]

where \( s \) is a parameter. Each ray through 0 is determined by selecting real values for \( \lambda_2 \) and \( \lambda_3 \) and determining \( \lambda_1 \) by satisfying the characteristic condition (C.3) at 0. In general, when \( \mathbf{A}, \mathbf{B}, \) and \( \mathbf{C} \) are not constant, \( \lambda_2, \lambda_3, \) and \( \lambda_1 \) vary along the ray. When (C.2) is a quasi-linear system like (C.2) the rays, conoids, and domains of dependence will depend on the solution vector \( \mathbf{Q}. \) In the case when the coefficient matrices \( \mathbf{A}, \mathbf{B}, \) and \( \mathbf{C} \) are constant \( \lambda_1, \lambda_2, \) and \( \lambda_3 \) satisfying (C.3) are constant along each ray. The rays generating the conoid are straight lines since the right hand sides of (C.4) are constant. The characteristic surfaces in this case are hyperplanes and cones formed by their envelopes.

In the present analysis \( \Delta Z, \Delta X, \Delta Y \) are assumed to be small, thus the domain of dependence of \( 0 = (Z_0 + \Delta Z, X_0, Y_0) \) can be approximated by considering the matrices \( \mathbf{A}, \mathbf{B}, \) and \( \mathbf{C} \) as constants with elements evaluated at the point \( (Z_0, X_0, Y_0). \) For the remainder of this appendix, it will be understood without changing notation that all quantities appearing in these matrices are held fixed at their values at \( (Z_0, X_0, Y_0). \)
Making the substitutions indicated above, the characteristic condition

\((C.3)\) for the system of equations \((C.2)\) is

\[
\mathcal{H} = \mathcal{H}_1(\lambda_1,\lambda_2,\lambda_3) \mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = 0
\]

where

\[
\mathcal{H}_1(\lambda_1,\lambda_2,\lambda_3) = \lambda_1 w + \lambda_2 A + \lambda_3 B
\]

\[
\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = \mathcal{H}_1^2 - \left[ (\lambda_1 + \lambda_2 x_z + \lambda_3 y_z)^2 + \lambda_2^2 x_r^2 + (\lambda_2 x_\phi + \lambda_3 y_\phi)^2 \right] \frac{1}{r^2} a^2
\]

The ray cone therefore has two sheets, one corresponding to \(\mathcal{H}_1 = 0\) and one corresponding to \(\mathcal{H}_2 = 0\). The rays generating the sheet corresponding to \(\mathcal{H}_1 = 0\) are given, using \((C.4)\), by

\[
\frac{dZ}{ds} = \omega, \quad \frac{dX}{ds} = A, \quad \frac{dY}{ds} = B.
\]

Hence the sheet corresponding to \(\mathcal{H}_1 = 0\) is a degenerate cone consisting of a single ray through 0. The domain of dependence for this sheet is a point in the plane \(Z = Z_0\) with coordinates \(X = X_0\) \((A/W)\Delta Z\) and \(Y = Y_0\) \((B/W)\Delta Z\). This ray corresponds in the physical space \((z,\phi,r)\) to a streamline. The sheet corresponding to \(\mathcal{H}_2 = 0\) is a true cone which corresponds in physical space to the Mach cone.**

**A characteristic surface \(\psi(Z,X,Y) = 0\) can be represented by \(\tilde{\psi}(z,r,\phi) = 0\) in physical space. By the chain rule

\[
\tilde{\psi}_z = \lambda_1 \psi_z + \lambda_2 x_z + \lambda_3 y_z, \quad \tilde{\psi}_r = \lambda_2 x_r + \lambda_3 y_r, \quad \text{and} \quad \tilde{\psi}_\phi = \lambda_2 x_\phi + \lambda_3 y_\phi,
\]

since \(\lambda_1 = \psi_z, \lambda_2 = \psi_r, \lambda_3 = \psi_\phi\). It follows that

\[
\mathcal{H}_2(\lambda_1,\lambda_2,\lambda_3) = (\tilde{\psi} \cdot \tilde{q})^2 - |\tilde{\psi}|^2 a^2 = 0
\]

where \(\tilde{q}\) is the velocity vector. This equation indicates that in \((z,r,\phi)\) space the cosine of the angle between the streamline and the normals to the characteristic surfaces enveloping the cone associated with \(\mathcal{H}_2 = 0\) is \(\pm a/|\tilde{q}| = \pm M^{-1}\).
space the streamline through the Mach cone vertex is interior to the cone. Since a continuous transformation cannot change this situation, it follows that in the computational space the cone corresponding to \( H_2 = 0 \) contains the ray corresponding to \( H_1 = 0 \). Therefore the domain of dependence of (C.2) is determined entirely by the sheet \( H_2 = 0 \).

The CFL stability condition for each of the MacCormack schemes is illustrated in Fig. C-2. The shaded area is the region formed by the intersection on the plane \( Z = Z_0 \) of the cone with vertex at 0 corresponding to \( H_2 = 0 \). Condition (C.1) for \( j = 0 \) or 1 is satisfied if and only if

\[
\max_{i=1,2} (\ell_i) \leq AX, \max_{i=3,4} (\ell_i) \leq AY, \quad \text{and} \quad \max_{i=5,6} (\ell_i) \leq \Delta X \Delta Y / \sqrt{\Delta X^2 + \Delta Y^2} \tag{C.6}
\]

In the above, \( \ell_i \) (i=1,...,6) are projections of the base of the cone in various directions as indicated in Fig. C-2. These distances depend on the value of \( \Delta Z \) (recall that \( \Delta X \) and \( \Delta Y \) are assumed fixed in the present analysis).

Consider now the determination of the projection of the cone's base on any directed line in the \( Z = Z_0 \) plane with, say, \( X \) and \( Y \) direction numbers \( \sigma_X \) and \( \sigma_Y \), respectively. Since any characteristic plane \( \psi(Z,X,Y) = 0 \) through 0 is tangent to the cone, the particular ones which have normals with \( \psi_X = \lambda_2 = \sigma_X \) and \( \psi_Y = \lambda_3 = \sigma_Y \) intersect the plane \( Z = Z_0 \) on lines which are both tangent to the cone's base and normal to the direction \( (\sigma_X, \sigma_Y) \). The situation is shown in Fig. C-3. The point of tangency, \( Q \), is the intersection on the \( Z = Z_0 \) plane of the bicharacteristic ray with \( \lambda_2 = \sigma_X, \lambda_3 = \sigma_Y \), and \( \lambda_1 = \sigma_Z \) where \( \sigma_Z \) is the solution of
Fig. C-2. CFL conditions for MacCormack schemes
Fig. C-3.
\( \mathcal{H}_2(\sigma_z, \sigma_x, \sigma_y) = 0 \). For \( w > a \), there are precisely two distinct solutions of this equation. The two values of \( \sigma_z \) determine two rays and, hence, two points of tangency for each direction \((\sigma_x, \sigma_y)\). The \( X, Y \) coordinates of \( Q \) for each value of \( \sigma_z \) are determined by integrating (C-4) from \( Z = Z_0 + \Delta Z \) to \( Z = Z_0 \); i.e.,

\[
X - X_o = -\left( \frac{\partial \mathcal{H}_2}{\partial \lambda_2} \right) \Delta Z, \quad Y - Y_o = -\left( \frac{\partial \mathcal{H}_2}{\partial \lambda_1} \right) \Delta Z
\]

where \( \frac{\partial \mathcal{H}_2}{\partial \lambda_i} \) (i=1,2,3) are evaluated at \( \lambda_1 = \sigma_z, \lambda_2 = \sigma_x, \lambda_3 = \sigma_y \). The projected distance \( \ell \) in Fig. (C.3), is given by

\[
\ell = \frac{|\sigma_x (X - X_o) + \sigma_y (Y - Y_o)|}{\sqrt{\sigma_x^2 + \sigma_y^2}}.
\]

Since \( \mathcal{H}_2(\lambda_1, \lambda_2, \lambda_3) \) is homogeneous in \( \lambda_1, \lambda_2, \lambda_3 \) it follows that

\[
\lambda_1 \frac{\partial \mathcal{H}_2}{\partial \lambda_1} + \lambda_2 \frac{\partial \mathcal{H}_2}{\partial \lambda_2} + \lambda_3 \frac{\partial \mathcal{H}_2}{\partial \lambda_3} = 0 \quad \text{when} \quad \mathcal{H}_2(\lambda_1, \lambda_2, \lambda_3) = 0.
\]

It therefore follows that

\[
\ell = |\sigma_z| \Delta Z / \sqrt{\sigma_x^2 + \sigma_y^2}.
\]

The distances \( \ell_1 \) and \( \ell_2 \) are projections on the direction \( \sigma_x = 1, \sigma_y = 0 \) (c.f., Fig. C-2). Solving \( \mathcal{H}_2(\sigma_z, 1, 0) = 0 \), we obtain using (C-7) that

\[
\ell_{1,2} = \left| X_z a^2 - AW \pm a \sqrt{(A - W X_z)^2 + (W^2 - a^2)(X_z^2 + \frac{1}{r^2} X_r^2)} \right| \frac{\Delta Z}{(w^2 - a^2)}.
\]

Substituting the above into the first inequality in (C.6), we obtain

\[
\Delta Z \leq \frac{\Delta X}{\mu_1} (w^2 - a^2)
\]
where \( u_1 \) is given in section 3.6. The distances \( l_3 \) and \( l_4 \) are projections on the direction \( \sigma_x = 0, \sigma_y = 1 \). Hence \( l_3 \) and \( l_4 \) are given by (C-7) with \( \sigma_z \) a solution of \( \mathcal{H}_2(\sigma_z, 0, 1) = 0 \); i.e.,

\[
l_{3,4} = \left| Y_z a^2 - wB \pm a \sqrt{(Y_z^2/r^2)(w^2 + v^2 - a^2)} \right| \frac{\Delta Z}{(w^2 - a^2)}
\]

Substituting this into the second inequality in (C.6), we obtain

\[
z \leq \frac{(w^2 - a^2)}{\nu_2} \Delta X \tag{C.9}
\]

where \( \nu_2 \) is given in section 3.6. The distances \( l_{5,6}^1 \) and \( l_{6}^1 \) are projections on the direction \( \sigma_x = \Delta Y, \sigma_y = -(1)^j \Delta X \). Hence \( l_{5,6}^1 \) and \( l_{6}^1 \) are given by (C-7) with \( \sigma_z \) a solution of \( \mathcal{H}_2(\sigma_z, \Delta Y, -(1)^j \Delta X) = 0 \); i.e.,

\[
l_{5,6}^1 = \left( \frac{\Delta Y \Delta X}{(\Delta X^2 + \Delta Y^2)(w^2 - a^2)} \right) \left\{ \delta(wB - Y_z a^2) - (wA - X_z a^2) \right. \\
\left. \quad \pm a \sqrt{(w^2 - a^2)[X_z^2 + \frac{1}{r^2} (X_x - \delta Y_x)^2] + (wX_z - A + \delta v Y_z/r)^2} \right\}
\]

where

\[
\delta = -(1)^j (\Delta X/\Delta Y)
\]

Substituting this into the last inequality in (C.6), we obtain

\[
\Delta Z \leq \frac{(w^2 - a^2)}{\nu_3} \Delta X \tag{C.10}
\]

where \( \nu_3 \) is given in section 3.6.
For any interior computational point \((Z_o + \Delta Z, X_o, Y_o)\), the largest value of \(\Delta Z\) for which the CFL condition (C.1) is satisfied is

\[
\Delta Z = \left\{ \frac{(w^2 - a^2)}{\max(u_1, u_2, u_3)} \right\} \Delta X \quad (C.11)
\]

where the terms in the bracket are evaluated at \((Z_o, X_o, Y_o)\). It remains now to consider the points on the boundary of the computational domain.

The computational scheme for points on the boundaries \(Y = 0\) and \(Y = 1\) which are not on \(X = 0\) or \(X = 1\) is essentially the same as the interior point scheme (c.f., section 3.5); hence, the CFL condition for such points is the same as for interior points. The domain of dependence of the partial differential equations for points on the boundaries \(X = 0\) and \(X = 1\) is essentially as described above for interior points except that only the portion of the characteristic cone's base lying in \(0 \leq X \leq 1\) is considered. The domain of dependence of the difference schemes used on the boundaries \(X = 0\) and \(X = 1\) are illustrated in Fig. C-4 and C-5. Note that at the boundary \(X = 0\) the domain of dependence of the scheme when \(j = 0\) is used for interior points depends on whether the second order option (c.f., eq. (3.19)) is used. Comparing Figs. C-4 and C-5 with C-2, we see that the CFL condition (C.1) for the points on the boundaries \(X = 0\) or 1 is satisfied if \(\Delta Z\) is chosen as if these points were interior points (i.e., using (C.11)). Therefore in order to insure that (C.1) is satisfied for all the computational points, we take the smallest \(\Delta Z\) obtained from (C.11).
Fig. C-4. Domains of dependence of numerical scheme for wall points ($X = 0$)

Fig. C-5. Domain of dependence of numerical scheme for bow shock points ($X = 1$)
6. REFERENCES


PART II: PROGRAMMING
7. GENERAL REMARKS

This part of the report contains a description of a FORTRAN program based on the numerical methods discussed in Part I.

The program consists of a main or executive routine, referred to as MAIN*, and several subroutines. MAIN contains the complete program structure and controls all aspects of the calculation. Most of the actual operations, however, are carried out in the subroutines many of which are called directly from MAIN. For the purpose of discussion, the subroutines are grouped into three functional types; subroutines used in the flow field calculation (see Sec. 10), auxiliary subroutines (see Sec. 11), and input-output subroutines (see Sec. 12). As a general rule, the transfer of arguments between the various subroutines and MAIN is via a COMMON block of variables. A table (see Sec. 8) is included which gives for each variable in COMMON, its description in terms of Part I notation (where possible), the primary routine in which it is defined, and the name of its COMMON block. The two COMMON blocks CSERCH and SAVRG, being used only in the real gas subroutines RGAS, HRGAS, and SERCH, are not included in this table. MAIN and all subroutines are described in Sections 9-12 which follow. These descriptions are intended to serve as a guide to the use of the FORTRAN listings which can be found in Appendix D.

In the following sections, we will occasionally refer to the second volume of this work.** This will be referred to herein as the "User's Manual".

*To distinguish between subroutine names and FORTRAN arrays or variable names, subroutine names are capitalized and underlined.

### 8. COMMON

<table>
<thead>
<tr>
<th>FORTRAN Symbol</th>
<th>Description (in Part I notation)</th>
<th>Principal Defining Routine</th>
<th>Common Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACH</td>
<td>$M_{\infty}$</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>ALNS</td>
<td>$\alpha_n = \text{ACONE}(1)$, see fig. 8 of User's Manual</td>
<td>BODYR</td>
<td>CBENT</td>
</tr>
<tr>
<td>ASQ(N,M)</td>
<td>$a^2(X_n,Y_m)$</td>
<td>DECODE</td>
<td>BLANK</td>
</tr>
<tr>
<td>ATTA</td>
<td>$\alpha$</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>AUQN</td>
<td>parameter used for bent cone calcs.</td>
<td>BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>AUQN</td>
<td>parameter used for bent cone calcs.</td>
<td>BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>AX</td>
<td>$a$, sound speed</td>
<td>RGAS</td>
<td>RGASS</td>
</tr>
<tr>
<td>A2(J)</td>
<td>$\sqrt{1 + b_z^2 + (b_{\phi}/b)^2} = \nu_{\phi}$, (1)$^+$</td>
<td>BODYP</td>
<td>BLK03$^{++}$</td>
</tr>
<tr>
<td>A3(J)</td>
<td>$[b_{\phi}/b - (b_{\phi}/b)^2]Y_{\phi}$, (1)</td>
<td>BODYP</td>
<td>BLK03</td>
</tr>
<tr>
<td>A4(J)</td>
<td>$b_{z\phi}/b - (Y_{z}/Y_{\phi})(b_{\phi}/b - (b_{\phi}/b)^2) - (b_{z}/b)(b_{\phi}/b)$, (1)</td>
<td>BODYP</td>
<td>BLK03</td>
</tr>
<tr>
<td>A5(J)</td>
<td>$b_{z\phi}/Y_{\phi}$, (1)</td>
<td>BODYP</td>
<td>BLK03</td>
</tr>
<tr>
<td>A7(J)</td>
<td>$[b_{zz} - b_{z\phi}(Y_{z}/Y_{\phi})]$, (1)</td>
<td>BODYP</td>
<td>BLK03</td>
</tr>
<tr>
<td>B(M)</td>
<td>$b(Y_m)$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BETA</td>
<td>parameter used for bent cone calcs.</td>
<td>BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>BJ</td>
<td>$-\delta$</td>
<td>MAIN</td>
<td>CEVAL</td>
</tr>
<tr>
<td>BPHI(M)</td>
<td>$b_{\phi}(Y_m)$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BPHIO(M)</td>
<td>$b_{\phi}(Y_m)$ at previous step</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
<tr>
<td>BPHIT(J)</td>
<td>temporary storage for $b_{\phi}$, (1)</td>
<td>BODYP</td>
<td>CBODY</td>
</tr>
<tr>
<td>BPHPHI</td>
<td>$b_{\phi\phi}$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BPHPHO(M)</td>
<td>$b_{\phi\phi}(Y_m)$ at previous step</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
</tbody>
</table>

$^+$Numbers in parenthesis refer to the remarks appearing at the end on this list.

$^{++}$The symbol $\phi$ here is used to denote zero.
<table>
<thead>
<tr>
<th>FORTRAN Symbol</th>
<th>Description (in Part I notation)</th>
<th>Principal Defining Routine</th>
<th>Common Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPHPHT(J)</td>
<td>temporary storage for $b_{\phi\phi}$, (1)</td>
<td>MAIN</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BSN</td>
<td>$\sin(\text{ALNS})$</td>
<td>BODYR</td>
<td>CBENT</td>
</tr>
<tr>
<td>BZ(M)</td>
<td>$b_z(M)$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BZO(M)</td>
<td>$b_z(M)$ at previous step</td>
<td>MAIN</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BZPHI</td>
<td>$b_{z\phi}$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BZPHIO(M)</td>
<td>$b_{z\phi}(Y_m)$ at previous step</td>
<td>MAIN</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BZPHIT(J)</td>
<td>temporary storage for $b_{z\phi}$, (1)</td>
<td>BODYP</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BZT(J)</td>
<td>temporary storage for $b_z$, (1)</td>
<td>BODYP</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BZZ</td>
<td>$b_{zz}$</td>
<td>BODY</td>
<td>CBODY</td>
</tr>
<tr>
<td>BZZO(M)</td>
<td>$b_{zz}(Y_m)$ at previous step</td>
<td>MAIN</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>BZZT(J)</td>
<td>temporary storage for $b_{zz}$, (1)</td>
<td>BODYP</td>
<td>CBODYYP</td>
</tr>
<tr>
<td>C(M)</td>
<td>$c(Y_m)$</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CE(I,N,L)</td>
<td>$E$, (2) and (3)</td>
<td>EVAL</td>
<td>BLK01</td>
</tr>
<tr>
<td>CENUF</td>
<td>parameter used for bent cone calcs.</td>
<td>INPUT</td>
<td>CBENT</td>
</tr>
<tr>
<td>CF(I,N,L)</td>
<td>$F$, (2) and (3)</td>
<td>EVAL</td>
<td>BLK01</td>
</tr>
<tr>
<td>CFL</td>
<td>stability factor; i.e.,</td>
<td>EVAL</td>
<td>CEVAL</td>
</tr>
<tr>
<td></td>
<td>$\max \left{ u/(w^2 - a^2) \right}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CG(I,N,J)</td>
<td>$G$, (1) and (2)</td>
<td>EVAL, EVALSY</td>
<td>BLK01</td>
</tr>
<tr>
<td>COSBN</td>
<td>$\cos(\text{THETABN})$</td>
<td>BODYSR</td>
<td>CBENT</td>
</tr>
<tr>
<td>COSPHI(M)</td>
<td>$\cos(\phi(Y_m))$</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
<tr>
<td>CPHI(M)</td>
<td>$c_\phi(Y_m)$ or $\frac{3c}{3Y}(Y_m)$</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CPHTY(J)</td>
<td>temporary storage for $c_\phi$, (1)</td>
<td>EVAL, EVALSY, EVALPR</td>
<td>BLK01</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Defining Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------</td>
<td>-----------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>CU(I,N,M); N#1,NC</td>
<td>( U(x_n, y_m) ), (2)</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CU(I,1,M)</td>
<td>at wall ( I = 1 ); ( s ) at wall ( I = 2 ); ( v_2 ) at wall ( I = 3 )</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CU(I,NC,M)</td>
<td>( c ) ( (I=1) ), ( c_z ) ( (I=2) ), ( c_\phi ) ( (I=3) )</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CUP(I,N,M); N#1,NC</td>
<td>( u^*(x_n, y_m) ), (2) and (4)</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CUP(I,1,M)</td>
<td>( p^* ) at wall ( I = 1 ); ( s^* ) at wall ( I = 2 ); ( v_2^* ) at wall ( I = 3 ), (4)</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CUP(I,NC,M)</td>
<td>( c^* ) ( (I=1) ), ( c_z^* ) ( (I=2) ), ( c_\phi^* ) ( (I=3) ), (4)</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>CZ(M)</td>
<td>( c_z(y_m) )</td>
<td>DECODE</td>
<td>BLANK</td>
</tr>
<tr>
<td>CZY(J)</td>
<td>temporary storage for ( c_z ), (1)</td>
<td>EVAL, EVALSY, EVALPR</td>
<td>BLK01</td>
</tr>
<tr>
<td>D(N,M)</td>
<td>( \rho(x_n, y_m) )</td>
<td>EVAL</td>
<td>BLANK</td>
</tr>
<tr>
<td>DCPHZ</td>
<td>( \frac{\partial c_\phi}{\partial z} )</td>
<td>SHOCK</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>DCZ</td>
<td>( \frac{\partial c}{\partial z} )</td>
<td>SHOCK</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>DCZZ</td>
<td>( \frac{\partial c_z}{\partial z} )</td>
<td>SHOCK</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>DDX</td>
<td>( 1/\Delta x )</td>
<td>MAIN</td>
<td>BLK04</td>
</tr>
<tr>
<td>DDY</td>
<td>( 1/\Delta y )</td>
<td>MAIN</td>
<td>BLK04</td>
</tr>
<tr>
<td>DELII</td>
<td>parameter used for bent cone calcs.</td>
<td>INPUT</td>
<td>CBENT</td>
</tr>
<tr>
<td>DELTA</td>
<td>parameter used for bent cone calcs.</td>
<td>BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>DELZ</td>
<td>intercept value of cone</td>
<td>BODY, BODYR</td>
<td>CBODY</td>
</tr>
<tr>
<td>DINF</td>
<td>( \rho_\infty )</td>
<td>INPUT</td>
<td>BLANK</td>
</tr>
<tr>
<td>DINF2</td>
<td>( \rho_\infty^2 )</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Defining Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------------------------------</td>
<td>----------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>DINX</td>
<td>$V_\infty \cos \theta \cos \alpha \sqrt{\rho_\infty / \rho_\infty}$</td>
<td>MAIN</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>DST</td>
<td>parameter used for bent cone calcs.</td>
<td>BODYR</td>
<td>CBENT</td>
</tr>
<tr>
<td>DY</td>
<td>$\Delta Y$</td>
<td>MAIN</td>
<td>BLK02</td>
</tr>
<tr>
<td>DYD3</td>
<td>$\Delta Y/3$</td>
<td>MAIN</td>
<td>CINTEG</td>
</tr>
<tr>
<td>DZ</td>
<td>$\Delta Z$</td>
<td>MAIN</td>
<td>CBODYP</td>
</tr>
<tr>
<td>D1INF</td>
<td>$V_\infty \cos \theta \sin \alpha \sqrt{\rho_\infty / \rho_\infty}$</td>
<td>MAIN</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>D2INF</td>
<td>$V_\infty \sin \theta \sqrt{\rho_\infty / \rho_\infty}$</td>
<td>MAIN</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>ELIM</td>
<td>error limit for real gas iterative procedures</td>
<td>INPUT</td>
<td>BLK04</td>
</tr>
<tr>
<td>EPSQ</td>
<td>parameter used for bent cone calcs.</td>
<td>BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>FA</td>
<td>$F_a$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>FAZ</td>
<td>$\frac{\partial F_a}{\partial z}$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>FN</td>
<td>$F_n$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>FNZ</td>
<td>$\frac{\partial F_n}{\partial z}$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>FY</td>
<td>$F_y$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>FYZ</td>
<td>$\frac{\partial F_y}{\partial z}$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>GAMMA</td>
<td>$\Gamma$, (5)</td>
<td>INPUT, DECODE</td>
<td>BLK04</td>
</tr>
<tr>
<td>GA2</td>
<td>$2\Gamma_\infty / (\Gamma_\infty - 1)$, (5)</td>
<td>MAIN, JUMP</td>
<td>BLK04</td>
</tr>
<tr>
<td>GB</td>
<td>$1 / (\Gamma_\infty - 1)$</td>
<td>MAIN</td>
<td>BLK04</td>
</tr>
<tr>
<td>GC(N,M)</td>
<td>parameter used in real gas iterative procedures</td>
<td>DECODE</td>
<td>CDECODE</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Defining Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------</td>
<td>----------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>GD</td>
<td>$(\Gamma - 1)/2, (5)$</td>
<td>MAIN, DECODE</td>
<td>BLK04</td>
</tr>
<tr>
<td>GE</td>
<td>$(\Gamma + 1)/2, (5)$</td>
<td>MAIN, DECODE</td>
<td>BLK04</td>
</tr>
<tr>
<td>GFF</td>
<td>$\Gamma + 1, (5)$</td>
<td>MAIN, DECODE</td>
<td>CDECODE</td>
</tr>
<tr>
<td>GG</td>
<td>$\Gamma^2 - 1, (5)$</td>
<td>MAIN, DECODE</td>
<td>CDECODE</td>
</tr>
<tr>
<td>GM(N,M)</td>
<td>$\Gamma$</td>
<td>MAIN, DECODE</td>
<td>BLK04</td>
</tr>
<tr>
<td>GIMI</td>
<td>$1 - \Gamma_\infty$</td>
<td>MAIN, CDECODE</td>
<td></td>
</tr>
<tr>
<td>GX</td>
<td>$\Gamma$ or $\gamma$</td>
<td>RGAS, RGASS</td>
<td></td>
</tr>
<tr>
<td>GY(M)</td>
<td>$T_{n-m}$</td>
<td>MAIN, CINTEG</td>
<td></td>
</tr>
<tr>
<td>GYMDY</td>
<td>$g_y$ at $Y = -\Delta Y$</td>
<td>TRANGD, CTRANG</td>
<td></td>
</tr>
<tr>
<td>GYYMDY</td>
<td>$g_{yy}$ at $Y = -\Delta Y$</td>
<td>TRANGD, CTRANG</td>
<td></td>
</tr>
<tr>
<td>GY1PDY</td>
<td>$g_y$ at $Y = 1 + \Delta Y$</td>
<td>TRANGD, CTRANG</td>
<td></td>
</tr>
<tr>
<td>GYY1PDY</td>
<td>$g_{yy}$ at $Y = 1 + \Delta Y$</td>
<td>TRANGD, CTRANG</td>
<td></td>
</tr>
<tr>
<td>HINF</td>
<td>$h_\infty$</td>
<td>MAIN, CDECODE</td>
<td></td>
</tr>
<tr>
<td>HN</td>
<td>parameter used for bent cone calcs.</td>
<td>BODYR, CBENT</td>
<td></td>
</tr>
<tr>
<td>HOT2</td>
<td>$2h_\infty + V_\infty^2$</td>
<td>MAIN, BLK04</td>
<td></td>
</tr>
<tr>
<td>HX</td>
<td>$h$</td>
<td>RGAS, RGASS</td>
<td></td>
</tr>
<tr>
<td>IBN</td>
<td>= 0, spherical nose</td>
<td>INPUT, CBENT</td>
<td></td>
</tr>
<tr>
<td>ICFL</td>
<td>step count used after expansion discontinuity for option to reduce step size</td>
<td>MAIN, BLK03</td>
<td></td>
</tr>
<tr>
<td>ICHECK</td>
<td>Flag indicating predictor or corrector for DECODE</td>
<td>MAIN, CDECODE</td>
<td></td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------</td>
<td>-------------------</td>
<td>--------------</td>
</tr>
</tbody>
</table>
| IDYAW          | = 0 symmetric case ($\phi_0 = \pi$)  
                | = 1 nonsymmetric case ($\phi_0 = 2\pi$) | MAIN  | BLANK |
| IERRPR         | number of previous steps to be printed out after error termination | INPUT  | CSAVE |
| IJMPKT(M)      | step count used after expansion and compression discontinuities for $X$ derivative cancellation option | WALL  | BLKØ3 |
| IJUMP(M)       | flag used to indicate that a discontinuity in body slope has been sensed | BODYP, MAIN  | BLKØ3 |
| IJUMP1(M)      | flag used to control options in WALL when body slope discontinuity has been encountered | JUMP, BODYP, WALL  | BLKØ3 |
| ISWMOD         | flag used to select options for wall point calculation | INPUT  | CWALL |
| ISWSMO         | for $0 \leq M \leq$ ISWSMO, entropy at wall is defined by extrapolation | INPUT  | BLKØ4 |
| JCFL           | = 1, 2, or 3; tells which of $\mu_1$, $\mu_2$, $\mu_3$ determines stability cond. | EVAL  | CEVAL |
| K              | step count | MAIN  | BLANK |
| KCFL           | number of steps to reduce step size after an expansion discontinuity | INPUT  | CWALL |
| KFAC           | step size is reduced by $\Delta z$/KFAC after an expansion discontinuity | INPUT  | CWALL |
| LCNT           | Maximum number of real gas iterations | INPUT  | BLKØ4 |
| MA             | MC-1 | MAIN  | CINTEG |
| MAS            | = MC (IDYAW=0)  
<pre><code>            | = MC-1 (IDYAW=1) | MAIN  | CSAVE |
</code></pre>
<table>
<thead>
<tr>
<th>FORTRAN Symbol</th>
<th>Description (in Part I notation)</th>
<th>Principal Defining Routine</th>
<th>Common Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC</td>
<td>M, the number of points in the Y direction</td>
<td>INPUT</td>
<td>BLANK</td>
</tr>
<tr>
<td>MCFL</td>
<td>value of m where stability condition is determined</td>
<td>EVAL</td>
<td>CEVAL</td>
</tr>
<tr>
<td>MCP</td>
<td>MC+1</td>
<td>MAIN</td>
<td>CTRANG</td>
</tr>
<tr>
<td>MCMAX</td>
<td>max. φ pts. in dimension statements</td>
<td>DATA Card</td>
<td></td>
</tr>
<tr>
<td>MOD1</td>
<td>= 1, second order accuracy for wall points = 0, first order accuracy for wall points</td>
<td>INPUT, WALL</td>
<td>CWALL</td>
</tr>
<tr>
<td>MX (REAL)</td>
<td>$M^n_c (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>MXZ (REAL)</td>
<td>$\frac{\partial M^n_c}{\partial z} (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>MY (REAL)</td>
<td>$M^c_y (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>MYZ (REAL)</td>
<td>$\frac{\partial M^c_y}{\partial z} (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>MZ (REAL)</td>
<td>$M^c_a (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>MZZ (REAL)</td>
<td>$\frac{\partial M^c_a}{\partial z} (z_c = 0)$</td>
<td>INTEG</td>
<td>CINTEG</td>
</tr>
<tr>
<td>NA</td>
<td>NC-1</td>
<td>MAIN</td>
<td>BLK04</td>
</tr>
<tr>
<td>NC</td>
<td>N, the number of points in the X direction</td>
<td>INPUT</td>
<td>BLANK</td>
</tr>
<tr>
<td>NCFL</td>
<td>value n where stability condition is determined</td>
<td>EVAL</td>
<td>CEVAL</td>
</tr>
<tr>
<td>NCMAX</td>
<td>max. r pts. in dimension statements</td>
<td>DATA Card</td>
<td></td>
</tr>
<tr>
<td>NFIRST</td>
<td>flag used in RGAS</td>
<td>MAIN, RGAS</td>
<td>RGASS</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------</td>
<td>-------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>NJMKTC</td>
<td>max. number of steps to modify X derivatives at wall after a compression discontinuity</td>
<td>INPUT, MAIN</td>
<td>CWALL</td>
</tr>
<tr>
<td>NJMPKT</td>
<td>max. number of steps to modify X derivatives at wall after an expansion discontinuity</td>
<td>INPUT, MAIN</td>
<td>CWALL</td>
</tr>
<tr>
<td>NGAS</td>
<td>flag to determine gas mixture in RGAS</td>
<td>INPUT</td>
<td>RGASS</td>
</tr>
<tr>
<td>NSGD</td>
<td>number of $\phi$ values to be read in</td>
<td>INPUT</td>
<td>CTRANG</td>
</tr>
<tr>
<td>NSFD</td>
<td>number of $\bar{x}$ values to be read in</td>
<td>INPUT</td>
<td>CTRANF</td>
</tr>
<tr>
<td>NTARGET</td>
<td>number of target points for printout to be read in</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>NTEST</td>
<td>$\geq 0$ then perfect gas $&lt; 0$ then real gas</td>
<td>INPUT</td>
<td>RGASS</td>
</tr>
<tr>
<td>P(N,M)</td>
<td>$p(X_n, Y_m)$</td>
<td>EVAL</td>
<td>BLANK</td>
</tr>
<tr>
<td>PDIF</td>
<td>$P_\infty / P_\infty$</td>
<td>MAIN</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>PHI(M)</td>
<td>$\phi(Y_m)$</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
<tr>
<td>PHIO</td>
<td>$\phi_o (= \pi$ or $2\pi$)</td>
<td>INPUT</td>
<td>BLANK</td>
</tr>
<tr>
<td>PHIJ, PHI2J</td>
<td>$\phi$ interval to turn JUMP subroutine off</td>
<td>INPUT</td>
<td>CBODYP</td>
</tr>
<tr>
<td>PI</td>
<td>$\pi$</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>PID2</td>
<td>$\pi/2$</td>
<td>BODYR</td>
<td>CBENT</td>
</tr>
<tr>
<td>PINF</td>
<td>$P_\infty$</td>
<td>INPUT</td>
<td>BLANK</td>
</tr>
<tr>
<td>PWY(J)</td>
<td>temporary storage for $p$ at wall, (1)</td>
<td>EVAL, EVALSY</td>
<td>BLK01</td>
</tr>
<tr>
<td>PZ</td>
<td>$\frac{3p}{2}$ at wall</td>
<td>WALL</td>
<td>CWALL</td>
</tr>
<tr>
<td>PZCOR(M)</td>
<td>$\tilde{\phi}^k_{1,m}$</td>
<td>WALL</td>
<td>CWALL</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Defining Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------</td>
<td>----------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>R(N,M)</td>
<td>( r(X_n, Y_m) )</td>
<td>TRANF</td>
<td>BLANK</td>
</tr>
<tr>
<td>RAD</td>
<td>( \pi/180^\circ )</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>RRX</td>
<td>gas constant</td>
<td>RGAS</td>
<td>RGASS</td>
</tr>
<tr>
<td>SFD(N)</td>
<td>( x = f(X_n) )</td>
<td>TRANFD</td>
<td>CTRANF</td>
</tr>
<tr>
<td>SFXD(N)</td>
<td>( f_X )</td>
<td>TRANFD</td>
<td>CTRANF</td>
</tr>
<tr>
<td>SFXXD(N)</td>
<td>( f_{XX} )</td>
<td>TRANFD</td>
<td>CTRANF</td>
</tr>
<tr>
<td>SGD(M)</td>
<td>( \phi = \phi_0 g(Y_m) )</td>
<td>TRANGD</td>
<td>CTRANG</td>
</tr>
<tr>
<td>SGYD(M)</td>
<td>( g_Y )</td>
<td>TRANGD</td>
<td>CTRANG</td>
</tr>
<tr>
<td>SGYD(M)</td>
<td>( g_{YY} )</td>
<td>TRANGD</td>
<td>CTRANG</td>
</tr>
<tr>
<td>SINF</td>
<td>( s_\infty )</td>
<td>MAIN</td>
<td>BLANK</td>
</tr>
<tr>
<td>SINPHI(M)</td>
<td>( \sin[\phi(Y_m)] )</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
<tr>
<td>SPDIF</td>
<td>( \sqrt{\rho_\infty/\rho_\infty} )</td>
<td>MAIN</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>SW(M)</td>
<td>( s ) at wall</td>
<td>DECODE</td>
<td>BLK04</td>
</tr>
<tr>
<td>SWY(J)</td>
<td>temporary storage for ( s ) at wall, (1)</td>
<td>EVAL, EVAULSY</td>
<td>BLK01</td>
</tr>
<tr>
<td>SZ</td>
<td>( \frac{3s}{\sqrt{2}} ) at wall, (6)</td>
<td>WALL</td>
<td>CWALL</td>
</tr>
<tr>
<td>TANBN</td>
<td>( \tan(\text{THETABN}) )</td>
<td>BODYR</td>
<td>CBENT</td>
</tr>
<tr>
<td>TANCO</td>
<td>( \tan(\text{CONE}) )</td>
<td>BODY, BODYR</td>
<td>CBODY</td>
</tr>
<tr>
<td>TARGETZ (100)</td>
<td>( Z ) values for targeting printout</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>TF4(N,L)</td>
<td>( T_{f_4} ), (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>TF6(N,L)</td>
<td>( T_{f_6} ), (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>TF7(N,L)</td>
<td>( T_{f_7} ), (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>FORTRAN Symbol</td>
<td>Description (in Part I notation)</td>
<td>Principal Defining Routine</td>
<td>Common Block</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------</td>
<td>-----------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>TG4(J)</td>
<td>$\varepsilon_Y$</td>
<td>TRANG</td>
<td>BLK02</td>
</tr>
<tr>
<td>TG5(J)</td>
<td>$\varepsilon_5 = \varepsilon_{YY}/\varepsilon_Y$, (1)</td>
<td>TRANG</td>
<td>BLK02</td>
</tr>
<tr>
<td>TG6(J)</td>
<td>$\varepsilon_{YZ}/\varepsilon_Y$, (1)</td>
<td>TRANG</td>
<td>BLK02</td>
</tr>
<tr>
<td>THETA</td>
<td>$\gamma = \gamma/\gamma_o$</td>
<td>TRANG</td>
<td>BLK02</td>
</tr>
<tr>
<td>THETABN</td>
<td>bend angle in bent cone calcs.</td>
<td>INPUT</td>
<td>CBENT</td>
</tr>
<tr>
<td>TX</td>
<td>temperature</td>
<td>RGAS</td>
<td>RGASS</td>
</tr>
<tr>
<td>U(N,M)</td>
<td>$u(X_n,Y_m)$</td>
<td>DECODE</td>
<td>BLANK</td>
</tr>
<tr>
<td>UNOR(N,L)</td>
<td>$(N = 1,2,3$, only) $A$, (3)</td>
<td>EVAL</td>
<td>BLK01</td>
</tr>
<tr>
<td>UZCOR(I,M)</td>
<td>$F_k^{N,m} + F_k^{N-2,m} - 2F_k^{N-1,m}$</td>
<td>SHOCK</td>
<td>CSHOCK</td>
</tr>
<tr>
<td>V(N,M)</td>
<td>$v(X_n,Y_m)$</td>
<td>DECODE</td>
<td>BLANK</td>
</tr>
<tr>
<td>VINF</td>
<td>$V_o$</td>
<td>MAIN</td>
<td>COUT</td>
</tr>
<tr>
<td>VWY(J)</td>
<td>$v/w$ at wall, (1)</td>
<td>EVAL,</td>
<td>BLK01</td>
</tr>
<tr>
<td>V1INF</td>
<td>$V_o \sin \alpha \cos \beta$</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>V2(J)</td>
<td>$V_2$ at wall, (1)</td>
<td>EVAL,</td>
<td>BLK01</td>
</tr>
<tr>
<td>V2INF</td>
<td>$V_o \sin \beta$</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>V2Z</td>
<td>$3V_2/3Z$ at wall</td>
<td>WALL</td>
<td>CWALL</td>
</tr>
<tr>
<td>W(N,M)</td>
<td>$w(X_n,Y_m)$</td>
<td>DECODE</td>
<td>BLANK</td>
</tr>
<tr>
<td>WINX</td>
<td>$V_o \cos \alpha \cos \beta$</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>X(N)</td>
<td>$X_n$</td>
<td>MAIN</td>
<td>BLK02</td>
</tr>
<tr>
<td>XINDEF</td>
<td>Undefined quantity</td>
<td></td>
<td>COUT</td>
</tr>
</tbody>
</table>
# FORTRAN Defining Common Symbol Description

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description (in Part I notation)</th>
<th>Principal Routine</th>
<th>Common Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPHI(N,L)</td>
<td>$X_{\phi}$ (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>XR(N,L)</td>
<td>$X_{r}$ (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>XZ(N,L)</td>
<td>$X_{z}$ (3)</td>
<td>TRANF</td>
<td>BLK02</td>
</tr>
<tr>
<td>XIKINEQ</td>
<td>$\Theta$</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>XIKINP2</td>
<td>$\Theta + 2$</td>
<td>MAIN</td>
<td>CDECODE</td>
</tr>
<tr>
<td>Y(M)</td>
<td>$Y_{m}$</td>
<td>MAIN</td>
<td>BLK02</td>
</tr>
<tr>
<td>YAW</td>
<td>$\beta$</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>YPHI(J)</td>
<td>$Y_{\phi}^{*}$ (1)</td>
<td>TRANG</td>
<td>BLANK</td>
</tr>
<tr>
<td>YZ(J)</td>
<td>$Y_{z}^{*}$ (1)</td>
<td>TRANG</td>
<td>BLANK</td>
</tr>
<tr>
<td>Z</td>
<td>$z_{k+1}$ or $z_{k}$</td>
<td>MAIN</td>
<td>CBODY</td>
</tr>
<tr>
<td>ZBB(M)</td>
<td>parameters used in bent cone calcs.</td>
<td>MAIN, BODY</td>
<td>CBENT</td>
</tr>
<tr>
<td>ZEND</td>
<td>$z_{end}$, final z value</td>
<td>INPUT</td>
<td>COUT</td>
</tr>
<tr>
<td>ZMAXS</td>
<td>parameter used in bent cone calcs.</td>
<td>BODY, BODYR</td>
<td>CBENT</td>
</tr>
</tbody>
</table>

### Remarks:

1. These quantities are not fully stored. The J index identifies a line $Y = constant$ and is either 1, 2, or 3. For quantities indexed as $(N,J)$, the index $N$ refers to $X = X_{n}$.

2. Quantities indexed $(I,N,M)$, $(I,N,J)$, or $(I,N,L)$ are 4 dimensional column vectors. The index $I = 1, 2, 3, or 4$ indicates the component (from top to bottom). The exceptions are $CUP(I,I,M)$, $CU(I,I,M)$, $CUP(I,NC,M)$, and $(CUI,NC,M)$.

3. These quantities are not fully stored. The L index identifies a line $Y = constant$ and is either 1 or 2. For quantities indexed as $(N,L)$, the index $N$ refers to $X = X_{n}$.

4. At certain points in the MAIN, these locations store the numerical $Z$-derivatives of the indicated quantities for the predictor step.

5. In real gas calculations, these quantities are defined with $\Gamma = \Gamma_{\infty}$ in MAIN. In perfect gas calculations, these quantities are constants.
9. MAIN

MAIN is divided into two sections. Section 1 is comprised of all operations performed at the initial entry to the program; hence, this section is executed only once in the entire calculation. Section 2 contains the predictor-corrector marching scheme and is, therefore, executed repeatedly. Each cycle through this section corresponds to one marching step of the calculation. The basic operation of MAIN, and hence the entire program, is shown schematically in Fig. 8. Each rectangle in the figure represents a major subsection of the program. Note well, these functional rectangles can be easily identified in the listing by locating the corresponding comment cards. The individual subsections are described in Secs. 9.1 and 9.2 below.

9.1 Section 1

Input - The initial flow field data is input from tape and rezoned if necessary. Also, various program controls and parameters are input from cards. Specific instructions and descriptions of both of these inputs are given in the User's Manual.

Initializations and Parameters - Various fixed parameters used throughout the calculation are computed and the X, Y, CU, and ASQ and GM arrays are initialized (except for N=1 and NC) using the initial flow field data.

Preliminary Predictor Loop - This loop computes the derivatives $\frac{\partial C}{\partial z}$, $\frac{\partial \phi}{\partial z}$, $\frac{\partial U}{\partial z}$, $\frac{\partial P}{\partial z}$, $\frac{\partial V}{\partial z}$, $\frac{\partial S}{\partial z}$ which are required in the predictor for the first step. These derivatives are determined using (3.9a) - (3.9c), (3.6a), (3.16) - (3.18); their values are stored in the CUP and CP arrays.
Fig. 8. Flow chart of MAIN
Here the CU, ASQ, and GM arrays (see sec. 8) for N=1 and NC are initialized using wall and shock quantities from the initial data. In each pass through this loop, the above operations are performed on the adjacent lines M-1 and M*. The quantities K1,K2 (= 1 or 2) are indices which identify certain quantities determined in the previous pass through the loop (see the comment statements in the listings for more details). When the loop has been completed, the stability parameter, CFL, for calculating the step size for the first step has been determined (in EVAL).

9.2 Section 2

ΔZ,new Z - In this subsection, the step size ΔZ is determined using the value of the stability parameter, CFL, obtained in the previous cycle or, in the case of the first step, in the Preliminary Predictor Loop. After an expansion discontinuity, ΔZ can be reduced if this option is selected by the user. The new value of Z is then obtained by incrementing the previous value of Z by ΔZ. This subsection also contains tests to determine if an axes shift is necessary in bent cone calculations (see User's Manual for details).

Predictor Update Loop - In this loop the predicted values of the conservation vector U, the wall quantities P,V₂,s, and the shock geometry c,c₂,c₆ for the new value of Z are determined using (3.6a), (3.13), and (3.10). Recall that the Z-derivatives appearing in these equations were computed in the previous cycle or, in the case of the first step, in the Preliminary Predictor Loop and were stored in the CUP and CP arrays. In the execution of the loop, the predicted values, as they are determined, replace the derivative values in these arrays.

*The set of computational points \{X(N),Y(M)\} where M is fixed and N=1,2,\ldots,NC will be referred to as the line M.
Corrector-Predictor Loop - This loop contains the major part of the marching step calculation. In the loop, various operations are performed on the adjacent lines M, M-1, and M-2. The quantities K1,K2 (= 1, or 2) and J1,J2,J3 (= 1, 2, or 3) are indices which identify certain quantities determined in previous passes through the loop (see, the comment statements in the listing for more details). In each pass through the loop, the following operations are performed in the sequence indicated**:

(i.) On the line M, the predicted values of the flow variables $\rho, p, u, v, w, a^2$ are determined from the predicted values of $U, P, V_2, S, c_0, c_2, c$ (c.f., DECODE, see Sec. 10.2). Note that the predicted values of $U, P, V_2, S, c_0, c_2, c$ were determined in the Predictor Update Loop.

(ii.) The corrected value of $c$ at $Y = Y(M)$ is determined using

(3.11).

(iii.) On the line M-1, the corrected values of the conservation vector $U$, the wall quantities $P, V_2, S$, and the shock geometry $c_0, c_2$ are determined using (3.6b), (3.14), (3.11). These are stored in the CU array.

(iv.) On the line M-1, the corrected values of $p, \rho, u, v, w, a^2$ are determined using the quantities obtained in (iii), and the corrected value of $c$ determined in (ii.) for the previous pass through the loop; c.f., DECODE, Sec. 10.2. Note that the corrected values obtained here replace the

---


**The sequence of operations indicated here is necessarily modified for the first and last two passes through the loop (see the listing for details).
predicted values in the \( P, D, U, V, W, ASQ, \Phi, Z \) arrays on the line \( M-1 \). The predicted values on \( M-1 \) are no longer needed in the calculation.

(v.) If for \( Y = Y(M-1) \), a discontinuity in body slopes has been found (by \texttt{BODYP}) , the true body geometry is substituted for the modified geometry (in \texttt{BODYPP}) and discontinuities in the surface flow variables are computed (in \texttt{JUMP}). This procedure is discussed in detail in Sec. 4.1.

(vi.) On the line \( M-1 \), the local stability parameters \( \mu/(w^2 - a^2) \) (c.f., Sec. 3.6) are computed using the corrected values of the flow variables and the maximum of these parameters taken over all previously computed lines is up-dated to include the line \( M-1 \) (in \texttt{EVAL}). Note that the stability parameters being considered here are for the next \( Z \) step.

(vii.) On the line \( M-2 \), the derivatives \( \frac{\partial U}{\partial Z}, \frac{\partial P}{\partial Z}, \frac{\partial V}{\partial Z}, \frac{\partial W}{\partial Z}, \frac{\partial ASQ}{\partial Z}, \frac{\partial \Phi}{\partial Z} \) required for the predictor for the next step are computed using (3.6a), (3.16)-(3.18), and (3.9a)-(3.9c). These quantities are stored in the \texttt{CUP} and \texttt{CP} arrays.

Note that after the Predictor-Corrector Loop has been completed, the arrays \( C, CZ, \Phi, P, U, V, W, ASQ \) contain corrected values of the corresponding quantities at the new value of \( Z \). The arrays \texttt{CUP} and \texttt{CP} contain the values of the \( Z \) derivatives required for the predictor step of the next cycle. Furthermore, the stability parameter, \( CFL \), for determining the size of the next step has been determined.

**Force and Moments, Output** - On this subsection, the aerodynamic force and moment results are computed (in \texttt{INTEG}) and outputs are performed.

The flow variables and the force and moment results for the current step
are output to a binary tape (TAPE 16). Also, if selected by the user, the current flow field results are output on-line (in FIELD). The various options available for on-line printout are discussed in the User's Manual.

If the calculation is to be continued (i.e., the current value of Z is less than ZEND and the step count K is less than the maximum number of steps selected by the user), the control is returned to the beginning of Section 2 and the cycle for the next step is performed. If, on the other hand, the calculation is to be terminated, the current flow variables and the force and moment quantities are output to a binary tape (TAPE 17). This tape serves as an input tape if the calculation is to be restarted from the current station in a different run (see the User's Manual for instructions on restarting). The final operation before termination is the on-line printout of the surface pressure data and the force and moment coefficients (see OUT, Sec. 12.3).
10. SUBROUTINES USED IN THE FLOW FIELD CALCULATION

The subroutines discussed in this section are all called from MAIN in the Preliminary Predictor Loop and the Corrector-Predictor Loop. These two loops and the subroutines of this section constitute the complete algorithm for computing one marching step of the flow field calculation. All the subroutines in this section are line operations; i.e., when the subroutine operation depends on X and Y, all points X(N), Y(M) where M is fixed and N = 1,2,\ldots,NC are considered for each entry. When the subroutine operation depends exclusively on Y, only the value Y = Y(M) is considered for each entry. Generally, the inputs to and outputs from these subroutines are via the COMMON BLOCK (see Sec. 8). Thus the arguments in the calling sequences of these subroutines only contain indices which identify the input/output quantities on the particular line for which the subroutines are to operate (the exceptions to this are DECODE (see, Sec. 10.2) and JUMP (see, Sec. 10.4).

10.1 BODYP, ENTRY BODYP

Calling sequence: Call BODYP(M,J3)

where M is the line index; i.e., Y = Y(M).

J3 (= 1,2, or 3) is the identification index corresponding to the line M for the body parameters A1,A2,\ldots,A5,A7.

Description: This routine defines the body parameters A1(J3),\ldots,A5(J3),A7(J3) for the line M (for definitions of these quantities, see Sec. 8). These quantities are computed using the body geometry contained in B(M), BPHI(M), BZ(M), BZZ, BZPHI, and BPHIPHI (determined in BODY). Note that BODYP is always called after BODY in the MAIN.
This routine also tests for discontinuities in body slope (see Sec. 4.1 for a discussion of the procedures used when such discontinuities are present). If a discontinuity is found and if subroutine JUMP is to be used, the flag IJUMP(M) is set to 1 and the old (previous) values of $b_2$, $b_3$, $b_4$, and $b_5$ (stored in BZO(M), BPHIO(M), etc.) are used to define $A1(J3)$, $A2(J3)$, etc. In this case, the true values of the body derivatives are stored temporarily in BZT(J3), BPHIT(J3), etc. The entry point BODYPP is used only when the JUMP routine is used. It is called from the Corrector-Predictor Loop for the purpose of redefining the arrays $A1(J3)$, $A2(J3)$, etc. using the true values of the body derivatives.

10.2 DECODE

Calling sequence: CALL DECODE (M, CV, J, NDIM, MDIM)

where

- $M$ is the line index; i.e., $Y = Y(M)$
- $CV$ is either the CU or CUP array
- $J (= 1, 2, \text{ or } 3)$ is the identification index corresponding to the line M for the body parameter $A2$
- $NDIM$ is the value of the dimension for the N index of CV
- $MDIM$ is the value of the dimension for the M index of CV

Description: This routine defines the flow variable arrays $P, D, U, V, W, ASQ$ and the shock arrays $C, CZ, CPHI$ for the line $M$. The flow quantities at the wall ($N = 1$) are determined from the values of $P$, $V_2$, and $s$ contained in the CV array for $N = 1$ using $p = \exp(P)$, (2.4b), and (3.15). For interior points $1 < N < NC$, the flow quantities are determined from the values of the
conservation vector contained in the CV array using the procedure described in Sec. 3.2. At the shock \( N = NC \), the flow variables are determined using the Rankine-Hugoniot relations with \( c, c_v, c_z \) contained in the \( CV(1, NC, M) \), \( CV(3, NC, M) \), and \( CV(2, NC, M) \) arrays (see Sec. 3.3). The thermodynamic properties needed in these procedures are supplied from subroutines RGAS and HGAS.

This routine also contains the selective smoothing procedure given in section 4.1. When an interior point has a negative pressure, the conservation vector CV is redefined at that point using (4.11) and flow variables are recomputed.

10.3 EVAL, ENTRY EVALSY, ENTRY EVALPR

**Calling sequence:**

```
CALL EVAL (L, M, IT, JSG, JCG, JCFCE)
```

where

- **L** is 0 in the predictor and 1 in the corrector
- **M** is the line index; i.e., \( Y = Y(M) \)
- **IT** (\( = 1 \) or \( 2 \)) is the identification index corresponding to the line M for \( XZ, XR, XPHI, TF4, TF6, TF7 \)
- **JSG** (\( = 1, 2 \), or \( 3 \)) is the identification index corresponding to the line M for \( YPHI, YZ, TG4, TG5, TG6 \)
- **JCG** is the identification index corresponding to the line M for \( CG, VOWY, PWY, SWY, V2 \)
- **JCFCE** (\( = 1 \) or \( 2 \)) is the identification index corresponding to the line M for \( CF, CE, UNOR \)

**Description:** This routine determines the flux vectors \( F \) and \( G \) and the source term \( E \) on the line M using the definitions given in (3.3a), (3.3b), and (3.3c). The flow variables \( p, \rho, \) etc. used in these equations
are the current values contained in the COMMON arrays P,D, etc. on the line M. On this line, these arrays may contain predicted values (when \( L = 1 \)) or corrected values (when \( L = 0 \)). The other quantities required in (3.3a) - (3.3c) are \( XZ,XR,XPHI,TF4,TF6,TF7 \) (from COMMON using the index IT) and \( YZ,TG4,TG5,TG6 \) (from COMMON using the index JSG). The values of F and E that are obtained are stored in COMMON arrays F and E using the JCFCE index; the value of G is stored in the CG array using the index JCG.

This routine also computes and/or stores various quantities used in WALL and SHOCK. These are \( A \) for \( N = 1,2,3 \) (stored in the COMMON array UNOR using the index JCFCE), the wall values (\( N = 1 \)) of \( v/w,p,s,V \) (stored in the COMMON arrays VOWY,PWY,SWY, and V2, respectively, using the index JCC) and the shock slopes \( c_\phi,c_2 \) (stored in COMMON arrays CPHTY and CZY using the index JCG).

For each step \( Z \) of the calculation, EVAL is entered twice; once when \( L = 0 \) (predictor) and once when \( L = 1 \) (corrector). When this routine is entered with \( L = 0 \) (corrected values in P,D, etc.) the local stability parameter \( \nu/(w^2 - a^2) \) (c.f., Sec. 3.6) is computed and compared to the maximum of these quantities taken over the previously executed lines. Note that the value of CFL in COMMON after EVAL has been executed is the maximum of the local values taken over the lines 1,2,...,M.

The entry EVALSY is used in the symmetric problem to define the COMMON arrays CG,VOWY,PWY,V2,CPHTY, and CZY on the "fringe" lines corresponding to \(-\Delta Y\) and \(1 + \Delta Y\). The flow variables on these planes are determined using the symmetry conditions (3.21) and (3.22). For this entry, the argument \( M \) is 2 (for \( Y = -\Delta Y \)) and MC-1 (for \( Y = 1 + \Delta Y \)); the index IT identifies the
elements of the COMMON array TC5 corresponding to \( Y = 0 \) or \( Y = 1 \); and the index JSG identifies the elements of the arrays YZ and YPHI corresponding to \( Y = - \Delta Y \) or \( Y = 1 + \Delta Y \).

The entry EVALPR is used in the non-symmetric problem to define the COMMON arrays CG, VOWY, PWY, VZ, CPHTY, and CZY on the planes \( -\Delta Y \) and 1 using (3.24). In this entry \( M \) is MC-1 for the plane \( Y = - \Delta Y \) and 1 for the plane \( Y = 1 \); the index JCG identifies the elements of the COMMON arrays YZ and YPHI on these planes.

10.4 JUMP

Calling sequence: CALL JUMP (DBP, DBZ, MB)

where

- **DBP** is \([b_\phi/b]_1 - (b_\phi/b)_+\) (see Part I notation)
- **DBZ** is \([b_z/b]_1 - (b_z/b)_+\) (see Part I notation)
- **MB** is the line index; i.e., \( Y = Y(\text{MB}) \)

Description: This routine computes the discontinuities in the flow variables \( p, \rho, u, v, w, a^2 \) at the wall (\( N = 1 \)) associated with discontinuities in \( b_z \) and/or \( b_\phi \); c.f., Sec. 4.1. This routine is called only when a discontinuity is found (in BODYP) for \( Y = Y(\text{MB}) \) (i.e., the flag IJUMP(MB) = 1).

The routine computes the surface flow variables on the downstream side of the discontinuity (subscripted + in Sec. 4.1) using the formulas given in Sec. 4.1. The flow variables with subscript - in Sec. 4.1 are input from COMMON in the P, D, etc. arrays with \( N = 1 \). The output flow quantities (corresponding to the subscript + in Sec. 4.1) are stored in these locations when the routine is executed. The routine also sets the flag IJUMP(MB) (see comment cards of JUMP for details), puts ISWSMO = 0, if there is a compression corner, and starts the counts ICFL and IJMPKT(MB).

These are used in WALL and the MAIN to control
various special procedures for the wall point calculations downstream of the discontinuity (see, Sec. 4.1 for details).

10.5 TRANF, ENTRY TRANFW

Calling sequence: CALL TRANF(M,J,I)

where $M$ is the line index; i.e., $Y = Y(M)$

$J$ (= 1, 2, or 3) is the identification index corresponding to the line $M$ for $YZ, YPHI, TG6$

$I$ (= 1 or 2) is the identification index corresponding to the line $M$ for $XR, XZ, XPHI, TF4, TF6, TF7$

Description: This routine defines the arrays $R, XR, XZ, XPHI, TF4, TF6$, and $TF7$ on the line $M$ using the definitions given by (3.3g), (3.3i), and $r = b + \bar{x}(c - b)$. The quantities $\bar{x} = f, f_Z, f_Y$, etc. appearing in these equations must be specified in this routine by the user. Specifically, defining relations for the following FORTRAN variables must be programmed into this routine as functions of $(X,Y,Z)$:

$$SX = f, \quad SFX = f_X, \quad SFY = f_Y, \quad SFZ = f_Z$$

$$SFXX = f_{XX}, \quad SFYX = f_{YX}, \quad SFZX = f_{ZX}$$

(see Sec. 4.3 for a discussion of the requirements on the choice of the mapping function $f(X,Y,Z)$ and a non-trivial example). In the routine, any of the above variables which do not depend explicitly on $X$ can be defined outside the loop on $N$; all variables defined which depend on $X$ must be defined inside the loop on $N$ (see listing). Note that version 1 of this routine given in Appendix D has two options. One is for the case of no clustering in the radial direction; i.e., $f(X,Y,Z) = X$, hence, $SX = f = X(N)$, $SFX = 1.0$, and $SFY = SFZ = SFXX = SFYX = SFZX = 0$. The other option allows the user to select the desired mesh spacing in the radial direction by directly inputting the values of $\bar{x} = f$ to be used in the calculation (see sec. 4.3 for details). When this option is used, the necessary data is read-in and the quantities $f, f_X, f_{XX}$ (all $Y$ and $Z$ derivatives of $f$ are zero) are computed.
in **TRANFD** (see sec. 12.5). These derivatives are input to **TRANF** via the
**COMMON** arrays **SFD**, **SFXXD**, and **SFXXD**. The other quantities needed in
the evaluations of **R**, **XR**, etc. are **YZ**, **YPHI**, **TG6** (input from **COMMON** using
the index **J**) and **B**, **BZ**, **BPHI**, **C**, **CZ**, and **CPHI** (input from **COMMON** using the
index **M**). The quantities **XZ**, **XPHI**, **TF4**, **TF6**, and **TF7** are stored in **COMMON**
using the index **I**. Note, this routine is called only once per line.

For this call, **C** contains the corrected value, **CPHI** and **CZ** contain predicted
values. When the corrected values of **CZ** and **CPHI** are determined, the
quantities **XZ**, **XPHI**, **TF6**, **TF7** (the only ones depending on **CZ** and **CPHI**) are
updated in the Corrector-Predictor Loop.

The entry point **TRANFW** is used only once in the program (called
from Section 1 of **MAIN**). Its purpose is to print out on the heading page
an identification of the particular mapping function **f** used in the routine.

10.6 **TRANC**, ENTRY **TRANCW**

**Calling sequence:**

```
CALL TRANG(YY,M,J)
```

where

<table>
<thead>
<tr>
<th>YY</th>
<th>the value of <strong>Y</strong> = <strong>Y</strong>(<strong>M</strong>)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>M</strong></td>
<td>the line index</td>
</tr>
<tr>
<td><strong>J</strong></td>
<td>the identification index</td>
</tr>
</tbody>
</table>

| **J** is such that **YY** = **Y**(**M**) for **YPHI**, **YZ**, **TG4**, **TG5**, **TG6** |

**Description:** This routine defines the quantities **THETA**, **YPHI**, **YZ**, **TG4**, **TG5**,
**TG6** for **Y** = **YY**

where

```
THETA = \bar{y} = \delta, \quad YPHI = 1/(\delta \delta_Y), \quad YZ = - \delta_Z/\delta_Y
```

```
TG4 = \delta_Y, \quad TG5 = \delta_{YY}/\delta_Y, \quad TG6 = \delta_{ZY}/\delta_Y
```

These quantities are stored in **COMMON** using the **J** index.

The quantities \( \delta, \delta_Y, \delta_Z \), etc. appearing in the above definitions must
be specified in this routine by the user. Specifically, defining relations
for the following **FORTRAN** variables must be programmed into this routine
as functions of (**YY**, **Z**):

```
SG = \delta, \quad SCG = \delta_Y, \quad SGZ = \delta_Z
```

```
SGYY = \delta_{YY}, \quad SCGY = \delta_{YZ}
```
(see Sec. 4.3 for a discussion of the requirements on the choice of the mapping function \( g(Y,Z) \) and a non-trivial example). Note that version 1 of this routine given in Appendix D has two options. One is for the case of no clustering in the azimuthal direction; i.e., \( g(YY,Z) = YY \), hence, \( SG = YY \), \( SCY = 1.0 \), \( SCZ = 0 \), \( SCYY = 0 \), \( SCYZ = 0 \). The other option allows the user to select the desired mesh spacing in the \( \phi \)-direction by directly inputing the values of \( \phi(M) \) to be used in the calculation (c.f., sec. 4.3). For this option, the necessary data is read-in and the quantities \( g, g_Y, g_{YY} \) (Z derivatives of \( g \) are zero) are computed in \textsc{tranfd} (see sec. 12.6). These are input to \textsc{tranf} via the \textsc{common} arrays \( SGD, SCYD, SCYDD \) using the index \( M \).

The entry point \textsc{tranw} is used only once in the program (called from Section 1 of \textsc{main}). Its purpose is to print out on the heading page an identification of the particular mapping function \( g \) used in the routine.

10.7 \textsc{wall}.

Calling sequence: \textsc{call wall}(M,JR,JL,JSG,IF,L)

where

\( M \) is the line index; i.e., \( Y = Y(M) \)

\( JR,JL \) (= 1, 2, or 3) are line identification indices used for the \( Y \) differences (i.e., corresponding to \( M \) and \( M + 1 \), respectively, for the predictor and to \( M - 1 \) and \( M \), respectively, for the corrector)

\( JSG \) (= 1, 2, or 3) is the identification index corresponding to the line \( M \) for \( YPHI, YZ, TC5, TC6, A3, A4, A5, A7 \)

\( IF \) (= 1 or 2) is the identification index corresponding to the line \( M \) for \( XR, TF6, TF7, UNOR \)

\( L \) is 0 in the predictor and 1 in the corrector

Description: In this routine, the derivatives \( \frac{dp}{dz} = \frac{1}{p} \frac{dp}{dz}, \frac{3v_2}{2}, \frac{3s}{2} \) are computed for use in both the predictor and corrector for the wall.
points \((X = 0, N = 1)\); c.f., Sec. 3.4. This routine contains both the formulations described in Section 3.4. These formulations are denoted in the code using the following terminology:

- **MOD 0** indicates that (3.16a) and (3.17a) are used in (3.16) and (3.17), respectively.
- **MOD 3** indicates that (3.16b) and (3.17b) are used in (3.16) and (3.17), respectively.

When the flag ISWMOD = 0, MOD 0 is used; when the flag ISWMOD = 3, MOD 3 is used. This routine also contains the option for using second order accurate differencing for the wall points; i.e., (3.19). This option can be used with either the MOD 0 or the MOD 3 formulations. It is controlled by the MODI flag; i.e., second order accuracy is used when MODI = 1 and is not used when MODI = 0. Another option contained in this routine is that of wall entropy reduction (see, Sec. 4.2). This option can be used with any combination of the other options; it is controlled by the flag ISWSMO. When ISWSMO ≠ 0 and M ≠ ISWSMO, the routine computes the wall value of \(s\) (not its derivative) using the extrapolation formula (4.10.2).

Initially, the user can select which of the above options are to be used (see User's Manual for instructions). When discontinuities in body slope are encountered on the line \(M\), modifications in the computational procedure at the wall are automatically made on the line \(M\) and other options come into play (see Sec. 4.1 for a discussion of these procedures). The wall point calculation on the line \(M\) is controlled by the flag IJUMP1(M). When IJUMP1(M) = 0 the user selected options are used; i.e., there is no body slope discontinuity on the line \(M\). Immediately after a discontinuity is found on the line \(M\) by BODYP, IJUMP1(M) (in JUMP or BODYP) is set to: 2 if JUMP finds no pressure change across the discontinuity
(or if JUMP is not used), 3 if JUMP finds a pressure change due to an expansion, 4 if JUMP finds a pressure change due to a compression. When IJUMP1(M) = 2, the wall point calculation is as follows for the remainder of the run:

(i.) MOD 0 is used on the line M
(ii.) the entropy reduction option is turned off (ISWSMO is set to zero in JUMP) if a compression corner exists
(iii.) on the line M, second order accuracy is turned off (if originally selected)

When IJUMP1(M) = 3 or 4, (i.) - (iii.) are used with the option for zeroing the X derivative terms in (3.16) (c.f., Sec. 4.1). This option will then be used for ensuing marching steps on the line M until the test (4.10.2) is satisfied on the line M or until a maximum number of steps downstream from the discontinuity have been taken. The maximum number of steps used in this procedure can be chosen by the user and can differ for expansions or compressions. When either of the above criteria are satisfied, the flag IJUMP1(M) is set to 2 for the remainder of the run.

Note, the associated counting and testing is performed in this routine.

In the evaluation of \( \frac{\partial P}{\partial Z}, \frac{\partial V}{\partial Z}, \frac{\partial s}{\partial Z} \), the quantities which must be Y differenced are input from COMMON in the arrays PWY, VOWY, SWY, V2 or CG (N = 1) depending on whether the MOD 0 or the MOD 3 formulation is used. The Y differences (forward for the predictor, backward for the corrector) are controlled by the MAIN using the indices JR, JL. The final results for \( \frac{\partial P}{\partial Z}, \frac{\partial V}{\partial Z}, \) and \( \frac{\partial s}{\partial Z} \) are returned to MAIN using the COMMON variables PZ, VZ, and SZ, respectively; however, when the entropy is extrapolated the value of s, not its derivative, is returned in SZ.
10.8 SHOCK

Calling sequence: CALL SHOCK (M,JR,JL,JSG,IF,L)

where

M is the line index, i.e., \( Y = Y(M) \)

JR, JL (=1,2, or 3) are line identification indices used for
for \( Y \) differences (i.e., corresponding to \( M \) and \( M-1 \), respectively, for the predictor and to \( M-1 \) and \( M \), respectively, for the corrector)

JSG (=1,2, or 3) is the identification index corresponding
to the line \( M \) for \( YPHI \), and \( YZ \)

IF (=1 or 2) is the identification index corresponding
to the line \( M \) for \( CF \), \( CG \), and \( CE \)

L is 0 in the predictor and 1 in the corrector

Description: In this routine, the derivatives \( \frac{3c}{3z} \), \( \frac{3\phi}{3z} \), and \( \frac{3z}{3z} \)
are computed for use in both the predictor and corrector steps; c.f., Sec. 3.3. The quantities to be \( Y \) differenced are input from COMMON in the arrays \( CZY \), \( CPHIY \), and \( CG \). The \( Y \) differences (forward for the predictor, backward for the corrector) are controlled by MAIN using the indices JR, JL. The final results for \( \frac{3c}{3z} \), \( \frac{3\phi}{3z} \), \( \frac{3z}{3z} \) are returned to MAIN using the COMMON variables DCZ, DCPHZ, DCZZ, respectively.
11. AUXILIARY SUBROUTINES

11.1 INTEG

Calling sequence: CALL INTEG (IFLAG)

where IFLAG is 0 for the first entry and 1 for all other entries

Description: This routine numerically integrates the surface pressure results to obtain the components of the aerodynamic force and moment and their z derivatives. The definitions of these quantities and the procedures used for their evaluation are described in Section 5. In this routine, the center for the moment is the origin; i.e., $z_c = 0$. When the routine is called for the initial value of $Z (= z_0)$, IFLAG = 0 and only the z derivatives are computed. When the routine is called for each subsequent step, $Z^k$, IFLAG = 1 and the z derivatives and the components are computed. The latter corresponding to the body truncated at $z = Z^k$. The quantities required for evaluating the integrands in (5.1) - (5.5) are input using the COMMON arrays $P(N = 1), B, BZ, BPHI, GY, COSPHI, SINPHI$. The results are stored in the COMMON arrays $FN, FY, FA, MX, MY, MZ, FNZ, FYZ, FAZ, MXZ, MYZ, MZZ$. Note that (for the symmetric problem only) the non-zero quantities are computed. Also, the numerical formulas for determining the z derivatives are slightly different from those used for the non-symmetric problem.
11.2 INTRPL

Calling sequence: CALL INTRPL (L,X,Y,N,XX,YY)

where

L is the number of pts. in the input arrays X and Y
X is the array containing abscissas of the input table to be interpolated
Y is the array containing the ordinates of the input table to be interpolated
N is the dimension of the XX and YY arrays
XX is the array containing the abscissas at which the interpolant is to be evaluated
YY is the array containing the ordinates obtained by evaluating the interpolant at the XX values

Description: This routine is called from REZONE and SHFAX for the purpose of interpolating the given points (X,Y) to find the values YY corresponding to the specified XX values. The routine given in Appendix D uses standard linear interpolation. The routine assumes that the input data in the X and XX arrays are increasing; i.e., \( X(I) < X(I + 1) \) and \( XX(I) < XX(I + 1) \).

11.3 REZONE

Calling sequence: CALL REZONE (NCNEW,MCNEW,ROLD,PHIOLD,DCUB,DARR1,DARR2,DARR3,DARR4,ND1M,MD1M)

where

NCNEW is the number of points in X direction for the run
MCNEW is the number of points in Y direction for the run
ROLD is a dummy array used to store R array from input tape
PHIOLD is a dummy array used to store the PHI array from input tape
NSWC/WOL/TR 77-28

DCUB is an array used to store the P,U,V,W arrays from input tape*

DARR1,DARR2,DARR3 are temporary storage arrays corresponding to the index M

DARR4 is temporary storage array corresponding to the index N

NDIM is the value of the dimension for the N index

NDIM is the value of the dimension for the M index

Description: This routine must be used when the coordinates r and s of the initial data on the input tape are different than the r, s coordinates corresponding to the computational mesh for the run. The routine generates initial data at the points of the computational mesh by interpolating the data obtained from the input tape. The interpolations are performed in INTRPL (see, Sec. 11.2). Instructions for using the routine are given in the User's Manual.

11.4 RGAS

Calling sequence: CALL RGAS (PX, RX, SX, NUMX)

where

PX is pressure
RX is density
SX is entropy
NUMX is a flag indicating mode of operation (see below)

Description: This routine was developed at NASA Ames to provide thermodynamic properties of 13 different gas mixtures. The variable NGAS indicates which mixture is to be used. When NUMX is 4, pressure

*Note that the use of DCUB in this routine requires that the arrays P,U,V,W are consecutive in COMMON.

131
and density are input and entropy (SX), enthalpy (HX), sound speed (AX) and temperature (TX) are found directly using table look-ups. When NUMX is 5, entropy and pressure are input and an iterative procedure is used to determine density. This value of density and the given pressure are then used to find the other variables (as in the case NUMX=4). If NTEST is non-negative, perfect gas relations are used. In this mode GX contains γ and RRX the perfect gas constant. The variables AX, HX, TX, RRX, GX, NTEST, NGAS are transmitted to and from the routine via COMMON/RGASS. For a more detailed description of RGAS see the following: Eaton, R. R. and Larson, D. E. "Improved Real Gas Routines for Sandia's NASA Ames Flowfield Program", SAND 75-0493, Feb., 1976.

11.5 HRGAS, ENTRY ARGAS

Calling sequence: CALL HRGAS (PX, RX, QX, N1)

where

PX is pressure
RX is density
QX is sound speed square
N1 is a flag indicating mode of operation

Description: This routine is a shortened version of RGAS which calculates only enthalpy (HX) and sound speed given the pressure and the density. This routine is called only in real gas calculations. If N1 = 2 only enthalpy is returned; for N1 = 1 both quantities are returned. ENTRY ARGAS is similar except that the values of pressure and density are those defined in the last HRGAS call. This subroutine must be used in conjunction with RGAS since this latter routine loads the COMMON arrays used by HRGAS.
11.6 SERCH, LOCATE

These routines are called only in RGAS and HRGAS. They have no direct use in the flow field calculation and therefore will not be discussed here.

11.7 SHFAX, SHFAXD

Calling sequence: CALL SHFAX (I,NDIM,MDIM,RN,UN,VN,WN,PN,DN,CPP, CZO,CON,CN,CNO,CPHIO)

where

- I is \((=1,2)\) a flag indicating which of the two criteria is used for axis shifting (c.f., sec. 11, User's Manual)
- NDIM is the value of the dimension for the N index
- MDIM is the value of the dimension for the M index
- RN, UN, VN, WN, PN, CPP, CZO, CON, CN, CNO, CPHIO are dummy storage locations

Description: SHFAX is called when the coordinate system is to be shifted by a parallel displacement of the z axis in the x-z plane (see Fig. 1). This procedure is used in bent nose calculations. For a discussion of this mode of calculation see the user's manual. The coordinate system is shifted by the amount ZAS (see user's manual for explanation of this and other parameters used). The x,y,z coordinates of the new (shifted) origin in the original system is \((- ZAS,0,0)\). The routine determines the flowfield variables and shock geometry for restarting the calculation on the initial plane in the shifted coordinates which corresponds to the last computational plane in the original coordinate system. This is performed using bilinear interpolation of the known flow field in the original coordinate system. The aerodynamic moments are also
referred to the new origin. The flow variables, shock geometry, and moments in the original coordinates are input from COMMON; also the corresponding quantities in the shifted coordinates are output using COMMON. The dummy storages are used internally for the interpolations.

SHFAK is called from SHFAXD (see below).

Calling sequence: CALL SHFAXD (I,NDIM,MDIM, CV,CVP,CP,CZO,CON, CN,CNO,CPHIO)

I is (=1,2) a flag indicating which of the two criteria is used for axis shifting (sec. 11, User's Manual)
NDIM is the value of the dimension for the N index
MDIM is the value of the dimension for the M index
CV,CVP,CP,CZO,CON,CN, are dummy storage locations
CNO,CPHIO

Description: This routine calls SHFAX. It sets-up the dummy storages used in SHFAX.
12. INPUT-OUTPUT SUBROUTINES

12.1 BODY, ENTRY BODYW, ENTRY BODYR

Calling sequence: CALL BODY(M)

where \( M \) is the line index; i.e., \( Y = Y(M) \)

Description: This routine inputs to the program the values of the
body shape function, \( b(\phi, z) \), and certain of its derivatives (see below)
for \( \phi = \text{PHI}(M) \) and \( z = Z \). Where, the values of \( \text{PHI}(M) \) and \( Z \) are input to
the routine from COMMON. The coordinate system in which the function
\( b(\phi, z) \) is specified is illustrated in Fig. 1. The specific COMMON
variables which are defined in this routine are:

\[
\begin{align*}
B(M) &= b, & BZ(M) &= b_z, & BPHI(M) &= b_\phi \\
BZPHI &= b_{z\phi}, & BZZ &= b_{zz}, & BPHPHI &= b_{\phi\phi}
\end{align*}
\]

This subroutine must be supplied by the user to describe the particular
body geometry to be considered in the calculation. The only programming
requirements are that the COMMON block CBODY (also CBENT if bent nose is used)
must be included in the subroutine and the above quantities must be defined
for \( \phi = \text{PHI}(M) \) and \( z = Z \). The version of this routine supplied in the
listings (Appendix D) is discussed in the User's Manual.

The entries BODYW and BODYR are used only once in the program (both
are called from Section 1 of MAIN). The entry BODYR is used to read-in
and compute the parameters used in the body shape function. The entry
BODYW is used to print-out on the heading page a message which identifies
the particular geometry being considered in the run.
12.2 FIELD

Calling sequence: CALL FIELD

Description: The purpose of this routine is to print-out the flow field data for a fixed axial station, \( Z \), where \( Z \) is input from COMMON. The execution of this routine is controlled by the user with various output options (see the User’s Manual for descriptions and instructions). This routine can be easily adapted to suit the individual needs of the user. Note that for each entry to the routine, the final (or corrected) values of the flow variables at the axial location \( Z \) are contained in the COMMON array \( P, D, U, V, W, ASQ \). The specific outputs contained in the version of FIELD given in the listings are discussed in the Users’ Manual.

12.3 OUT

Calling sequence: CALL OUT

Description: This routine is executed when the flow field calculation is completed. The purpose of the routine is to print out on-line surface pressure distributions and force and moment data. A description of these outputs is given in the User’s Manual. The data to be printed out in this routine is read from the output tape (TAPE16) generated during the run. The routine converts the force and moment data to coefficient form and computes the centers of pressure (when defined) (see, Section 5 for definitions). The values of \( A_{ref}, z_c, z_{ref} \) used in the definitions of these quantities are input in this routine (see User’s Manual for details).

12.4 RECOVR, SAVE

Calling sequence: EXTERNAL SAVE

CALL RECOVR (SAVE, FLAGS, CHECKSUM)
where

SAVE is the name of subroutine to be executed if flagged conditions occur

FLAGS is the octal value for conditions under which recovery code is to be executed. In this code, subroutine SAVE is executed if there is an arithmetic mode error, PP call or auto-recall error, or time or storage limit exceeded.

CHECKSUM has to do with taking check sums. If equal to 0 no checksum desired.

Description: RECOVR is a special recovery routine supported by CDC on their operating systems SCOPE 3.4 and KRONOS 2.1. The RECOVR subroutine allows a user program to gain control at the time that abnormal job termination procedure would otherwise occur. In this program, subroutine SAVE is called.

Calling sequence: CALL SAVE (EX, ENRUN, RAPO)

where

EX is a 17 word integer array of the exchange package. The program does not use this array.

ENRUN is a flag that determines the type of program termination. The program does not use this flag.

RAPO may be an array starting at RA + 1. The program does not use this array.

Description: The subroutine SAVE calls subroutine FIELD (see, Sec. 12.2) for the last IERRPR steps of the calculations. See the User's Manual for a description of the variable IERRPR. Also, subroutine SAVE calls subroutine OUT (see, Sec. 12.3).
12.5 **TRANFD**

**Calling sequence:** CALL TRANFD

**Description:** This routine is called when the mesh spacing in the radial direction is read in from cards (see user's manual for details). The routine is called only once in the program (from Section 1 of MAIN). **TRANFD** reads in the values of \( \bar{x} = f(X_n) \) to be used in the calculation and computes numerically the derivatives \( f_X \) and \( f_{XX} \) (c.f., sec. 4.3). The quantities \( \bar{x} = f, f_X, \) and \( f_{XX} \) are returned using the COMMON arrays SFD, SFXD, and SFXXD, respectively. Note that NSFD is the number of radial points in the computational mesh.

12.6 **TRANGD**

**Calling sequence:** CALL TRANGD

**Description:** This routine is called when the mesh spacing in the \( \phi \)-direction is read in from cards (see user's manual for details). The routine is called only once in the program (from Section 1 of MAIN). **TRANGD** reads in the values of \( \phi = \phi_0 g(Y_m) \) to be used in the calculation and computes numerically the derivatives \( g_Y \) and \( g_{YY} \) (see, sec. 4.3). The quantities \( g, g_Y, \) and \( g_{YY} \) are returned using the COMMON arrays SGD, SGYD, SGYYD respectively. Note that NSGD is the number of \( \phi \) planes in the computational mesh. When a symmetric problem is being computed **TRANGD** also computes \( g_Y \) and \( g_{YY} \) on the fringe planes \(-\Delta Y\) and \(1+\Delta Y\). These quantities are returned using the COMMON variables GYMDY and GYYMDY (for \( Y = -\Delta Y \)) and GYIPDY and GYYIPDY (for \( Y = 1+\Delta Y \)).

*In this option, the mesh clustering function \( f(X,Y,Z) \) is assumed independent of \( Y \) and \( Z \).

**In this option, the mesh clustering function \( g(Y,Z) \) is assumed independent of \( Z \).
APPENDIX D
LISTINGS

In this appendix the Fortran listings of the code are given (with the exception of \texttt{RECOVR} which is a CDC system routine). The listings given here contain the error mode update, \texttt{IDENT DUMP}, which is described in sec. 12.2 of the User's Manual.

\begin{tabular}{ll}
\hline
\textbf{Routine} & \textbf{Page No.} \\
\hline
\texttt{MAIN, D3CSS} (sec. 9) & 140 \\
\texttt{BODY} (sec. 10.1) & 153 \\
\texttt{DECODE} (sec. 10.2) & 154 \\
\texttt{EVAL} (sec. 10.3) & 159 \\
\texttt{JUMP} (sec. 10.4) & 161 \\
\texttt{TRANF} (sec. 10.5) & 165 \\
\texttt{TRANG} (sec. 10.6) & 167 \\
\texttt{WALL} (sec. 10.7) & 169 \\
\texttt{SHOCK} (sec. 10.8) & 172 \\
\texttt{INTEG} (sec. 11.1) & 174 \\
\texttt{INTRPL} (sec. 11.2) & 176 \\
\texttt{REZONE} (sec. 11.3) & 177 \\
\texttt{RGAS} (sec. 11.4) & 179 \\
\texttt{HRGAS} (sec. 11.5) & 184 \\
\texttt{SERCH} (sec. 11.6) & 185 \\
\texttt{LOCATE} (sec. 11.6) & 186 \\
\texttt{SHFAX} (sec. 11.7) & 187 \\
\texttt{SHFAXD} (sec. 11.7) & 189 \\
\texttt{BODY} (sec. 12.1) & 190 \\
\texttt{FIELD} (sec. 12.2) & 199 \\
\texttt{OUT} (sec. 12.3) & 200 \\
\texttt{SAVE} (sec. 12.4) & 204 \\
\texttt{TRANFD} (sec. 12.5) & 205 \\
\texttt{TRANOD} (sec. 12.6) & 206 \\
\texttt{DMPSQRT} & 207 \\
\hline
\end{tabular}
PROGRAM D3CSS((INPut=100,OUTPUT=TAPE3=512,TAPE1=4,TAPE17=512) D3CSS 2
1 TAPE15=TAPE18=512,TAPE=INPUT=TAPE=OUTPUT,TAPE0) D3CSS 3

THIS PROGRAM COMPUTES 3-D SUPERSONIC FLOW OVER BODIES GIVEN BY R=PHI(z) D3CSS 5
PRINCIPLE FEATURES ARE AS FOLLOWS D3CSS 7
1. WEAK CONSERVATION FORM FOR EQUATIONS ARE SOLVED. I.E. DSCS 8
   CONSERVATION EQUATIONS ARE SOLVED IN THE FOLLOWING FORM.
   \[ \text{CONSERVATION EQUATIONS} \]
   \[ \text{CONTINUITY EQUATION} \]
   \[ \text{MOVEMENT EQUATION} \]
   \[ \text{ENERGY EQUATION} \]
D3CSS 9
2. CONSERVATION DEPENDENT VARIABLES. \( \text{CONSERVATION DEPENDENT VARIABLES} \)
   ARE RELATED TO THE NON-CONSERVATION VARIABLES (PRESSURE (P), DENSITY (D))
   VELERITY COMPS (U,W,V)
   \[ \text{CONSERVATION DEPENDENT VARIABLES} \]
   \[ \text{CONSERVATION DEPENDENT VARIABLES} \]
   \[ \text{CONSERVATION DEPENDENT VARIABLES} \]
D3CSS 10
3. MACCORMACK 2ND ORDER SCHEME USED AT INTERIOR PTS.
D3CSS 11
3.1 IF INTERIOR PRESSURE IS NON-POSITIVE9 THEN THE CONSERVATION QUANTITIES
   ARE REDEFINED USING AVERAGES IN X DIRECTION
D3CSS 12
4. AT WALL, CHARACTERISTIC COMPATABILITY RELATIONS ARE USED
   IN PREDICTOR-CORRECTOR MANNER.
D3CSS 13
4.1 WALL ENTROPY RELAXATION IS AN OPTION.
D3CSS 14
5. SHOCK SHAPE (C) AND SHOCK SLOPES (CZ) AND (CHI) ARE DETERMINE IN PREDICTOR-CORRECTOR MANNER.
D3CSS 15
6. EITHER PERFECT GAS (CONSTANT GAMMA) OR REAL GAS EQUILIBRIUM THERMO. CAN BE USED
D3CSS 16
7. MESH CLUSTERING TRANSFORMATIONS IN THE RADIAL AND CIRCUMFERENTIAL DIRECTIONS ARE INCORPORATED
D3CSS 17
8. THE USER CAN SELECT EITHER OF TWO PROBLEMS -
   (1) THE SYMMETRIC PROBLEM
   (2) THE NON-SYMMETRIC PROBLEM
D3CSS 18
9.1 IN THE SYMMETRIC PROBLEM - BODY IS SYMMETRIC WITH RESPECT TO PITCH PLANE (ZERO YAW). PHIO=0 AND PHIO=180 (DEGS) ARE SYMMETRIC PLANES.
   P, U, W ARE SYMMETRIC, PHIO AND V IS ANTI-SYMM. AT SYMM. PLANES.
D3CSS 19
9.2 IN THE NON-SYMMETRIC PROBLEM - BODY HAS NO RESTRICTIONS AND YAW MAY BE NON-ZERO. ALL FLOW VARIABLES ARE PERIODIC WITH PERIOD = 360 (DEGS)
D3CSS 20
9. THIS VERSION HAS PROVISIONS FOR LOCAL EXPANSION AND COMPRESSION JUMPS AT DISCONTINUITIES OF BZ AND/OR RPHI.
D3CSS 21
10. IN Z DIRECTION
D3CSS 22
10. THIS VERSION HAS PROVISIONS FOR BENT NOSE BODY GEOMETRIES REQUIRING A SHIFT OF AXES (SEE USERS MANUAL FOR DETAILS)

COMMON NC=NC,PK(CF,INF,PHIO(1:2)Y=PI,PHI)
COMMON CU(i:+20:25),CU(i:+29:25)

140
NSWC/WOL/TR 77-28

---

```fortran
C *** READ IN INITIAL DATA AND PROGRAM CONTROLS ***
C
C KA IS THE MAXIMUM NUMBER OF STEPS TO BE TAKEN
C ZED0 IS THE LAST Z VALUE
C FACTOR IS THE CFL FACTOR
C DZPRINT IS THE Z INCREMENT USED FOR PRINTING
C KOUT(I) IS THE NUMBER OF STEPS BETWEEN PRINT OUTF PUTS WHEN
C ZPRINT(I-1) =LF. Z =LT. ZPRINT(I)
C IZONE = 0 MEANS DO NOT REZONE
C = 1 MEANS REZONE
C MCNEW IS THE NUMBER OF POINTS IN RADIAL DIRECTION
C MCNEW IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION
C IPC = 1 THEN FORWARD DIFFERENCE FOR PREDICTOR STEP
C = 0 THEN BACKWARD DIFFERENCE FOR PREDICTOR STEP
C IPC = 1 THEN BACKWARD DIFFERENCE FOR CORRECTOR STEP
C = 0 THEN FORWARD DIFFERENCE FOR CORRECTOR STEP
C KXINST IS THE K IN THE 1-K-1 SMOOTHING OF
C CONSERVATION VECTORS IF P IS NEGATIVE
C ISWMO = ISWMO MEANS EXTRAPOLATION WALL ENTROPY FOR ISWMO PLANES
C = 0 MEANS NO EXTRAPOLATION OF WALL ENTROPY
C = 3 MEANS MOD 3 FOR WALL B.C.
C MON = 1 MEANS SECOND ORDER ACCURACY AT WALL EXCEPT AFTER
C DISCONTINUITIES IN BZ AND/OR BPM
C = 0 MEANS NO SECOND ORDER ACCURACY
C ZMODION IS THE Z VALUE AT WHICH TO TURN ON SECOND ORDER ACCURACY
C ZMODIOF IS THE Z VALUE AT WHICH TO TURN OFF SECOND ORDER ACCURACY
C (PHIJD>PHIJ2D) IS THE PHI OPEN INTERVAL IN WHICH NOT TO
C USE SUBROUTINE JUMP
C (ZCFL1-ZCFL2) IS THE Z OPEN INTERVAL TO USE FACTOR/KFAC
C AS THE CFL FACTOR
C KCFL IS THE NUMBER OF STEPS AFTER AN EXPANSION JUMP TO USE
C FACTOR/KFAC AS THE CFL FACTOR
C NJMPK = THE MAXIMUM NO OF STEPS AFTER AN EXPANSION JUMP
C TO SET X DERIVATIVES TO ZERO AT WALL
C NJMK2 = THE MAXIMUM NO OF STEPS AFTER A COMPRESSION JUMP
C TO SET X DERIVATIVES TO ZERO AT WALL
C NTARGET IS THE NUMBER OF Z TARGET POINTS
C TARGETZ IS THE ARRAY OF THE Z TARGET POINTS
C IERRPR = THE LAST IERRPR STEPS WILL BE PRINTED IF AN ERROR OCCURS
C ISTART = 1 MEANS RESTART FROM TAPE15
C = 0 MEANS DO NOT RESTART FROM TAPE15
C KSTART IS THE K STEP NO. ON TAPE15 AT WHICH TO RESTART
C ELIM AND LCNT ARE ERROR AND NUMBER LIMITS ON ITERATIVE PROCEDURES
C IFACFL IS THE NO. OF STEPS BETWEEN PRINTOUTS OF CFL INFORMATION
C ISWDF = 1 THEN THE DIFFERENCING IS SWITCHED FROM STEP TO STEP
C = 0 THEN DIFFERENCING IS NOT SWITCHED
C NSGD IS THE NO. OF PHIS TO BE READ
C SJD IS THE ARRAY OF PHIS (DEGREES) READ IN SUBROUTINE TRANSF
C NSFD IS THE NO. OF SFIXS TO BE READ IN SUBROUTINE TRANSF
C SFD IS THE ARRAY OF SFIXS READ IN SUBROUTINE TRANSF
C
C PI=4.*ATAN(1) $ RADI=PI/180.
C DO $ IMM=1,5
C KOUT(1)=200 * ZPRINT(1)=1000.000.
C S CONTINUE
C KA=2000 $ FACTOR=.9 $ DZPRINT=100000.
```

---

142
C D3CSS 155
C NC IS THE NUMBER OF POINTS IN RADIAL DIRECTION
C MC IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION
C NA IS THE NUMBER OF INTERVALS IN RADIAL DIRECTION
C MA IS THE NUMBER OF INTERVALS IN TANGENTIAL DIRECTION
C TAW IS THE ANGLE OF ATTACK IN DEGREES
C ACH IS THE MACH NUMBER
C K IS THE NUMBER OF STEPS IN THE AXIAL DIRECTION
C

IF (IERRPR .GE. 0) CALL RECOVR(SAVE9790)
IERRPR=ABS(IERRPR)

IF (ISTART .EQ. 0) GO TO 9
7 READ (15) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,PHIO,K+Z
A =NGAS*NTEST,RPX
1 =FNY FY FA MX MY MZ FNZ FYFZ FAX MZ MYMZ
2 =PHI(M),C(M),CZ(M),CPHI(M),M=1,MC
3 =(R(N,M)+U(N,M)+V(N,M)+W(N,M)+P(N,M)+D(N,M)+M=1,MC)+N=1,NC
IF (EOF(15)) 10 915
8 WRITE (16) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,PHIO,K+Z
A =NGAS*NTEST,RPX
1 =FNY FY FA MX MY MZ FNZ FYFZ FAX MZ MYMZ
2 =PHI(M),C(M),CZ(M),CPHI(M),M=1,MC
3 =(R(N,M)+U(N,M)+V(N,M)+W(N,M)+P(N,M)+D(N,M)+M=1,MC)+N=1,NC
IF (K .LT. KSTART) GO TO 7
GO TO 15
9 READ (3) NC,MC,ATTACK,YAW,ACH,GAMMA,PINF,PHIO,K+Z
A =NGAS,NTEST,RPX
1 =FNY FY FA MX MY MZ FNZ FYFZ FAX MZ MYMZ
2 =PHI(M),C(M),CZ(M),CPHI(M),M=1,MC
3 =(R(N,M)+U(N,M)+V(N,M)+W(N,M)+P(N,M)+D(N,M)+M=1,MC)+N=1,NC
IF (EOF(3)) 10 915
10 WRITE (6,4000)
4000 FORMAT(*1 NO DATA ON TAPE3 ---- STOP ----*)
STOP
12 WRITE (6,4005) K,KSTART $ STOP
4005 FORMAT(1M1,* THE LAST K ON TAPE IS IS, * LESS THAN IS, 1 * ---- STOP ----*)
C
C *** PARAMETERS ***
C 15 ATTACK $ ATTACK $ ATTACK $ ATTACK $ RAD $ OZ=0.

143
IF (NSGD.GT.0) CALL TRANGO
IF (NSFD.GT.0) CALL TRANFO
SINAL=SIGN(ATTACK) * COSAL=COS(ATTACK)  
YA=YAW*PI/180. SINBET=SIN(YA) * COSBET=COS(YA)  
PO1=PINF/DINF
OINF2=OINF*DINF
NFRST=100
GZ=EUSSIN
SINF=0.
IF (IZONE GE.0) GO TO 50
CALL RGA(S(PINF,DINF,SINF+))
HINF=H
GAMMA=1.0/(1.0-PINF/DINF)
VINF=AX=ACH
GO TO 60

50 VINF=SQRT(HINF=PINF/DINF)*ACH
HINF=H/(GAMMA=1.0)*DINF
60 HINF=MX

17 CONTINUE
IF (IZONE NE.0) GO TO 17
1CALL REZONE(MCNEW,ASQ,Wurations)
1CONTINUE
444
WRITE (6,3010) D3CSS 252
WRITE (6,3020) KAZENOFACOE D3CSS 253
WRITE (6,3030) ELIM+LCNT D3CSS 254
WRITE (6,3040) ZPRINT D3CSS 255
WRITE (6,3045) DZPRINT D3CSS 256
WRITE (6,3050) ACH,ATTA,YAINF D3CSS 257
WRITE (6,3051) PINF+DINF+HINF+MOT2/2,+SINF D3CSS 258
IF (NGAS .LT. 0) WRITE (6,3054) GAMMA,RRX D3CSS 259
IF (NGAS .GT. 0) WRITE (6,3056) NGAS D3CSS 260
WRITE (6,3060) PHIOD D3CSS 261
WRITE (6,3070) Z D3CSS 262
WRITE (6,3090) NA,MA D3CSS 264
IF (IZONE .NE. 0) WRITE (6,3091) D3CSS 265
CALL BOOMW(IOUM) D3CSS 266
CALL TRANGW(IOUM,IOUM) D3CSS 267
CALL TRANFO(IDUM,IOUM,IOUM) D3CSS 268
WRITE (6,4010) D3CSS 269
4010 FORMAT(11X,25X,*ADDITIONAL FEATURES*) D3CSS 270
IF (IP, .EQ. 1) WRITE (6,3105) D3CSS 271
IF (IP, .EQ. 0) WRITE (6,3109) D3CSS 272
IF (ISWHMO .NE. 0) WRITE (6,4019) ISWHMO D3CSS 273
IF (ISWHMD .EQ. 0) WRITE (6,4023) D3CSS 274
IF (ISWHMO .EQ. 3) WRITE (6,4033) D3CSS 275
IF (IZMOD1ON .LT. ZENV) WRITE (6,4035) ZMOD1ON D3CSS 276
IF (MOO! .EQ. 1) WRITE (6,4040) ZMOD1OF D3CSS 277
WRITE (6,3126) KINEG D3CSS 278
WRITE (6,3152) PHI12D,PHI12JD D3CSS 279
WRITE (6,3135) FACJMP+ZCFL1,ZCFL2 D3CSS 280
WRITE (6,3136) FACJMP+KCF D3CSS 281
WRITE (6,3146) NJMPK+NJMNKS D3CSS 282
IF (ISWHDF .NE. 0) WRITE (6,4055) D3CSS 283
150 CONTINUE D3CSS 284
150 FORMAT(11X,*TIME09A12) D3CSS 285
300 FORMAT(11X,*FLOW IS NONSYMMETRICAL #) D3CSS 286
301 FORMAT(11X,*FLOW IS PERIODIC WITH PERIOD = U) D3CSS 287
302 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'G')) D3CSS 288
303 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'H')) D3CSS 289
304 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'S')) D3CSS 290
305 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'O')) D3CSS 291
306 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'W')) D3CSS 292
307 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'R')) D3CSS 293
308 FORMAT(11X,*REAL GAS (GAS NUMBER IS 'T')) D3CSS 294
145
NSWC/WOL/TR 77-28

3040 FORMAT(1X,HEADING THE MESH IN THIS RUN) D3CSS 3040
3109 FORMAT(1X,HACKWARD DIFFERENCE FOR PREDICTOR STEP AND *, D3CSS 3109
1 *HACKWARD DIFFERENCE FOR CORRECTION STEP IN X DIRECTION*) D3CSS 3109
3199 FORMAT(1H0,10X,*HACKWARD DIFFERENCE FOR PREDICTOR STEP AND *, D3CSS 3199
1 *FORWARD DIFFERENCE FOR CORRECTION STEP IN X DIRECTION*) D3CSS 3199
3135 FORMAT(1H0,10X,*THE CFL FACTOR IS REDUCED TO*OFF,6,3*, D3CSS 3135
1 *WHEN/ IS IN THE INTERVAL (*,FRZ2,9**,FRZ2,***)*) D3CSS 3135
3136 FORMAT(1H0,10X,*USE CFL FACTOR **,OFF,6,3**, D3CSS 3136
1 *FOR* 11,* STEPS AFTER AN EXPANSION JUMP OCCURS*) D3CSS 3136
3147 FORMAT(1X,PXWALL = 1, D3CSS 3147
1 *SAVE MODIFIED FORM*/ D3CSS 3147
2 1H,13,* STEPS AFTER AN EXPANSION JUMP AND*13*, D3CSS 3147
1 *STEPS AFTER A COMPRESSION JUMP*) D3CSS 3147
3152 FORMAT(1H0,10X,*JUMP WHICH COMPUTES JUMPS CORRESPONDING *, D3CSS 3152
1 *TO DISCONTINUES IN BU AND/OR Bound EXCEPT FOR THE PHI INTERVAL (*, D3CSS 3152
2 F7,2,**,F7,2,***) D3CSS 3152
3206 FORMAT(1H0,10X,*IF PRESSURE IS NEGATIVE THEN THE CONSERVATION *, D3CSS 3206
1 *VECTORS ARE SMOOTHED BY 1**12,===* D3CSS 3206
4019 FORMAT(1H0,10X,*WALL ENTROPY EXTRAPOLATION FOR*13,= PLANES*, D3CSS 4019
1 *UNTIL A COMPRESSION JUMP AND THEN NO EXTRAPOLATION*) D3CSS 4019
4020 FORMAT(1H0,10X,*NO WALL ENTROPY EXTRAPOLATION*) D3CSS 4020
4029 FORMAT(1H0,10X,*MOD 0 FOR WALL POINTS*) D3CSS 4029
4030 FORMAT(1H0,10X,*MOD 3 FOR WALL POINTS UNTIL A JUMP OCCURS AND *, D3CSS 4030
1 *THEN MOD 0 IS USED*) D3CSS 4030
4035 FORMAT(1H0,10X,*OPTION FOR SECOND ORDER ACCURACY AT WALL POINTS *, D3CSS 4035
1 *IS TURN ON AT Z **,1PE15,6** D3CSS 4035
4040 FORMAT(1H0,10X,*SECOND ORDER ACCURACY IS USED AT WALL POINTS *, D3CSS 4040
1 *FOR LESS THAN*1PE15,6** OR UNTIL JUMP IS CALLED*) D3CSS 4040
4055 FORMAT(1H0,10X,*THE DIFFERencing (FORWARD - BACKWARD) IS * D3CSS 4055
1 *SWITCHED FROM STEP TO STEP*) D3CSS 4055
C C *** INITIALIZATIONS *** D3CSS 4339
DO 25 N=1,NC D3CSS 4339
25 X(N)=N-1./NX D3CSS 4339
DO 1 I=M,1,4S D3CSS 4341
I JUMP1(I)=IUMPT(I)=0 D3CSS 4341
I YM(N)=N+1/NC D3CSS 4344
DO 10 M=1,NC D3CSS 4345
10 T(CP(N,M))=TPNMP(DUMM,4) D3CSS 4346
CALL RGAS(TPNM,TPNM,DUMM,4) D3CSS 4347
W(N,M)=TPNM+5QRHOTZ=1.**X=U(N,M)**2-V(N,M)**2 D3CSS 4348
TCU=CUI(N,M)=TPNM*TPNM D3CSS 4349
CUI2(N,M)=TPNM*TPNM D3CSS 4350
CUI3(N,M)=TPNM*TPNM D3CSS 4351
CU11(N,M)=V(N,M)*TCU D3CSS 4352
CU12(N,M)=V(N,M)*TCU D3CSS 4353
CUI(N,M)=CUI(N,M)+CUI(N,M) D3CSS 4354
10 CUAL(N,M)=CUAL(N,M) D3CSS 4355
15 ASU(N,M)=JAX D3CSS 4356
15 CONTINUE D3CSS 4357
C IF (IDYAW = 0D0) CPHI(1)=CPHI(4C) D3CSS 4358
C C *** PRELIMINARY PREDICTOR LOOP (INITIAL STEP ONLY) *** D3CSS 4359
C IN THIS LOOP, K1 CORRESPONDS TO M=1 AND X2 TO M IN CF,CF*CE D3CSS 4360
C CF*CE CORRESPONDS TO VECTORS AND TRANSFORM QUANTITIES. D3CSS 4361
C NOTE WELL, IN THIS LOOP CUP AND CP STORES PREDICTED D3CSS 4362
C Z-DIFFERENCES NOT PREDICTED VALUES. D3CSS 4363
C

146
C  COMPUTE CG VECTOR AT Y=1/DY USING SYMMETRY CONDITIONS
CALL TRANQ(YMCP,MC1,K2)
CALL EVALSY(DUM,MAK1,9K2,K2*DUM)
GO TO 1045
C  COMPUTE CG VECTOR AT Y=1
65 CALL TRANQ(Y1,NK2,K2)
CALL EVALPR(DUM,M1,K2,K2,DUM)
GO TO 80
69 CALL TRANQ(YM,K2,K2)
PHI(M)=THETA*PHIO
COSPHI(M)=COS(PHI(M))
IF (IBN .LT. 1) GO TO 1045
ZAB(M)=1.E0A
GO TO 1047
C  COMPREHENSION CONE JUNCTURE FOR BENT NOSE
1045 SCR=(COSPHI(M)*TANBN)*2
AAH=1.*SCR
BBC=2.*(TCH-1.-SCR)
IF (PHI(M),LT,P102,OR.PHI(M).GT.O102)
ZRA(M)=(-BBC-SQRT(BBC*BBC-4.*AA9*CCC))/(2.*AAB)
GO TO 1047
1046 ZRA(M)=(-BBC-SQRT(BBC*BBC-4.*AA9*CCC))/(2.*AAB)
CONTINUE
C  INITIALIZATION OF CU(I,M),I=1,2,3
PM=P(M)
C(U,I,M)=ALOG(PM)
CALL RGAS(PM,D(M),SW(M),4)
ASQ1(M)=AX*AX
CU21M=SW(M)
CU31M=V1(M)+U(M)*PHI(M)/R(M)
CU11NC1M=C(M)*CU21NC1M=CU21NC1M
CU41NC1M=CU31NC1M=CU31NC1M
CALL TRANQ(M,K2,K2)
CALL EVALQ(M,K2)
IF (M .LT. 1) GO TO 100
80 M=M+1
DO 90 N=2,NA $ NP=N+1
90 CONTINUE
CALL WALL(MA,K1,K1,K1,0)
CUP(1,M)=PZ $ CUP(3,1,M)=V27
CONTINUE
C  DSS  315
DSS  316
DSS  317
DSS  318
DSS  320
DSS  321
DSS  322
DSS  323
DSS  324
DSS  325
DSS  326
DSS  327
DSS  328
DSS  329
DSS  330
DSS  331
DSS  332
DSS  333
DSS  334
DSS  335
DSS  336
DSS  337
DSS  338
DSS  339
DSS  340
DSS  341
DSS  342
DSS  343
DSS  344
DSS  345
DSS  346
DSS  347
DSS  348
DSS  349
DSS  350
DSS  351
DSS  352
DSS  353
DSS  354
DSS  355
DSS  356
DSS  357
DSS  358
DSS  359
DSS  360
DSS  361
DSS  362
DSS  363
DSS  364
DSS  365
DSS  366
DSS  367
DSS  368
DSS  369
DSS  370
DSS  371
DSS  372
DSS  373
DSS  374
DSS  375
DSS  376
DSS  377
DSS  378
DSS  379
DSS  380
DSS  381
DSS  382
DSS  383
DSS  384
DSS  385
DSS  386
DSS  387
DSS  388
DSS  389
DSS  390
DSS  391
DSS  392
DSS  393
DSS  394
DSS  395
DSS  396
DSS  397
DSS  398
DSS  399
DSS  400
DSS  401
DSS  402
DSS  403
DSS  404
DSS  405
DSS  406
DSS  407
DSS  408
DSS  409
DSS  410
DSS  411
DSS  412
DSS  413
DSS  414
DSS  415
DSS  416
DSS  417
DSS  418
DSS  419
DSS  420
DSS  421

CUP(2*1+M8)=52
CALL SMCK(M8*K2+K1*K1+K1+0)
CUP(1+NC,M8) = DCZ & CUP(2+NC,M8) = DCZ & CUP(3+NC,MR) = DCPHZ

100 CONTINUE
IF (ISDIF .NE. 0) IPC=1CP
IF (IDYAM .EQ. 1) GO TO 125
C *** SET SYMMETRY CONDITIONS ***
DO 110 N=2-NC
110 CAM(4-N,1+)=CAM(4,N+1)=0.
CAM(1,1+)=CAM(3,1+)=CAM(3,1,MC)=CAM(3,NC,MC)=0.

C******************************************************************************
C *** PRINT THE INITIAL INPUT DATA ***
125 IF (PRINT=1) CALL FIELD $ PRINTZ=2 $ CALL INTEG(0)
IF (ISTART .EQ. 0) GO TO 725
GO TO 750
C******************************************************************************
C *** THIS SECTION CONTAINS THE MAIN CALCULATION LOOP ***
C******************************************************************************
C 200 K*K=1
C *** COMPUTE DZ AND UPDATE CUP(N,M) AND CP(M) ***
IF (Z .LE. ZCFL1) GO TO 202
ZCFL1=ZCFL2 $ ZCFL2=.ZEND
DUM=FACTOR $ FACTOR=FACT1 $ FACT1=DUM
IDUM=KFAC $ KFAC=KFAC1 $ KFAC1=IDUM
202 IF (ICFL .LE. 0) GO TO 203 $ IF (ICFL .LE. 0) GO TO 203
ICFL=IFCFL+1 $ IF (ICFL .GT. KCFL) ICFL=0
FACTOR=FACTOR
203 DZ=FACTOR*XZFCL
IF (Z .GE. 1.E-04) GO TO 205
WRITE (6,3900) $ CALL SAVE(DUM,DUM)
3900 FORMAT(1H1,CUP=N,M) AND CP(M)***

C 3950 FORMAT(1H1, CUP=0, $ CALL SAVE(DUM,DUM)
3950 FORMAT(1H1, CUP=0, $ CALL SAVE(DUM,DUM)
3950 FORMAT(1H1, CUP=0, $ CALL SAVE(DUM,DUM)

C 205 IF (Z .LT. ZDZ) GO TO 206
WRITE (6,3672) K,DZ$CFL$NDT$N,MC$MCFL$MCFL$Z
3672 FORMAT(1H0, $ CALL FIELD $ PRINTZ=2 $ CALL INTEG(0)
IF (OZTANH(NETABM=ALS$LT$Z(FTOAT(MC)*T/M)+PHI0)=CENUF) GO TO 1 207
1 I=1 $ GO TO 213
207 IF (Z .LT. 0ST) GO TO 209
I=I+1 $ GO TO 207
213 IF (IDYAM .EQ. 0) GO TO 212
PHI(MC)=PHI0 $ C(M)=C(0) $ CZ(M)=CZ(1) $ CP(HI)(MC)=CP(HI)
DO 211 N=1,NC
R(N,MC)=R(N,1) $ U(N,MC)=U(N+1) $ V(N,MC)=V(N+1)
D(N,MC)=D(N,1)
211 CONTINUE
DO 212 M=2,NC
CALL SHFAD (I,NC,MC,CUP,Y,GY,PZCOR,R2,Z,BPHI)
ATTACK=ATTACK
ATTACK=ATTACK
ATTACK=ATTACK
ATTACK=ATTACK
GO TO 19
209 CONTINUE
C FINISH K,M CORRESPONDING TO K1,K2
C CORRECT TO K1,K2,K3
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CORRECT TO K1,K2,K3
C CORRECT TO K1,K2,K3
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
C CALL SHFAD I,K2,K3,MC,CUP,Y1,GY1,PZCOR,R2,Z1,BPHI.
NSWC/WOL/TR 77-28

MM2=R M4=M N=M
IF (M.NE. MC) GO TO 310 IF (MC25 .EQ. 0) GO TO 310

310 K=K-1 IF K=2 GO TO 310
JU=J IF J=2 GO TO 310 JJ=J
IF (MC .LT. MCP) GO TO 315
IF (M.NE. MCP) GO TO 319
IC=I+1 GO TO 317

11P=3
C *** COMPUTE CG VECTOR AT Y=1+DY USING SYMMETRY CONDITIONS ***
CALL TRANG(Y+2,J2)
CALL EVALSY(DUM,MA+1,J2,J2,DUM)
GO TO 560
C *** COMPUTE CG VECTOR AT Y=1 USING CORRECTED VALUES) ***
317 CALL EVALPY (DUM+1+J2,J2,DUM)
GO TO 560

320 CALL TRANG(Y,M+1)
PHI(M)=THETA*PHI0
B0(M)=BPHI(M)
CALL BODYP(M+1)
BZ0(M)=BZ0T(J3) $ BPHI(M) $ BPHI(M+1)
IF (MT .LT. MC) GO TO 325 $ IF (M .EQ. MC) GO TO 325
GO TO 345

325 COSPHI(M)=COS(PHI(M))
SINPHI(M)=SIN(PHI(M))
ICHECK=0
CALL OECOOE(M9CP,J39NCMAX,TMCMAX)
C *** CORRECTOR FOR SHOCK SHAPE ***
C(M)=CU(I,NC+2)XCP(I,J)+CU(I,J+1,NC+1)+2XICZ(M)
I =Y-Z(J3+1)/YPHI(J3)
IF (MT .GT. 3) GO TO 345
IF (M .NE. 3) GO TO 330
IP=3 GO TO 345
330 IF (MT .EQ. 3) GO TO 335

335 IIC3
C *** COMPUTE CG VECTOR AT Y=-DY USING SYMMETRY CONDITIONS ***
CALL TRANG(Y-2,J1)
C *** THE FOLLOWING CALL IS TO FIND R(N+2) ONLY ***
CALL TRANG(Y+2,J1)
CALL EVALSY(DUM+2+J1,J1,J2,DUM)

345 CALL TRANG(MB,J2)
CALL EVAL(1,MB+2,K2+2)
CALL 1(MB+K2+2,J2,J3)

GO 400 Nw=NA $ NPE=N+1CP
GO 400 1=1$IC
CU(I+MB)=$SUM(CU(I+MB)+CU(I+MB))$1
1 =CF[I+NP+1]+1)*N0X

536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592
CONTINUE

C *** CORRECTOR FOR WALL POINTS ***

CALL WALL(M, J3, J2, J2, K2, 1)
CCE(J1, 1, M2, 1, M1, 1, M4) = CCE(J1, 1, M2, 1, M1, 1, M4) + DZ * CE(J1, 1, M2)

IF (MJ1 LE. 15SMN) GO TO 420
CCE(J1, 1, M2) = .5 * (CCE(J1, 1, M2) + CCE(J1, 1, M2))
GO TO 425

420 CJ1, 1, MM2 = 0Z
CONTINUE

425 CONTINUE

IF (IIC, EQ. 4) CJ1, 1, MR = .5 * (CJ1, 1, M2, 1, M4, 1, M2) + CZ * V2Z)

D3CSS 590

C *** CORRECTOR FOR SHOCK POINTS ***

CALL SHOCK(M, J3, J2, J2, K2, 1)
D3CSS 591

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 425

420 D3CSS 596
CONTINUE

425 D3CSS 597

IF (IUPH, EQ. 4) D3CSS 598

CALL DECODE(M, UC, J2, NC, M2, 1, M2, 1, M2)
D3CSS 599

B9 = .5 * CORRECTOR FOR SHOCK POINTS
D3CSS 600

CALL SHOCK(M, J3, J2, J2, K2, 1)
D3CSS 601

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 435

D3CSS 602

DDBP = DDBZ = 0Z
D3CSS 603

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 604

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 440

D3CSS 605

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 606

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 435

D3CSS 607

CALL SHOCK(M, J3, J2, J2, K2, 1)
D3CSS 608

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 440

D3CSS 609

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 610

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 435

D3CSS 611

CALL DECODE(M, UC, J2, NC, M2, 1, M2, 1, M2)
D3CSS 612

B9 = .5 * CORRECTOR FOR SHOCK POINTS
D3CSS 613

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 614

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 440

D3CSS 615

DDBP = DDBZ = 0Z
D3CSS 616

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 617

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 435

D3CSS 618

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 619

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 440

D3CSS 620

DDBP = DDBZ = 0Z
D3CSS 621

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 622

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 435

D3CSS 623

CALL WALL(M, J3, J2, J2, K2, 1)
D3CSS 624

CU(1, NC, M2, 1, M2) = .5 * (CU(1, NC, M2, 1, M2) + CZ) * 0.7Z
GO TO 440

D3CSS 625

C *** UPDATE TRANF QUANTITIES USING CORRECTED CZ AND CPHI ***

DO 450 J = 1, NC
D3CSS 626

XH = XH(J, 2) + DSH * (XH(J, 2) - XH(J, 2)) + DSX * DSX
D3CSS 627

XH(J, 2) = XH(J, 2) + DSX * DSX
D3CSS 628

XH(J, 2) = XH(J, 2) + DSX * DSX
D3CSS 629

TF6(N, K2) = TF6(N, K2) + DORZ * DDPC
D3CSS 630

TF6(N, K2) = TF6(N, K2) + DORZ * DDPC
D3CSS 631

TF6(N, K2) = TF6(N, K2) + DORZ * DDPC
D3CSS 632

450 CONTINUE
D3CSS 633

C *** COMPUTE Z DIFFERENCES FOR PREDICTOR ***

DO 500 I = 1, NC
D3CSS 634

CUP(1, NC, M2, 1, M2) = - (CUP(1, NC, M2, 1, M2) + CPC(1, NC, M2, 1, M2)) * DDX
DO 500 I = 1, NC
D3CSS 635

CUP(1, NC, M2, 1, M2) = -(CUP(1, NC, M2, 1, M2) + CPC(1, NC, M2, 1, M2)) * DDX
DO 500 I = 1, NC
D3CSS 636

C *** NOTE THAT CP AND CUP STORE THESE QUANTITIES ***

IF (M = 1, EQ. 0) GO TO 600
D3CSS 637

500 CONTINUE
D3CSS 638

CUP(1, NC, M2, 1, M2) = (CUP(1, NC, M2, 1, M2) + CPC(1, NC, M2, 1, M2)) * DDX
D3CSS 639

CUP(1, NC, M2, 1, M2) = (CUP(1, NC, M2, 1, M2) + CPC(1, NC, M2, 1, M2)) * DDX
D3CSS 640

CUP(1, NC, M2, 1, M2) = (CUP(1, NC, M2, 1, M2) + CPC(1, NC, M2, 1, M2)) * DDX
D3CSS 641

500 CONTINUE
D3CSS 642

IF (MOD0MN, EQ. -1) MOD1 = 1
D3CSS 643

CALL WALL(M, 2, J2, J1, J1, K1, 1, 0)
D3CSS 644

IF (MOD0MN, EQ. -1) MOD1 = 1
D3CSS 645

CUP(1, 1, MW2) = DZ * CPC(1, 1, MM2) + VZ
D3CSS 646

CUP(1, 1, MW2) = DZ
D3CSS 647

CALL SHOCK(M, 2, J2, J1, J1, K1, 1, 0)
D3CSS 648

CUP(1, 1, MW2) = DZ * CPC(1, 1, MM2) + VZ
D3CSS 649

151
600 CONTINUE
IF (ISWOIF .EQ. 0) GO TO 604
IPC=1-IPC $ IPC=IPC $ BJ=-BJ
604 CONTINUE

C****** OUTPUT - FORCES - MOMENTS ******
IF (04CM2S .EQ. 0) GO TO 610
DO 625 I=1,NC
DO 626 I=1,MC
625 CONTINUE
610 IF (IPRINT .EQ. 1) GO TO 675
IF ((Z-PRINTZ) .GT. DZPRINT) GO TO 670
IF (Z .LT. ZPRINT) GO TO 650
625 CONTINUE
GO TO 700
650 IF (MOD(K*KOUT(I)) .NE. 0) GO TO 700
670 IPRINT=1
CALL FIELD S
PRINTZUZ
C ACH IS THE MACH NUMBER
C ATTA IS THE ANGLE OF ATTACK
C CONE IS THE SHOULDER ANGLE
C K IS THE STATION NUMBER
C NC IS THE NUMBER OF POINTS IN RADIAL DIRECTION
C MC IS THE NUMBER OF PLANES IN TANGENTIAL DIRECTION
C Z IS THE LENGTH ALONG BODY AXIS
C CZ IS +TAN(SIGMA)
C CPHI IS -TAN(DELTA)
C INDEX 1 MEANS FLOW VARIABLES ON THE BODY
C INDEX NC MEANS FLOW VARIABLES ON THE SHOCK
CALL INTEG(I)
725 WRITE (16) NC,MC,ATTA,YAW,A,ACH,GAMMA,PINF,DINF,PHI0,K,Z
A =NGAS,NTEST,RRX
1 =FN+FY+FA+HX+HY+HZ+FNZ+FYZ+FAZ+HXZ+HYZ+MZZ
2 =PHI(M)+C(M)+CZ(M)+CPHI(M)+M+1+MC
3 =R(N+M),U(N+M),V(N+M),W(N+M),P(N+M),D(N+M),M+1,MC
4 =N+1,NC
750 IF (Z .LT. ZEND .AND. K .LT. KA) GO TO 200
IF (IPRINT .EQ. 0) CALL FIELD
WRITE (17) NC,MC,ATTA,YAW,ACH,GAMMA,PINF,DINF,PHI0,K,Z
A =NGAS,NTEST,RRX
1 =FN+FY+FA+HX+HY+HZ+FNZ+FYZ+FAZ+HXZ+HYZ+MZZ
2 =PHI(M)+C(M)+CZ(M)+CPHI(M)+M+1+MC
3 =R(N+M),U(N+M),V(N+M),W(N+M),P(N+M),D(N+M),M+1,MC
4 =N+1,NC
CALL OUT $ STOP $ END
V 152
SUBROUTINE RODYP(M, J3)
C BODYP COMPUTES CERTAIN BODY PARAMETERS NEEDED IN
C SUBROUTINE WALL. THIS ROUTINE ALSO TESTS FOR DISCONTINUITIES
C IN Z2 AND RPM!
C J3=1,2,3 IS A LINE INDEX FOR BODY PARAMETERS
C
COMMON NC,NCX,K,PI,INF,INF,PI,10,10D1,PI,7,12
COMMON Y2(3),YPM(3),C(25),CZ(25),C(25),R(20,25)
COMMON CU(+20,25),CUP(+20,25)

C *** END OF BLANK COMMON ***
C
COMMON /CBODY/ Z,B2,BPHIT1,BZPHI1,TANCO,DEL7 CRODY 2
COMMON Z,BPHIT1,BZPHI1,TANCO,DEL7 CRODY 2
1 +PHI(25), R(25), BZ(25), BPHI(25), CO3PHI(25), SINPHI(25)

COMMON /CBODYP/ DZ,PHI1,J,PHI2(J),BZT(3),BPHIT(3),AZZT(3)
COMMON /CBODYP/ DZ,PHI1,J,PHI2(J),BZT(3),BPHIT(3),AZZT(3)

COMMON /CBLK03/ ICFI+2(A3)+A4(A3)+A5(A3)+A7(A3) CBLK03 2
1 +1JUMP(25)+1JUMP1(25)+1JUMP2(25)
C
COMMON /C800Y/ ZBZZP,PHII,PHIBZTANCODEL7 CRODY 2
C
COMMON /CBODYP/ DZ,PHI1,J,PHI2(J),BZT(3),BPHIT(3),AZZT(3)
C
COMMON /C800Y/ ZBZZP,PHII,PHIBZTANCODEL7 CRODY 2

C BZM=BZT(J3)=BZ(M) $ BPHIT(J3)=BPHI(M)
C B2M=B2(M) $ BZPHI1M=BZPHI1(M)
B2M=B2(M) $ BZPHI1M=BZPHI1(M)

C TEST1=AMAXI(ABS(BZZ)+ABS(BZM)) $ TEST1=ABS(BZM-BZOM)*DZ*TEST1
TEST1=AMAXI(ABS(BZZ)+ABS(BZM)) $ TEST1=ABS(BZM-BZOM)*DZ*TEST1

C TEST2=AMAXI(ABS(BPH1)+ABS(BPHI1)) $ TEST2=ABS(BPH1-BPHI1)*DZ*TEST2
TEST2=AMAXI(ABS(BPH1)+ABS(BPHI1)) $ TEST2=ABS(BPH1-BPHI1)*DZ*TEST2

IF (TEST1 -LE. 1.E-6 *AND. TEST2 -LE. 1.E-6) GO TO 200

C *** IF (PHI1,J -LE. PHIM1(J) -LT. PHIM2(J) THEN NO JUMP ***
C
IF (PHI1,J -LE. PHIM1(J) -LT. PHIM2(J) THEN NO JUMP ***

IF (PHI1,J -LE. PHIM1(J) -LT. PHIM2(J) THEN NO JUMP ***

ICFL1=1 $ IJUMP1(M)=2 $ GO TO 200

IF (PHI1,J -LE. PHIM1(J) -LT. PHIM2(J) THEN NO JUMP ***

10 B2(M)=BZOM $ BPH1(M)=BPHI1(M) $ IJUMP(M)=1
B2(M)=BZOM $ BPH1(M)=BPHI1(M) $ IJUMP(M)=1

ENTRY BODYPP

200 BM=M=M $ RM=M=M $ PHIM1=M=M $ YZ(J3)=YPM1(J3)

BODYPP 25

A22=DUM3=PHI1(RM) $ BPHI1=PHI1(M) $ B2M=PHI1(RM)

A22=DUM3=PHI1(RM) $ BPHI1=PHI1(M) $ B2M=PHI1(RM)

A2(D,M)=SORT(A2) $ A3(J3)=DUM1=YPHI1(J3)

B2M=PHI1(RM) $ A4(J3)=DUM3=PHI1(RM) $ A5(J3)=BZPHI1/YPHI1(J3)

RETURN

END
SUBROUTINE DECODE(M,CV,J,NDIM,MOM)

C DECODE FINDS FLOW VARIABLES FROM THE CONSERVATION VECTOR, CV.
C THIS VERSION ASSUMES THAT CV(1,J,M)=LOG(P), CV(2,J,M)=S,
C CV(3,J,M)=2*V*(PHI/J*M)/U
C CV(1,NC,M)=CV(2,NC,M)=CZ
C CV(3,NC,M)=CPH
C THIS ROUTINE CONTAINS SELECTIVE SMOOTHING OF CONSERVATION
C VECTOR WHEN DECODED PRESSURE IS NEGATIVE.
C NOTE THAT CV IS EITHER CU OR CPR AS THE
C SEQUENCE IN THE MAIN PROGRAM INDICATES
C J=1,2,3 IS A LINE INDEX FOR BODYP PARAMETERS

COMMON/RGASS/AX,HX,TX,RXS,NGE,NTST,NGAS,NFIRST
COMMON/NC,M,PI,NGP,IDX,AM,RA,RAV
COMMON/Y1(3),YPH1(3),C(25),CPH1(25),R(20,25)
COMMON/O(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASG(20,25)
COMMON/CU(4,20,25),CP(4,20,25)

C *** END OF BLANK COMMON ***

COMMON/GAS/XY,XT,ARX,GX,TEST,NGAS,N
COMMON/NC,M,PI,NGP,IDX,AM,RA,RAV
COMMON/Y1(3),YPH1(3),C(25),CPH1(25),R(20,25)
COMMON/O(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASG(20,25)
COMMON/CU(4,20,25),CP(4,20,25)

C *** END OF BLANK COMMON ***

COMMON/GAS/XY,XT,ARX,GX,TEST,NGAS,N
COMMON/NC,M,PI,NGP,IDX,AM,RA,RAV
COMMON/Y1(3),YPH1(3),C(25),CPH1(25),R(20,25)
COMMON/O(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASG(20,25)
COMMON/CU(4,20,25),CP(4,20,25)

C *** END OF BLANK COMMON ***

DIMENSION CV(4,20,25)
U32CV(3,,m)
SWIW)xSwm=CV(2,19M)
Tl=RZCM)
S
ST2=BPHI(MP/R(M)
I
T3xl.*T2*02
IF (CCV(1,1,m) .GT. -600. .AND. CV(191smI
*LT. 700.) GO TO 2001
P(1,M)=PMEPCV(1,loMfl
CALL RGAS(P,DM,SWM,5)
D(1,M)=DM
ASQ(1,M)=AXDAX
Q50=0T2-AHMA
CDUMP=Q50*1-T1*U3
IF (CDUMP .GE. 0.) GO TO 1001
CALL DMP5QMGXADECODE=1,2,K,M+1,CDUMP)
1001 WH=CONP(CDUMP)/A2(CJ)
IF (TEST=GE.0) GO TO 18
GNT=1/(1.-OM/(DM*HX))
OM(1,M)=GNT
18 U(1,M)=WM
V(1,M)=VM(13-T2*TW)/T3
U(1,M)=UM(13-T2*TM)
D0 10 N=2,NA
KNT=0
10 CONTINUE
IF (TEST) 19, 20, 20
C.**** USE LAST VALUE OF GAMMA AS ESTIMATE FOR NEW VALUE
19 IN=0
GAM=GAM(N,M)
17 GG=GAM*GAM-1
GFF=GAM+1
20 CONTINUE
U1=CV(1,N,M); U2=CV(2,N,M); U3=U1/2
UNM=U(N,M); UVM=U(N,M)*U1
V=X+2*VNM**2
PMH=MT-UK
CRAH1=U21*U21
PMUH=PMH-CHAP
IF (KNT.LT.2) GO TO 32
IF (IN.EQ.NA) WRITE (6,3268) ZNM
32 IF (PMUH.GT.0.0) GO TO 91
IF (IN.EQ.NA) WRITE (6,3268) ZNM
3268 FORMAT(IHO..PSTAYS NEGATIVE AVERAGING..F7.3..)*95X..OZN9MI(=**F7.3)
1 315)
KNT=KNT+1 $ IF (KNT.LE.2) GO TO 36
WRITE (6,3267) Z*N*M
3267 FORMAT(1H0..F*THE CONSERVATION VECTOR IS 1-*F7.3..*ZgNvm*K
=..F7.3*35)
GO TO 91
34 IF (IN.EQ.1) GO TO 4
X1K1Neg=X1K1SP2
X1K1Neg=0. $ X1K1NP2=2.
4 IF (IN.EQ.2) GO TO 5
DO 6 1=1,4
6 CV(I,N,M)=CV(I+1,N,M)*X1K1Neg*CV(I,N+1,M)/X1K1NP2
GO TO 2
5 TCU=DM*W
VC(I+2,M)=CV(I+2,M)*X1K1Neg*CV(I+2,M)/TCU/X1K1NP2
VC(I+2,M)=CV(I+2,M)*X1K1Neg*TCU+W/X1K1NP2
VC(I+2,M)=CV(I+2,M)*X1K1Neg*VC(I+2,M)*W/X1K1NP2
VC(I+2,M)=CV(I+2,M)*X1K1Neg*VC(I+2,M)*W/X1K1NP2
2 WRITE (6,3273) X1K1Neg-Z*N*M
3273 FORMAT(1H0..THE CONSERVATION VECTOR IS 1-..F7.3..*Z*N*M
=..F7.3.*1315)
GO TO 20
91 CONTINUE
CRAPP=CHTR/CRAH1
CRAPP2=CRAPP*GAM
CDUMP=-1.-CRAPP
IF (CDUMP NE. 0.) GO TO 1002
CALL DMPSTOP(4..DECODE..2,Z*N*M,CDUMP)
1002 CRAPP=CRAH2/(1.+SORT(CDUMP))*GAM
WMM=1.-CRAPP
PMM=PMH+PMH
SUM=SUM+PMH
1 IF (TEST) 21, 22, 22
21 CALL NRGAS(PNM+DNM+DUM+DUM)z2)
ERR=2.*HX*WMM*WMM*H
WMM=WMM
2 IF (ABS(Err/HX)-ELIM..23,23,24
35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
24 IF (ICNT) 28, 16
C. Quasi-Real Gas Approach Fails, Iterate for Solution.
C. First Bracket Correct Solution
16 WNM = WNM1 = WNM
ERRx = ERR1 = ERR
GCx = GC(N, M)
WMx = ERR/GC + WNM
IF (ACC-LT.1.E+99) GO TO 80
GMT = 1/(1-DNM/(DNM*HX))
ICNT = ICNT + 1
GO TO 17
25 ICNT = ICNT + 1
80 IF (WNM GT U21) WM = U21 - 1.E-10
PNM = U2 - WNM*U1
DNM = U/WNM
CALL HRGAS(PNM, DNM, DUMMY, Z2)
ERRx = ERR2*HX*WNM*WNM - PHT
WNM = WNM
IF (ABS(ERRx/HX) - ELIM) 23, 23, 42
C. Solution Not Accurate Enough
26 IF (ICNT-LCNT) 28, 28, 30
30 WRITE (6, 2000) N
STOP
28 IF (ERRx*ERRx) 44, 44, 45
C. Solution Not Bracketed
29 DNMD = WNM1 = WNM2 = 0
WMx = ERR2*HX*WNM*WNM - PHT
IF (ABS(ERR2*HX) - ELIM) 23, 23, 42
42 IF (ICNT-LCNT) 43, 43, 30
43 IF (ERR2*ERR2) 44, 44, 45
C. Solution is Bracketed
27 IF (ICNT = ICNT + 1)
WMx = WNM1 = WNM2 + WNM1/ERR1/(ERR1-ERR)
PNM = U2 - WNM*U1
DNM = U/WNM
CALL HRGAS(PNM, DNM, DUMMY, Z2)
ERRx = ERR2*HX*WNM2*WNM2 - PHT
IF (ABS(ERRx*HX) - ELIM) 23, 23, 42
42 IF (ICNT-LCNT) 43, 43, 30
43 IF (ERR2*ERR2) 44, 44, 45
C. WMx2 and WNM Bracket Solution
44 WNM = WNM2
ERRx = ERR2
GO TO 25
C. WNM1 and WNM2 are Bracket Points
45 WNM = WNM2
ERRx = ERR2
GO TO 27
C. Convergence Achieved
23 GM(N, M) = GMT = 1/(1-DNM/(DNM*HX))
IF (ICNT.GT.1) GCN(NW)=(ERR2-ERRX)/(WNMX-WNM2)
WIN(NW)=WNM2
PIN(NW)=PNM
DN(NW)=WNM
IF(ICN1.EQ.11)CALL ARGAS(PNM,ONM+AX2,1)
GO TO 10
22 CONTINUE
AX2=WNM2/WM
WIN(NW)=WNM
10 AS(NW)=AX2
C
SHOCK SHAPE AND THEREFORE V FREE STREAM ARE KNOWN.
PRESSURE AND DENSITY ARE CALCULATED FROM SHOCK RELATIONS.
C
WIN=VIN*COH(M)+VIN*INPHI(M)
VIN=VIN*COH(M)-VIN*INPHI(M)
C3=M=C3(M)=CV(3,NC,M)
CPH1=C1(M)=CV(1,NC,M) $ CPH=CPH1/CM
DMU=C12+CPH1**2+1
CVN=WIN*CN=VIN*VIN*CPH
CVN=CVN*2/DMU2
IF(NTEST.EQ.0)GO TO 48
ICNT=0
GI=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
G=GM(NC,M)
IF(ABS(ERR1/HX)-ELIM)52.52.58
58 IF(ICYNT-CYNT)55.55.57
57 WRITE(6,2003)N
STOP
2003 FORMAT(1H1,1*LIMIT ON SHOCK ITERATIONS EXCEEDED*)
55 IF(ERR1*ERR1)56.56.56
C******POINT NOT BRACKETED
76 DCVN=(SN2-CVTN)*ERR1/(ERR1-ERR1)
71 IF(ABS(DCVN)*GT.ABS(3.46CH))DCVN=3.46CH*AR1(DCVN)/DCVN
72 CVN=CVN+DCVN
73 IF(ABS(ERR1)*GT.3.46CH)GO TO 78
74 SN2=CVN
75 GO TO 54
78 SN2=CVT2
62 IF(CNT-LCNT)29.629
61 IF(ERRN*ERR1)68.68
C******SOLUTION BRACKETED
56 CVT2=SN2*(CVTN-SN2)*ERR1/(ERR1-ERR1)
57 ICNT=ICNT+1
58 CDUMP=CVN2/CVT2
59 IF(CDUMP*GE.0.)GO TO 1005
60 CALL DMPDMT(6,DECODE5,IZ0,MCNT,CDUMP)
1005 DNM=DINF*SQRT(CDUMP)
61 IF(CYNT-LCNT)62.62
60 IF(ERRN*ERR1)69.69
C******ERR AND ERRN BRACKET THE SOLUTION
68 CVTN=CVT2
67 ERR=ERR1
66 GO TO 56
C******ERRN AND ERRN BRACKET THE SOLUTION
69 SN2=CVT2
68 ERR=ERRN
GO TO 56
C******CONVERGENCE ACHIEVED
52 GM(NCM)=1./(1.-PNM/DNM)
51 CALL ARGAS(PNM,DNM,DMU2)
50 CONTINUE
XLCM=CVN2I1.-PNM/DNM)/DMU2
1N=M(NCM)=VINX-XLCA*CPC
1N=M(NCM)=WNC=WINX-XLCA*CM
P(NCM)=PNC
1N=M(NCM)=DNM
ASG(NCM)=AX2
RETURN
END
SUBROUTINE EVAL(I,IT,JSG,JCG,JCFCE)
C EVAL COMPUTES FLUX AND SOURCE VECTORS CF,CG,CF
FROM FLOW VARIABLES, TRNS AND TRNS QUANTITIES,
C EVAL ALSO COMPUTES CFL PARAMETER K.
C IF L=1 EVALUATES FUNCTIONS AND CHECK CFL (PREDICTOR)
C IF L=2 IS A LINE INDEX FOR X2.*XPHI*,TF4*,TF6*,TF7.
C JSG=1,2,3 IS A LINE INDEX FOR X1,YPHI*,X7,T04*,T05*,T06.
C JCG,JCFCE ARE LINE INDEXES FOR CG AND (CF,CE) RESPECTIVELY
C**** END OF BLANK COMMON ****
COMMON NC,MC,K,PNF,INF,PHI0,1,DIYANPI,RA
NEWCOM 1
COMMON /REVAL/ NCFL,UCFL,MCFL,CFL,RJ CEVAL
NEWCOM 2
COMMON /CPODY/ Z.BZZ,BPHI,BZP1,TANCO,OELZ,CBODY
NEWCOM 3
COMMON /BLK01/ CZY(JCG),CPHIY(JCG),BZ(JSG),BPHI(JSG),CPHI(JSG)
NEWCOM 4
COMMON /BLK02/ THFTAOY,TG4(JSG),T05(JSG),TG6(JSG)
NEWCOM 5
COMMON /BLK04/ GANMA,G8,LDGE,G9,DX,DY,NOT2,ELIM,LCNT,ISWSMO,NA
NEWCOM 6
COMMON /SW(25),XZ(20,25),XR(20,25),XPHI(120,2)
NEWCOM 7
COMMON /BLK04/ 159
C *** CHECK OF CFL CONDITION ***
IF (PNM LE. 0.) GO TO 10
ASQNM=ASQ(N*M)
ETA=W(N*M)-ASQNM
DUM1=(ETA*(XRN*XRN+XPOR**2)+AD**2)*ASQNM
YPB=YPOR/RH
DUM2=(ETA*VNM*VNM)*YPB**2)*ASQNM
WXX1=W(N*M)-ASQNM*XRN
WXX2=(W(N*M)+ASQNM*YZJ)*BJ
IF (DUM1 -GF. 0.) GO TO 1001
CALL DMPSORT(4HEVAL1*ZoKqMNtDUMI)
1001 CONTINUE
SIG=ABS(WWX1)*SORT(DUMI)
SIG2=ABS(WWX2)*SORT(DUM2)
IF(SIG2.LT.SIG) GO TO 7
SIG3=ABS(WWX1*WWX2)
IF(SIG3.LT.SIG) GO TO 8
JCF2=SIG/ETA
IF(CFLX.LE.CFL) GO TO 10
NCFL=NCFL+1
MCFL=MCFL+1
JCF=JCF+1
CFL=CFLX
10 CONTINUE
RETURN
ENTRY EVALSY
C*** EVALSY COMPUTES CG VECTOR, WALL VALUES OF Pt V29 ***
C V/W, S AND SHOCK SLOPES CZ AND CPHI AT PLANES
C
Y=0.4Y AND Y1=+Y/ (EVALSY)
C Y=-DY AND Y1= (EVALPR)
C
YZJ=YZ(JSG) S YPHIJ=YPHI(JSG)
CZY(JCG)=CZ(M)+ PWT(JCG)*P(1,M) S YPHI(JCG)=YPHI(JCG)
V2(JCG)=V(1,M)+U(1,M)*BPHI(M)/B(M)
CPHIY(JCG)=CPHI(M)
FAC=-1.
GO TO 15
ENTRY EVALSY
FAC=2.05*TO51(T) S FAC2=(2.0-FA11)/(2.0+FA11)
IF (M .NE. 2) FAC=1.0/FAC
YZJ=YZ(JSG) S YPHIJ=YPHI(JSG)
VZ(JCG)=V(1,M)+W(1,M)*BPHI(M)/B(M)
CPHIY(JCG)=CPHI(M)
1500 NC=M-1
NC=M-1
PNM=PNM+1 S RNMP=RNMP+1 S WNNM=W(N*M)
VNM=-V(N*M)+FAC S BB=YZJ*VNM+YPHIJ*VNM/RNM
CG(JCG)=BB=0.0(N*M)+BB
CG(JCG)=YZJ*PNM+BB
CG(JCG)=BB=0.0(N*M)+BB
CG(JCG)=YPHI*PNM/RNM+BB
CONTINUE
RETURN
END
C ** THE FLOW DOES NOT CROSS EDGE FROM MINUS TO PLUS **
C ** (NO JUMPS IN P, O, S, OSQ) **

WRITE (6,3135)
3135 FORMAT(1H ** THE FLOW DOES NOT CROSS EDGE IN MARCHING DIRECTION **)
JUMP(MA)=2
OSQP=QSM
GO TO 110
$ THETRACOS(NX/2/KNMX) $ CONST=0.75 $ QSM=OSQP $2$ THETA=THETAR/RAD $ AMACH=SQRT(15/O5O/ASQW)
IF (AMACH > 1.0) GO TO 10
C *** SUBSONIC CORNER FLOW (NO JUMPS IN P, O, S, OSQ) ***
JUMP(MA)=2
OSQP=QSM $ GO TO 110
C 10 $ QSM LT. 0.0 $ GO TO 20
C *** SUPERSONIC EXPANSION CORNER ***
WRITE (6,3140)
3140 FORMAT(1H ** SUPERSONIC EXPANSION CORNER WHERE **)
WRITE (6,3150) THETRACOS(RAD,AMACH,QT,OSQP)
3150 FORMAT(1H + float(T,AMACH,QT,OSQP)
CALL RGAS(PW,OM=DUMMY,4)
CALL RGAS(PW,OM=DUMMY,4)
QSM=CONST-2.*OSQP
ASQW=1.*QSM
VPVR=SQRT(OSQP/ASQW-1.0)
C CONTINUE
C 15 CONTINUE
IF (CC-1.0) GO TO 17
C *** SUPERSONIC COMPRESSION CORNER ***
WRITE (6,3160)
3160 FORMAT(1H ** SUPERSONIC COMPRESSION CORNER WHERE **)
WRITE (6,3150) THETRACOS(RAD,AMACH,QT,OSQP)
3150 FORMAT(1H + float(T,AMACH,QT,OSQP)
C *** (PERFECT GAS ORLIGIE SHOCK RELATIONS) ***
SINTH2=1.-COSTH2
AM4=AM4SQ*$ $ AM2=AM2SQ
AM2=AM2 SQ*$ $ C3=COSTH2/AM4
REAL  GA2, ASQW, AMMSQ

C  REAL  GA, OBLIQUE SHOCK  

TEST=0.0
CALL RGAS(PW, DW, DUMMY,4)
WRITE(6,300) PW, DW, AX, HX, ASQW, AMMSQ

300 FORMAT(1H, *AE16.8)
HMX
SIND=SORT(1.,-COSTH2)
TAND=SIND/SORT(COSTH2)
SIN1=DWC+.001+SIGN(1./SORT(AMMSQ))
IF(SIND,GE,SIN1) SIN1=SIND+.001-DWC
GO TO 108

101 SIN1=SIN2-ERROR2*(SIN3-SINE)/ERROR3-ERROR2) 

103 COS1=SORT(1.,-SIN1+SIN1)
CTN1=COS1/SIN1
UT=DW*U
VT=DW*V
PT=PW+UT*U+VT*V
WRITE(6,300) PW, DW, U, V, PT
CALL RGAS(PW, DT, DWT, DUMMY, 4)
STEP=SQRT(U+V+V+V) 
IF(-STEP.GE.1.) GO TO 109
IF(DW<.1,.051) GO TO 113
DWC=2./SIN1
SIN1=SIN1-DWC
GO TO 108

113 WRITE(6,300)

3000 FORMAT(* NO OBLIQUE SHOCK SOLUTION. NORMAL SHOCK MODE USED. *)
GAMMA=GM(1., MB)
GA2=GA2+GAMMA/(GAMMA-1.)
GCC=GMCC+1./GMCC-1.
GE=GMCC+1./2.
TEST=.2.
PT=PW*(GA2*AMMSQ-1.)/GCC
DWT=DW*/AMMSQ/(AMMSQ/GCC+1./GE))
CALL RGAS(PWTO, OW1, SW(MB), 4)

ASQ = AX * AX

GO TO 9A

109 ERR1 = 2.0 * (HIX - HX) + U1 * U1 - U2 * U2

ASQ = AX * AX

WRITE(16, 1000) ICNT, SIN1, ERR1, SIN2, ERR2, SIN3, ERR3

200 FORMAT(1H + 15, 6E16.8)

IF (ICNT.GT.30H) GO TO 9A

ICNT = ICNT + 1

ERR = ERR2

SIN3 = SIN2

SIN2 = SIN1

ERR2 = ERR1

GO TO 111

106 IF (ICNT .LE. 2 * LCNT) GO TO 103

WRITE(6, 200)

200 FORMAT(1H + 15, 6E16.8)

STOP

103 IF (ERR1.GE.0.0) GO TO 104

ERR3 = ERR1

SIN3 = SIN1

GO TO 101

104 IF (ERR3.GE.0.0) GO TO 105

ERR2 = ERR1

SIN2 = SIN1

GO TO 101

105 ERR = ERR1

ERR2 = ERR1

SIN3 = SIN2

SIN2 = SIN1

GO TO 111

98 CONTINUE

D = DWT

P = DWT

IF (TEST .GE. -1.0) GO TO 100

CONST = HX * DSP * 2

P = DWT * (1.0 - COSH2)

IF (P > GT, PW) GO TO 75

P = DWT $ IJUMP(MB) = 2 $ IMPK(T(MB)) = 0

75 CALL RGAS(PW, DWT, SW(MB), 5)

ASQ = AX * AX

DSP = SQRT(2.0 * (CONST - HX))

100 D1(MB) = D1 * P(MB) = PW $ ASQ(MB) = ASQW

110 A1 = D1 * X(I) * X(I) $ A2 = D2P / (DUM * XNP)

V(1, MB) = A2 * (XNP * XMP * XIP) - A1 * DBZ

U1(MB) = A1 * X2 + A2 * (XMP * XNP)

W(1, MB) = A1 * X2 + A2 * (XNP * ETM * ETAP)

WRITE(6, 3170)

3170 FORMAT(1H, 15, 30X, * THE OUTPUT VARIABLES ARE AS FOLLOWS*)

WRITE(16, 3180) PW, DWT, U1(MB), V(1, MB), W(1, MB), SW(MB), ASQ(MB)

RETURN

END
SUBROUTINE TRANF(M,J,I)

TRANF DEFINES QUANTITIES ASSOCIATED WITH THE CLUSTERING TRANSFORMATION IN THE R DIRECTION (SEE STATEMENTS 1-9 BELOW). THE CLUSTERING TRANSFORMATION IS ASSUMED IN THE FORM

\[ \begin{align*}
S_x &= \text{SF}(x,y,z) \\
S_y &= \text{SFX}(y) \\
S_z &= \text{SFZ}(z) \\
S_{xx} &= \text{SFXX}(x) \\
S_{yx} &= \text{SFYX}(y,z) \\
S_{zz} &= \text{SFZ}(z) \\
S_{xx} &= \text{SFXX}(x) \\
S_{yx} &= \text{SFYX}(y,z) \\
S_{zz} &= \text{SFZ}(z)
\end{align*} \]

SEE USERS MANUAL FOR RESTRICTIONS AND INSTRUCTIONS J=1,2,3 IS A LINE INDEX FOR TRANF QUANTITIES I=1,2 IS A LINE INDEX FOR TRANF QUANTITIES

COMMON NC,MC,K,PINF,DIINF,PHI0,DIYAW,PI,RA

COMMON YZ(3),YPHI(3),C(25),CZ(25),PHI(25),P(20,25)

COMMON D(20,25),P(20,25),U(20,25),W(20,25),A(20,25)

COMMON C4(20,25),CUP(4,20,25)

END OF BLANK COMMON

COMMON /CTRFN/ NSFD,SDFD(20),SFX(20),SFXX(20),SFY,SFZ

COMMON /CBOY/ Z, BZ, BPHI, BPHI, TANCO, DELZ

COMMON /BLK02/ TETA, DY, T6, D6, T6, D6

COMMON /CTRANF/ NSFD, NSFD, SFXX, SFXX, SFY, SFZ

COMMON /CBOY/ Z, BZ, BPHI, BPHI, TANCO, DELZ

COMMON C4(20,25), CUP(4,20,25)

C THIS ROUTINE IS VERSION 1 OF TRANF CORRESPONDING EITHER TO NO CLUSTERING, I.E., SFx(X,Y,Z)=X OR THE USER HAS READ IN THE SFx(X) DATA POINTS

C

C$FX=1.0$ C$SFXX=0.0$ C$FY=0.0$ C$SZ=0.0$

DO 100 N=1,NC IF (NSFD .EQ. 0) GO TO 25

C THE USER READ IN THE SFx(X) DATA POINTS

C$Sx=SFx(N)$ C$SFX=SFxx(N)$ C$SFxx=SFxxD(N)$

GO TO 50

C CORRESPONDS TO NO CLUSTERING

C$Sx=x(N)$

C THIS ROUTINE IS VERSION 1 OF TRANF CORRESPONDING EITHER TO NO CLUSTERING, I.E., SFx(X,Y,Z)=X OR THE USER HAS READ IN THE SFx(X) DATA POINTS

C$SFX=1.0$ C$SFXX=0.0$ C$FY=0.0$ C$SZ=0.0$

DO 100 N=1,NC IF (NSFD .EQ. 0) GO TO 25

C THE USER READ IN THE SFx(X) DATA POINTS

C$Sx=SFx(N)$ C$SFX=SFxx(N)$ C$SFxx=SFxxD(N)$

GO TO 50

C CORRESPONDS TO NO CLUSTERING

C$Sx=x(N)$

C THIS ROUTINE IS VERSION 1 OF TRANF CORRESPONDING EITHER TO NO CLUSTERING, I.E., SFx(X,Y,Z)=X OR THE USER HAS READ IN THE SFx(X) DATA POINTS

C$SFX=1.0$ C$SFXX=0.0$ C$FY=0.0$ C$SZ=0.0$

DO 100 N=1,NC IF (NSFD .EQ. 0) GO TO 25

C THE USER READ IN THE SFx(X) DATA POINTS

C$Sx=SFx(N)$ C$SFX=SFxx(N)$ C$SFxx=SFxxD(N)$

GO TO 50

C CORRESPONDS TO NO CLUSTERING

C$Sx=x(N)$
5 TFK(N,I) = SFXX/SFX
7 TFK(N,I) = TGFJ * (SFZX * SFYX * YZJ) / SFX * BZMCZ / CMH
8 TFK(N,I) = SFYX * YPHI * J / SFX * BPMCP / CMH
100 CONTINUE
RETURN
ENTRY TRANFW
IVERSON=1
WRITE (6,3000) IVERSON
IF (NSFD = EQ. 0) WRITE (6,3010)
'F (NSFD .NE. 0) WRITE (6,3020)
3000 FORMAT (1HO20X, *PROGRAM TRANFW*, 6X, *VERSION*, I4)
3010 FORMAT (11X, *EQUAL SPACING IN RADIAL DIRECTION*)
3020 FORMAT (11X, *SFX WAS READ IN AS DATA POINTS*)
RETURN
END
SUBROUTINE TRANG(YY,M,J)

TRANG DEFINES QUANTITIES ASSOCIATED WITH THE CLUSTERING
TRANSFORMATION IN THE PHI DIRECTION (SEE STATEMENTS -
1-6 BELOW). THE CLUSTERING TRANSFORMATION IS ASSUMED
IN THE FORM

\[ \Theta = S_G(Y_Z) \] WHERE \( \Theta = \Phi/\Phi_0 \)

THE USER CAN SPECIFY THE FUNCTION \( S_G(Y_Z) \). THE FOLLOWING

\[ S_G = S_G Y; S_G Z; S_G Y Y; S_G Y Z \]

SEE USERS MANUAL FOR RESTRICTIONS AND INSTRUCTIONS

\( \text{M} \) IS THE INDEX FOR THE TANGENTIAL PLANE

\[ J = 1, 2, 3 \] IS A LINE INDEX FOR TRANG QUANTITIES

\[ \text{COMMON} \]

\[ /CNGMC/, PINF, DINF, PHI=100, PHI_0, PI, RAD \]

\[ /YZ(3), YPHI(3), C(25), CZ(25), CPHI(25), R(20, 25) \]

\[ /COMMON(20, 25), P(20, 25), U(20, 25), V(20, 25), W(20, 25), ASQ(20, 25) \]

\[ /COMMON(CU(4, 20, 25), CUP(4, 20, 25)) \]

\[ /END \]

*** END OF BLANK COMMON ***

\[ /COMMON(C/CTRANG/, NSGD, SGO(25), SGO(25), SGYY(25), SGYO(25), SGYY(25)) \]

\[ /1, YMDO, GYYMDO, GYIDPDY, GYYYPDY \]

\[ /1, MCP \]

\[ /COMMON(C/CDBODY/, Z*BZ, BPHM, BZPHI, TANCO, DELZ \]

\[ /1, PHI(25), BZ(25), BPHI(25), CPHI(25), CSPHI(25), SINPHI(25) \]

\[ /COMMON(CLK02/, THETA, Y, T(4), T(3), T(3), T(3), T(3), T(3), T(3) \]

\[ /1, X(20), X2(20, 2), XR2(20, 2), XPARHI(20, 2), XPARY(20, 2) \]

\[ /2, TF4(20, 2), TF6(20, 2), TF7(20, 2) \]

\[ /T4(20, 2), T6(20, 2), T7(20, 2) \]

\[ /END \]

THIS ROUTINE IS VERSION 1 OF TRANG CORRESPONDING EITHER
TO NO CLUSTERING, I.E., \( S_G(Y_Z) \) = \( Y_Z \)
OR THE USER HAS READ IN THE PHIS

IF (NSGD .NE. 0) GO TO 50

THE USER READ IN THE PHIS

IF (M .NE. MCP) GO TO 30

SG = GYIDPDY & SGY = GYYPDY & SGZ = SGOZ = 0.

GO TO 2

30 IF (M .NE. 0) GO TO 35

SG = YMDO & SGY = GYYMDO & SGZ = SGOZ = 0.

GO TO 2

35 SG = SGOZ(W) & SGY = SGOYD(W) & SGYU = SGYYD(W) & SGZ = SGOZ = 0.

GO TO 1

CORRESPONDS TO NO CLUSTERING

50 SG = YY & SGY = 1.0 & SGZ = SGOY & SGOZ = 0.

*** THE FOLLOWING STATEMENTS SHOULD APPEAR IN ALL VERSIONS ***

1. \( \Theta = S_G \)

2. \( \Phi / \Phi_0 \cdot S_G(Y_Z) \)

3. \( YZ(J) = S_G / S_G Y \)

4. \( TG4(J) = S_G Y \)

5. \( TG5(J) = S_G Y Y / S_G Y \)

6. \( TG6(J) = S_G Y Z / S_G Y \)

RETURN
ENTRY TRANGW
IVERSON=1
WRITE (6,3000) IVERSON
IF (NSGD .EQ. 0) WRITE (6,3010)
IF (NSGD .NE. 0) WRITE (6,3020)
3000 FORMAT(13X*PROGRAM TRANG*,6X*VERSION*,I4)
3010 FORMAT(1X*EQUAL SPACING IN TANGENTIAL DIRECTION*)
3020 FORMAT(11X*THE PHIs WERE READ IN BY THE USER*)
RETURN
END
SUBROUTINE WALL(*,JR,JL,JSG,F,L)

C WALL COMPUTES PREDICTED OR CORRECTED 2 DERIVATIVES OF
C P, V2, AND S (ENTITY) USING CHARACTERISTIC CMP. ELS.
C V2 IS VEL. COMP. TANGENT TO WALL.
C VJ IS VEL. COMP. TANGENT TO WALL.
C V2, S, V2, AND P RESP. WHERE J=JR OR JL
C JY AND JZ ARE LINE IDENT. INDEXES FOR Y DIFFS.
C JSG=1,2: LINE INDEX FOR TRANS AND BODY PARAMETERS
C IF=1,2: LINE INDEX FOR TRANS PARAMETERS
C L = 0 CORRESPONS TO PREDICTOR
C = 1 CORRESPONS TO CORRECTOR
C THIS VERSION OF WALL CONTAINS SEVERAL OPTIONS FOR WALL B.C.
C COMMON /SLKn3/ IS A KEY TO MEANS WALL ENTROPY EXTRAPOLATION
C COMMON /RLK02/ IS A KEY TO MEANS SECOND ORDER ACCURACY
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 0 MEANS WALL B.C.
C COMMON /BLKn0/ IS A KEY TO MOD 3 MEANS WALL B.C.
CALL QMPSQHT(4, Walter, 1, 3, X, 11, C, U, 1)

1001 CALL SQRT(DUMP)
ALAM=ASOW*(ASOV#APM0#UW)
AUX=ASOV*(SIGNED(1, TF) # ODY # DUM#(SWY(JR)-SWY(JL))) # DOD
VY=(V2(JR)-V2(JL))/DDY
IF (IJMPK(M) # EQ. 0) GO TO 10
IF (IJMPK(M) # EQ. 1) GO TO 15

200 IF (IJMPK(M) # NF. 3) GO TO 210
IF (IJMPK(M) # GT. NJMPK) GO TO 230

IFAC=NMMP
GO TO 215
IF (IJMPK(M) # GT. NJMPK) GO TO 230

IFACE=NMMP
GO TO 215
IF (IJMPK(M) # GT. NJMPK) GO TO 230

IF (ICFL # EQ. 0) GO TO 10

210 IF (IJMPK(M) # GT. NJMPK) GO TO 230

IFACE=NMMP
GO TO 215
IF (IJMPK(M) # GT. NJMPK) GO TO 230

215 IF (IJMPK(M) # GT. NJMPK) GO TO 10

300 IF (ISWMOD # EQ. 0) GO TO 25

DOVY=(VOWY(JR)-VOWY(JL))/DDY # PY=(PWS(JR)-PWS(JL))/DDY
DUM=(SWY(JR)-DUM#(SWY(JR)-SWY(JL)))/DDY
1 -DUM=(BZM/BM*YPOR#0VOWY)*WW
2 -ALAM=ALAM*(TF6(JR)-TF6(JL))/WW

350 DUMM=(BZM/BM*YPOR#0VOWY)*WW
S OUMZBB*WW/ASOW-YZJ

400 VY=(V2(JR)-V2(JL))/WW
GO TO 50

50 TGS(JR)=TGS(JSL)
DO 40 I=1, 4

DCGY(I)=CG(I-1,JR)-CG(I-1,JL))/DDY
40 CONTINUE

CE1=DUM#(UW#ASOW#GB)*TF7(JR, IF))/VOP
DUM=(TGS(JR)-DUM#(TGS(JR)-TGS(JR)))/WW
DUM=(TGS(JR)-DUM#(TGS(JR)-TGS(JR)))/WW
DUM=(TGS(JR)-DUM#(TGS(JR)-TGS(JR)))/WW

600 DUM=(TGS(JR)-DUM#(TGS(JR)-TGS(JR)))/WW
S OUMZBB*WW/ASOW-YZJ

C *** THE EQUATION FOR X1 IS VALID FOR PERFECT GAS ONLY ***

X1=DUM#(ASOW#GB)
IF (I=TEST.DE.0) GO TO 60

1 X1=0.25*DUM#(UW#ASOW#GB)
CALL RAS(PW, DW, DUMMY, 4)
X1=0.25*DUM#(UW#ASOW#GB)

100 CONTINUE

P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
P2Z3=ALAM#(CE1-2.0DCGY(I)+XK1#DUM#(UW#ASOW#GB))
AAXX=(2.*UNOR(2,1F)-UNOR(3,1F))*NDX
PZCPR(M)=ALM*(XR*PXX-PXX-PRM*M)*AAAX/RETA)
GO TO 90
AN PZ=M*PZCPR(M)
90 IF (M*GT. (SWSMO) GO TO 100
CALL RGAS((P(3,4))/0(3,4),SW3,4)
CALL RGAS((P(2,4))/0(2,4),SW2,4)
S1=S2=SW2-5W3 IF (SW2+LT. SW3) S1=K*(SW2*SW3) GO TO 125
100 S2=AR*DSY/WW
125 CONTINU
PZ=PZ/PW
M001=M001+15MOD=KSMMOD
IF (IJUMP(M) .EQ. 1 .AND. L .EQ. 0) PZ=0,
RETA=BET
RETURN
END
SHOCK COMPUTES PREDICTED OR CORRECTED DERIVATIVES OF C, C'L, C'1, SHOCK 2
AND C'2, JR AND JL ARE LINE INDEXES FOR Y DIFFS.
SHOCK 3
J5 = JR + JL, LINE INDEX FOR TRAN QMNTFFS
SHOCK 4
IF = 1, 2 LINE INDEX FOR CF AND CE AREAS
SHOCK 5
L = CORRESPONS TO PREDICTOR
SHOCK 6
L = CORRESPONS TO CORRECTOR
SHOCK 7
THIS VERSION HAS SECOND ORDER DIFFERENCING
SHOCK 8
COMMON /C*PHI, PINF, DINF, PH10, IDAY, PA1, P20/ NEWCOM 1
SHOCK 9
COMMON Y(J1, J2), C1(25), C2(25), CPHI(25), R(25, 25)
SHOCK 10
COMMON D1(20, 25), D2(20, 25), U(20, 25), V(20, 25)
SHOCK 11
SHOCK 12
SHOCK 13
SHOCK 14
SHOCK 15
SHOCK 16
SHOCK 17
SHOCK 18
SHOCK 19
SHOCK 20
SHOCK 21
SHOCK 22
SHOCK 23
NEWCOM 1
NEWCOM 2
NEWCOM 3
NEWCOM 4
C *** END OF ALANK COMMON ***
SHOCK 24
SHOCK 25
SHOCK 26
SHOCK 27
SHOCK 28
SHOCK 29
SHOCK 30
SHOCK 31
SHOCK 32
SHOCK 33
SHOCK 34
SHOCK 35
SHOCK 36
SHOCK 37
SHOCK 38
SHOCK 39
SHOCK 40
SHOCK 41
SHOCK 42
THE EQUATION FOR X1 IS VALID FOR PERFECTS GAS ONLY
SHOCK 43
SHOCK 44
SHOCK 45
SHOCK 46
SHOCK 47
SHOCK 48
SHOCK 49
SHOCK 50
SHOCK 51
SHOCK 52
SHOCK 53
SHOCK 54
SHOCK 55
SHOCK 56
SHOCK 57
SHOCK 58
SHOCK 59
SHOCK 60
CONTINUE
DUM1=KXI/DS \ DUM=DUM1*ALAM
RHS=(2.*ALAM+DUM*(WS2*US**2+VS**2)*OCUZ(1))*(UV-ALAM*DUM*WS2)
1 =OCUZ(2) +US*(DUM+US)*OCUZ(3) +CPHIC*DUM*VS*OCUZ(4)
VNIOMU=(DINX-C7M+DINF*(COSPHI(4)*SINPHI(M)*CPHIC))/CMU2
2 =DINF*(SINPHI(M)-COSPHI(M)*CPHIC)/CMU2
DS2=DS**2 \ XMNSQ=(CMU2*VNIOMU*2)/(AS0S*DS2)
DUM1=KXI/DS \ DUM=DUM2*(1.-XMNSQ)*KXI*(PS-1.)/DS2/(1.-XMNSQ)
DUM1=KXI*CMU2
C1=VNIOMU*A1+DUM3*A0
DUM4=(PS-1.)/CMU2
C2=V5-CPHIC*VNIOMU/DS*A1+CM*CPHIC*DUM4
RHS=RHS-(AI-DS*WS)*(V5+CPHIC*US)*YZJ/YPHIJ
IF (ISCINEG.FO.1.) GO TO 40
IF (C1>C0.01.D0.01) GO TO 40
WRITE (6,1000) C1,L,M,K
1000 FORMAT (1H0,6*IN SURROUNTE SHOCK C1 =**1PF15.6Z3X+L+M,=1315)
CALL SAVE (DUM,DUM,DUM)
40 C1=0*C1
ISCINEG=0
DCZ=CZM-CPHIM*YZJ/YPHIJ
DCPHZ=YPHIY(1)-YZJ*(CPHIY(JR)-CPHIY(JL))*DY
DCZ=(RHS-C2*(DCPHZ-CPHIC*DCZ)/CMU)/C1
RETURN
END
SUBROUTINE INTEGRATE(FLAG)

C INTEGRATE THE WALL PRESSURE TO OBTAIN THE FORCES AND MOMENTS (ABOUT ORIGIN) AND THEIR DERIVATIVES.
C THIS ROUTINE USES SIMPSON'S RULE FOR PHI INTEGRATION.
C AND TRAPEZOIDAL RULE FOR Z INTEGRATION.

COMMON NC,MC,K,PINF,FINF,PHI0,DIYAW,PI,RAD
COMMON YZ(25),YPH1(25),C2125),COPH1(25),R(25),ASQ125)
COMMON (20,25),P(20,25),U(20,25),W(20,25),ASQ(20,25)
COMMON CUX,20,25),CUP(4,20,25)

C END OF ALANK COMMON

COMMON /CINTEG/ FN,FY,FZ,FA,MX,MY,MZ,FXZ,MYZ,MZZ
COMMON /CBODY/ Z,8ZZ,8PHI,8ZPHI,8ZPA4ITANCOiDELZ
DIMENSION SUMJ(6,2),SF(6),SFI(6),SF2(6),F(6)

IF (IDYAW .EQ. 1) GO TO 300

C SIMPSON'S RULE FOR SYMMETRY CASE (PHIO=180)
SUMJ(1,1)=SUMJ(1,2)=SUMJ(2,1)=SUMJ(2,2)=SUMJ(3,1)=SUMJ(3,2)=0.

C CONTINUE

I=1
DO 200 M=1,MA
BM=RM(M) $ BPHIB=BPHI(M)/BM
DUM2.*,P(1,M)-PINF) *BM*PHIO*GY(M)
SF(1)=DUM*($COSP=BPHI*G(SIP)
SF(3)=DUM2Z(M) $ DUM1=SF(31)BM
SF(2)=DUM*COSP
IF (M .NE. 3) GO TO 125
SF1(1)=SUMJ(1,1) $ SF1(2)=SUMJ(1,2) $ SF(11)=SUMJ(3,1)
125 DO 150 I=1,3
150 SUMJ(I,1)=SUMJ(I,2)+SF(1)
K=I-1
200 CONTINUE
SF2(1)=-2.*P(1,M)-PINF) *BM*PHIO*GY(M)
SF2(3)=SF2(1)+R(M)
SF2(2)=SF2(3)+R(M)
DO 250 I=1,3
F(I)=DYO3(4,*,SUMJ(I,2)+2.*SUMJ(I,1))
IF (K .EQ. 1) GO TO 225
F(I)=F(I)+DYO3(1.5,*,SF1(1)+1.5*SF2(I))
250 CONTINUE

C SIMPSON'S RULE FOR NON-SYMMETRY CASE (PHIO=360)

DO 325 I=1,MA
SUMJ(I,1)=0, $ SUMJ(I,2)=0.
325 CONTINUE

C CONTINUE

GO TO 600

174
$SF(3) = DUM1 * COSP * SF(4) = DUM1 * SINP$
$SF(4) = DUM1 * BPHIR * COSP * SINP$
$SF(5) = DUM1 * BPHIR * COSP * SINP$
$SF(6) = DUM1 * BPHIR$

IF ($K1 \neq P$) GO TO 375
DO 150 $I = 1 \ldots A$
$SF(I) = SUMJ$(I + 1) \neq SF(I) = SF(I)$
150 CONTINUE

$DO 400 1 = 1 \ldots A$
$SUMJ$(I + 1) = SUMJ$(I + 1) + SF(I) 

K1 = K1 + 1
500 CONTINUE

$DO 525 I = 1 \ldots A$
$F(I) = DYD3$(I) + SUMJ$(I + 2) + 2 * SUMJ$(I + 1))$
IF ($K1 \neq 1$) GO TO 525
$F(I) = F(I) + DYD3$(A * SF(I) - SF(2) - 2 * SF(1))$
525 CONTINUE

$C$

600 $F(2) = F(2) + Z \cdot F(1)$
$IF (IDYAW \neq 0) GO TO 625$
$F(4) = F(4) - Z \cdot F(5) \neq F(6) = F(6)$

625 $IF (IFLAG \neq 0) GO TO 700$
$HH = (Z - ZP) / 2$
$MY = HY * HH * F(2) + MYZ$
$FA = FA + HH * F(3) \neq FAZ$
$IF (IDYAW \neq 0) GO TO 700$
$GX = MX + HH * F(4) \neq MXZ$
$FY = FY + HH * F(5) \neq FYZ$
$MZ = MZ + HH * F(6) \neq MZZ$

700 $ZP = Z$
$FY = F(1) \neq WYZ = F(2) \neq FAZ = F(3)$
$IF (IDYAW \neq 0) GO TO 800$
$WZ = F(4) \neq FYZ = F(5) \neq MZZ = F(6)$

900 RETURN
END
SUBROUTINE INTRPL(L,X,Y,N,XX,YY)

C THIS IS A LINEAR INTERPOLATION ROUTINE. THE DATA IN THE X AND
C XX ARRAYS ARE ASSUMED TO BE INCREASING AND NO CHECKING IS DONE.
C
DIMENSION X(L), Y(L), XX(N), YY(N)

M=2
DO 200 I=1,N
DO 100 J=M,L
IF (XX(I) .LE. X(J)) GO TO 150
100 CONTINUE
J=M
150 YY(I)=Y(J-1)*((Y(J)-Y(J-1))*XX(I)-X(J-1))/((X(J)-X(J-1))
200 CONTINUE
RETURN
END
SUBROUTINE REZONE(INCNEW, MCNEW, ROLOD, PHIOLD, DCUB, 
DARR1, DARR2, DARR3, DARR4, NOIIM, MDIM)

C REZONE TAKES THE INITIAL DATA AND REZONES THE MESH

C COMMON NC, MC, K, PINF, DINF, PHI0, IDYAW, PI, RAD
COMMON YZ(3), YPHI(13), C125, CZ(25), CPHI(25), R(20, 25)
COMMON D120, 25, U(20, 25), V(20, 25), W(20, 25), A50(20, 25)
COMMON CU(4, 20, 25), CUP(4, 20, 25)

C END OF BLANK COMMON

C COMMON /CBODY/ Z, B2D, BPHI, B2PHI, TNC0, DELZ
COMMON /BLK10/ THETA, TG4(3), TG5(3), TG6(3)
COMMON /X(20, 25), X(20, 25), X(20, 25), Y(25)

C DIMENSION ROLOD(NOIM, MDIM), DARR1(NOIM, MDIM), DARR2(NOIM, MDIM)
C COMMON /CBOOY/ Z, B2D, BPHI, B2PHI, TNC0, DELZ
COMMON /BLK10/ THETA, TG4(3), TG5(3), TG6(3)
COMMON /X(20, 25), X(20, 25), X(20, 25), Y(25)

C NOTE THAT THE WAY DCUB IS USED IN THIS ROUTINE 
C REQUIRES THAT DX, U, V ARE CONSECUTIVE IN COMMON

C HCOLD=NC $ MCOLD=MC $ MC=MCNEW $ NC=NCNEW
C DO 50 NC=1, MCOLD
D0 50 M=1, MCOLD
PHI0=M*PHI(M)
DO 20 N=1, HCOLD
ROLOD(N, M)=R(N, M)
20 CONTINUE
25 CONTINUE
DO 50 N=1, NC
X(N)=X(N-1)/ODY
50 CONTINUE
DO 75 M=1, MC
Y(M)=(M-1)/ODY
CALL TRANG(Y(M), M, PHI(M), THETA=M, 10
COSPHI(M)=COS(PHI(M)), SINPHI(M)=SIN(PHI(M))
CALL BODY
75 CONTINUE
CALL INTRPL(MCOLD, PHIOLD, C, CG, PHI, DARR1)
CALL INTRPL(MCOLD, PHIOLD, C, CG, PHI, DARR2)
CALL INTRPL(MCOLD, PHIOLD, C, CG, PHI, DARR3)
DO 85 M=1, MC
C(M)=DARR1(M)
CZ(M)=DARR2(M)
CPHI(M)=DARR3(M)
80 CONTINUE
DO 85 M=1, MC
CALL TRASM(M, N, 1)
85 CONTINUE
DO 100 M=1, MCOLD
DO 100 M=1, MCOLD
DARR1(M)=ROLOD(N, M)
100 CONTINUE
CALL INTRPL(MCOLD, PHIOLD, DARR1, MC, PHI, DARR2)
DO 125 M=1, MC
ROLOD(N, M)=DARR2(M)

177
125 CONTINUE
   DO 200 I=1,4
   DO 150 M=1,MCOLD
   DARR1(M) = DCUB(N,M,I)
150 CONTINUE
   CALL INTRPL(MCOLD,PHIOLD,DARR1,MC,PHI,DARR2)
   DO 175 M=1,MC
   DCUR(N,M,I) = DARR2(M)
175 CONTINUE
200 CONTINUE
225 CONTINUE
   DO 300 I=1,4
   DO 275 M=1,MC
   CALL INTRPL(MCOLD,ROLOD(1,M),DCUR(1,M,I),NC,ROL(1,M),DARR4)
   DO 275 N=1,NC
   DCUB(N,M,I) = DARR4(N)
275 CONTINUE
300 CONTINUE
RETURN
END
SUBROUTINE RGAS(PX,RX,SX,NUMX)

C.. WALKER TEMP CONVERTED TO RANKINE
COMMON/RGAS/4X,IX,TX,RRX,GX,TEST,NGAS,NFIRST
DIMENSION TH(5600),NDL(4,11),NDU(4,11),AN(4),C(7),ANR(17),RN(4)
COMMON/CSEARCH/F,THS(600),NL,NU,NOUT,NER
COMMON/SAVFO/PO,RO,BO,ECM,NDL,NDU,MIDL,MR,RTND,SOPR0,CAX,CHX
DIMENSION MIDL(4,11),SAVFO(4)
DATA TFDN/.707/
C..... THE ARRAYS NDL AND NDU MUST BE STORED IN ADJACENT LOCATIONS.
C
C
P=PX
S=SX
RX=RX
NUM=NUMX
NUMS=NUM-6
IF(INFIRST-NGAS).GT.10.9.10
10 NFIRST=NGAS
IF(INTEST).GT.19.19
19 ANR(1)=GRX
ANR(2)=GX
ANR(3)=ANR(1)/(ANR(2)-1.)
ANR(4)=ANR(1)*ANR(3)
ANR(5)=1./ANR(2)
ANR(6)=ANR(4)/ANR(1)
ANR(7)=ANR(1)/ANR(2)
ANR(8)=ANR(9)-ANR(3)*ALOG(171.6/.001)*ANR(2)
GO TO 8
C..... LOAD PERFECT GAS CONSTANTS.
7 CONTINUE
CALL LOCATE(NGAS,9)
READ(9,9999)NDL(N),N=1,89
Nd=N=Nd(89)
READ(9,9999)TH(N),N=1,NDM,WTM(N,C(N),N=1,7
READ 9999
DO 115 N=1,44
115 MIDL(N)=NDL(N)+NDU(N)/2
NDUM=NDU(44)
DO 120 N=1,NDUMX
120 THS(N)=THS(N)
F=0.
DO 21 N=1,4
NL=NDL(N)
NU=NDU(N)
NER=MIDL(N)
CALL SSEARCH
C..... LOAD REAL GAS CONSTANTS.
21 SAVFO(N)=THS(N)*NOUT
CONC=WTM(N,28.966
PO=116.
RD=.002498*CONC
RRP=17K/CONC
PPX=RRP
RT0=DD*93.635
SOPR0=SGRT(RD/PO)
E=TH(NDM-2)
E=TH(NDM-1)
N=TH(NMM)

F=2.1632*1.468*CONC

A=FM

RT=FM+1

CC=FM

PL=0.0

RST=100.0

Z=PO/10.*T

P=7.4

P=P*PO*10.**R

Z1=PO

IF0=0+1

F=0.0

DO 23 NI=1.4

23

B=4VFO(N1)

B1=BN(1)/SOPORO

B1=BN2*RT0

B1=BN3*1.8

B1=BN4*RRR

ANR(9)=PR/(Z2*ANR(3))

RR=ANR(9)

ANR(12)=BN(2)/ANR(3)

ANR(10)=1+ANR(9)/ANR(12)-ANR(9)

ANR(11)=ANR(12)/ANR(10)

ANR(13)=1./ANR(10)

ANR(14)=ANR(12)/ANR(9)

ANR(16)=BN(4)-ANR(11)*ALOG(Z1/Z2**ANR(10))

ANR(17)=ANR(1)*BN(1)*Z2/Z1

C=ANR(14)

IF(NTEST)160,8

C.CALCULATE F AND NR, INITIALIZE CONTROL INTEGERS

IF(NUMS)40+11.70

G=REAL-C(1)-C(2)*P/(C(3)+P*(C(4)+P*C(5)))

R=ABS((RST-R)/R)

RST=R

IF(PERE<0.1) R=RLAST

RL=0

CC=CC-P

R=R=CC*(1+AA*CC/(BB*BB))/BB+.005

IF(RLT=7.1) R=R=7.0

IF(2.LT.RH) R=RH

IF(RGT3.0) RL=3.0

IF(3.LT.RL) RL=RL

NUMM=0

NUM=0

N=0

NUM=5

NUM*P=NUM9-9+NUM

P=RAG(P/PO)/2.3025851

GO TO 42

R=RLOG(R/PO)/2.3025851

NUMM=5

NUM*P=NUM9-9+NUM

180
NUP=NUM
NROT=1
42 CONTINUE
IF (R) 11,12,13
11 NR=0.0
IF (NR.LT.-7) NR=-7
GO TO 15
12 NR=1.0
GO TO 15
13 NR=2
IF (NR.GT.2) NR=2
15 Dx=P=FLOAT(NR)
NR=NR+B
FR=(R-R)/((1.0*E+D*R))
IF (F) 15B/160,160
15B IF (F.GT.-1.OE-08) F=0.0
160 CONTINUE
C IF (NUMM=9+NUM) 22*162,22
162 IF (FM.LT.F) GO TO 44
161 IF (FNUMMOP) 22*162,22
C SEARCH FOR CORRECT COEFFICIENTS AND CALCULATE DESIRED PROPERTIES.
22 DO 17 NI=NI+1,NUP
NL=NDL(N1,NR)
NU=NDU(N1,NR)
NEP=MIDL(N1,NR)
CALL SERCH
Y1=TH(I,NOUT)*FX*(TH(2,NOUT)*FX*(TH(3,NOUT)*FX*(TH(4,NOUT)*FX*(TH(5,NOUT)*FX*TH(6,NOUT)*FX*TH(7,NOUT)*FX*TH(8,NOUT)*FX ))) AN(N1)=Y1*DXY*(Y2-Y1)
17 CONTINUE
C IF (NUMM=51,52,52)
52 IF (FNUMMOP) 39*108,39
10A RX=P+10.0
51 GO TO (121*122,123,124,125,126,127)
C NORMALIZE FINAL QUANTITIES.
C CALL SERCH
Y2=TH(I,NOUT)*FX*(TH(2,NOUT)*FX*(TH(3,NOUT)*FX*(TH(4,NOUT)*FX*TH(5,NOUT)*FX*TH(6,NOUT)*FX*TH(7,NOUT)*FX*TH(8,NOUT)*FX ))) AN(N1)=Y1*DXY*(Y2-Y1)
10A RX=P+10.0
51 GO TO (121*122,123,124,125,126,127)
C IF (NUMM=51,52,52)
52 IF (FNUMMOP) 39*108,39
10A RX=P+10.0
51 GO TO (121*122,123,124,125,126,127)
C ENTROPY INTEGRAL
39 IF (FF=ABS((REAL-AN(NUP))/REAL)
IF (FFF=0.O) 37*37,38
C
37 IF (NUMM=9+NUM)
38 NUMM=NUM+1
NIMX = NIMX * 1
IF(N(MX,67+20) GO TO 44
43 IF(NUMH=2) 92 R3+84
82 IF(REAL-AN(NUP)) 85 37 86
85 R1=0
S1=AN(NUP)
R2=3
IF(PL .LT. R) R=PL
R2=0
L=0
GO TO 42
86 R2=AN(NUP)
S2=AN(NUP)
R2=3
IF(PL .LT. R) R=PL
R1=0
L=1
GO TO 42
83 IF(L) 91 90 91
90 S2=AN(NUP)
R2=3
IF(S1 .NE. S2) R=(S1 .LT. R) R=RH
R1=0
GO TO 93
91 S1=AN(NUP)
R=(REAL-S1)/(S2-S1)*(R2-R1)*R1
IF(PL .LT. R) R=PL
GO TO 93
93 IF(R2=R) 104 37 105
104 NUMR=1
R1=R2
S1=S2
L=0
IF(R2 .LT. R) 210 211 211
211 R2=PL
R2=3
GO TO 42
210 R2=R2+3
R2=3
GO TO 42
105 IF(R=R1) 106 37 42
106 NUMR=1
R2=1
S1=S2
L=1
IF(R1 .LT. R1) 217 213 213
213 R1=RH
R1=RH
GO TO 42
212 R1=R1+3
R1=RH
GO TO 42
84 IF(REAL-AN(NUP)) 87 87 88
87 R1=PH
GO TO 91
88 R2=RH
GO TO 90
44 IF(F .LT. 0.000001) 27 444 44

182
C.....OUTSIDE GAS TABLES.
   444 N TIMES = N TIMES + 1
   WRITE (6, 190)
   190 FORMAT (1X, 2MP = E13.6)
   IF (N UMS) 192, 193, 193
   192 CONTINUE
   WRITE (6, 194) RX
   194 FORMAT (1X, 2MH = E13.6)
   GO TO 196
   193 CONTINUE
   WRITE (6, 195) SX
   195 FORMAT (1X, 2HM = E13.6)
   196 IF (N UMS - 10) 199, 197, 197
   197 CONTINUE
   WRITE (6, 198)
   198 FORMAT (20X, EXIT CALLED ON Tenth FAILURF)
   GO TO 25
   70 Z = Z * SQRT (-1.0)
   25 CALL EXIT
C.....REAL GAS BUT BELOW GAS TABLES.
   27 L = B
   P = PX
   R = RX
   L1 = 0
   GO TO 26
C.....PERFECT GAS
   8 L = 0
   L1 = 3
   26 CONTINUE
   IF (N UMS) 440, 69, 70
   440 QUOT = P/R
   GO TO 65 (65, 66, 67, 68, 69, 70, NUM)
   69 EX = 5 = ANR (L + R)
   EX = EXP (E X / A N R (L - 3))
   RX = P / E X ** A N R (L - 5)
   QUOT = RX
   GO TO 67
   68 S = A N R (L + 8) = A N R (L + 3) * I A LOG (P) - A N R (L + 2) * A LOG (R)
   67 T = QUOT / A N R (L + 1)
   66 H = QUOT * A N R (L + 6)
   65 LL = L + 1
   A = S Q R T ( A N R (L + 1) * QUOT)
   30 A X = A
   M X = H
   T X = T
   S = S
   R X = R
   109 CONTINUE
   RETURN
   END
SUBROUTINE HRGAS(PX,RX,OX,N1)

DIMENSION NL(4+11),NDL(4+11),MIDL(4+11),TH(5,600)

COMMON/GAS/PX,RX,DX,EF,FM,INDL,NDL,MIDL,TH,R0,SGPDR,CAX,CMX
COMMON/GASAX/AX,XX,XX,DXX,DDX,NNTEST,NGAS
COMMON/CSEXCH/F,TH(600),NL,NU,NOUT,NER

P=ALOG(PX/RN)/2.3025851
    P=ALOG(PX/RN)/2.3025851
    IF(I11)12 13
11 N=0-1 1.4
    IF(NR. LT.-7)NR=-7
    GO TO 15
    N=1 1.5
    IF(NR. LT.-7)NR=-7
    GO TO 15
13 NR=NR+1
    IF(NR. LT.-7)NR=-7
15 NX=FLOAT(NR)
    NR=NR+1
    F=(P-R-R)/(1.*R*(E+D*R))
    ENTRY HRGAS
    IF(F)15416.160
15R IF(F. LT.-1.E-08)F=0.0
    IF(F. LT.-1.E-08)F=0.0
160 IF(F. LT.-1.E-08)F=0.0
    IF(F. LT.-1.E-08)F=0.0
161 IF(F. LT.-1.E-08)F=0.0
    IF(F. LT.-1.E-08)F=0.0
    CALL SEXCH
    CALL SEXCH
    Y1=TH(1,NOUT)*F*(TH(2,NOUT)*F*TH(3,NOUT)*F*TH(4,NOUT)))
    Y1=TH(1,NOUT)*F*(TH(2,NOUT)*F*TH(3,NOUT)*F*TH(4,NOUT)))
    NL=NDL(N1, NR)
    NL=NDL(N1, NR)
    NU=NDU(N1, NR)
    NU=NDU(N1, NR)
    Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*TH(3,NOUT)+F*TH(4,NOUT)))
    Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*TH(3,NOUT)+F*TH(4,NOUT)))
    NL=NDL(N1, NR+1)
    NL=NDL(N1, NR+1)
    NU=NDU(N1, NR+1)
    NU=NDU(N1, NR+1)
    NEX=MIDL(N1, NR)
    NEX=MIDL(N1, NR)
    CALL SEXCH
    CALL SEXCH
    Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*TH(3,NOUT)+F*TH(4,NOUT)))
    Y2=TH(1,NOUT)+F*(TH(2,NOUT)+F*TH(3,NOUT)+F*TH(4,NOUT)))
    IF(N1. EQ.2) RETURN
    IF(N1. EQ.2) RETURN
    NX=NX/1.P0*SGPDR
    NX=NX/1.P0*SGPDR
    RETURN
    RETURN
27 IF(N1. EQ.1) GO TO 28
    RETURN
    RETURN
28 OX=CAX*PX/RX
    RETURN
    RETURN
44 WRITE(4,140)
44 WRITE(4,140)
140 FORMAT(1M11. OUTSIDE GAS TABLES*)
140 FORMAT(1M11. OUTSIDE GAS TABLES*)
STOP
STOP
END
END

184
SUBROUTINE SERCH
COMMON/CERCH/X, Q(600), NL, NU, NOUT, NFR
NOUT = 70000
IF (X.GE.Q(NFR)) NL = NER
DO 10 I = NL, NU
IF (X.LT.Q(I)) RETURN
NOUT = I
10 CONTINUE
RETURN
END
SUBROUTINE LOCATE(INF, NT)
C     NF IS THE NUMBER OF THE FILE
C     NT IS THE NUMBER OF THE TAPE
C
REWIND NT
IF (INF .LE. 1) GO TO 999
IF (NF .EQ. 1)
DO 40 I = 1, INF
25 READ (INF) NT, DIM
IF (EOF(NF)) 50, 25
50 CONTINUE
999 RETURN
END
SUBROUTINE SHFAX(I,NDIM,MDIM,NN,UN,VN,WN,PN,DN,CPP,CZO,CON,CN,NO*,
  S H F A X  2
1 CPHI10)

C

THIS ROUTINE SHIFTS Z AXES BY A PARALLEL DISPLACEMENT OF ZAS

C

COMMON NC,MC,K,PINF,DINF,PHI10,DIYAW,PI,PAD

COMMON YZ(25),YPH(25),CZ(25),CPH(25),R(20,25)

COMMON D(20,25),P(20,25),U(20,25),V(20,25),W(20,25),ASQ(20,25)

COMMON CU(14,20,25),CUP(2,20,25)

NEWCOMMON

C

*** END OF BLANK COMMON ***

COMMON /CHENT/ 7BH(25),ALNS,OST,AUCN,BFSTA,RSN,CENUF,DELII,DELTAN

1 +COSINH,SZAXS,PID,TANPA1,CPH10,PHI10,THFTABN

COMMON /CBODY/ Z,H2Z,PRHMPH1,ZPHI1,TANCO,DFLZ

1 +PHI(25),R(20,25),R(25),RPH1(25),COSPH1(25),SINPH1(25)

COMMON /CINTEG/ FNY,FAX,MAXZ,MYZ,FY,FAZ,MSX,MYZ,MSZ

CINTEG

1 +OYD3*MA,GY(25)

REAL MXM,YZ,MYZMPH1.PZPHI,TANCO,DFLZ

CINTEG

1 +S N(25)GM(20,25)

DIMENSION RN(NDIM,MDIM),UN(NDIM,MDIM),VN(NDIM,MDIM)

1 +WN(NDIM,MDIM),PN(NDIM,MDIM),QN(NDIM,MDIM)

DIMENSION CPP1(1),CZO1(1),CON1(1),CN1(1),NO1(1),CPH10(1)

SEOMC=1 $ MA=MC=1

GO TO (10,11)

10 ZAS=AMIN1(ZAXS,HN) $ HN=HN-ZAS

11 DELII=DELI-ZAS $ GO TO 12

12 SMFDEL=ZAS*(COSNN**2+HSN**2)/AQN

DELTA=DELTAS+SMFDEL

DO 1 M=1,MC

CPP(M)=ATAN2(C(M)*SINPHI(M),C(M)*COSPHI(M)-ZAS)

IF (CPP(M).L-T.-1.E-8)

CPP(M)=CDP(M,+2.*PI)

CNO(M)= SQRT(C(M)**2-2.*C(M)*COSPHI(M)*ZAS*ZAS)

CPP(1)=0.

CPP(MC)=PHI10

CALL INTRPL(MC,CPP,CMP,CMPH1,CON)

CALL INTRPL(MC,CPP,CNO,MCWPHI,CM)

CALL INTRPL(MC,CPP,CPO1MCWPHI,CPO1)

DO 2 M=1,MC

CA=ATAN2(C(M)*SINPHI(M),C(M)*COSPHI(M)*ZAS)

CPP2=SINCA

$ C=COSCA

SCA=CP2*CPH10(1)/CON(M)

TCA=CA*PHI10(1)/CON(M)-CP2

UCH=SCA*COSH10(M)*TCA*SINPH1(M)

SHFAX1

CZ=CN1(UCR)

SHFAX1

CPHI1(CNZM)=SCA*PSINPH1(M)*TCA*COSPH1(M)/UCR

SHFAX1

SHFAX1

S1=SFSQ*SPINHI(M)**2+1. $ SCA=COSPH1(M)*DELTA

SHFAX36

R1(M)=S1*SCA*SCA+(R*DELTAS+DELTAS)**2+SCA/S1

R(MC)=CN(M)

RDIFF=R(NC,M)-R(1,M))/NA

10 N=N+1

DO 3 N=2,NA

3 R(N,M)=R(N,M)+FLOAT(N-1,)*RDIF

2 CONTINUE

DELTA=DELTAS-SMDEL

DO 20 N=1,MC

PHI1(NC1)

DO 20 N=1,MC

SHFAX 187
NSWC/WOL/TR 77-28

```
RNC=R((N,M)
SHFAX 47
ROC=S
RT(p',C**?..4R~lC*COSPHI(M)*ZAS+ZASQ7ASI
SHFAX 48
P'L=A
KCSINHI' NOO lV)* S
SHFAX 49
IF (PHO *LT.-I.E-8) PHO=PHO*2.0PI
SHFAX 50
IF (M .E0. PC)
PHO=PHI0
SHFAX 51
SINPrlO=SIN(PHO) S
COSPHO=COS(PHO)
SHFAX 52
S1=EPSQ*S1XL)'HO**;)*
SCA=COSPHO
DE
HOD=(~TSASA(IT+)LA*cEADLA*l-C)S
SHFAX 53
CALL INTRPLP(MCsPHI,Cv,PHO.COO)
SHFAX 54
XUP= (POC-0'Offl/
(COO-)lOO)
SHFAX 55
YOP=PH-O/PHIO
SHFAX 56
XOFF=XOFF*FLOAT (NA) +1.
SHFAX 57
J=INT (XOFF)
SHFAX 58
XOFF=XOFF-FLOAT (J)
SHFAX 59
YOFF=YOFF*FLOAT(mA) .1.
SHFAX 60
L'=INT
(SHEAXI 61
YOFF=YOFF-FLOAT (L)
SHFAX 62
J=NA $ XOFF=1.
SHFAX 63
25 IF(J.GT.0) GO TO 30
SHFAX 64
J=I
SHFAX 65
30 IF(L.LT.MC) GO TO 35
SHFAX 66
L=MA S
YOFF=1.
SHFAX 67
35 IF(L.GT.0) GO 40
SHFAX 68
L~1 S
YOFF=0.
SHFAX 69
40 XY='XOFF*YOFF
PN(N,M)=P(JqL)*(.-XOFYOFF.XY),P(JIL)(XOFF-Xy),P(JL)*(OFF
SHFAX 70
1-Xv) .P(J# 1.L* 11 XY
SHFAX 71
D(N(N,t)=D(J,L)*(1.-XOFF-YOFXY).O(J.1,L)*(XOFF-XY).O(JL.1)*(YOFF
SHFAX 72
1-XY)+U(J+1.L*1)*XY
SHFAX 73
UNN=V(J.L)*(1.-XOFF-YOFF+XY)+U(J.L)*(XOFF-XY)+U(JL.1)*(YOFF
SHFAX 74
1-XY)+V(J+1.L*1)*XY
SHFAX 75
SCR=VtJNm*S
1NqPH0-L1N1NM*COSPHO
SHFAX 76
TC= UN NY
*Si 'HO
*V.
NN
*CO S pH
SHFAX 77
UN(N,N)=TCOO~SINPHI(M)-SCR*COSPHI(m)
SHFAX 78
V(N(NM)=SC4P*SINpP 4()TCR*COSPHI(m)
SHFAX 79
20 CONTINUE
DO 100 N=1,mC
C(M)=CN
SHFAX 80
DO 100 N=1,mC
P(N,M)=PN(N,M)
SHFAX 81
DO 100 N=1,mC
D(N,M)=DN(N,M)
SHFAX 82
U(N,M)=UN(N,M)
SHFAX 83
W(N,M)=WN(N,M)
SHFAX 84
100 CONTINUE
DDELTA=DELTA+SMIFDEL
MY=M-ZAS*FA $ M2=M2+ZAS*FY
RE TUNA
END
```
SUBROUTINE SHFAXD(I+NDIM+MDIM+CV+CVP+CP+C70+CON+CN+CNO+CPHI0)
DIMENSION CV(NDIM,MDIM+4),CVP(NDIM,MDIM+4)
DIMENSION C70(1),CON(1),CN(1),CNO(1),CPHI0(1),CP(1)
CALL SHFAXD(I+NDIM+MDIM,CV(1+1+1)+CV(1+1+2)+CV(1+1+3)+CV(1+1+4),
1 CVP(1+1+1)+CVP(1+1+2)+CP+C20+CON+CN+CNO+CPHI0)
RETURN
END
SUBROUTINE ROYD(M)

C***************************************************************************
C THE ROUTINE COMPUTES THE RADIUS OF THE BODY AND DERIVATIVES ALONG
C THE BODY AT A GIVEN Z PHI VALUE. THE Z VALUE IS IN COMMON AND THE
C PHI VALUE IS PASSED AS AN ARGUMENT THROUGH THE PARAMETER M.
C NOTE THAT Z IS ASSUMED TO BE INCREASING.

COMMON /CBENT/ ZB125, ALNSOSTAUONBETA, RSNCENUFDELII, DELTA
COMMON /CBODY/ ZB25, PHII(25), B(25), BZPHI(25), COSPHI(25), SINPHI(25)
DIMENSION ZCONES(1), ZCONEb(1), ACONE(1)
NAMELIST/BOOYRO/NCONEIRNDIEFL, ACONEZCONFZRNnoRRNDZFLARE
THFTAFLTHF, TAGLNWNS, NLIFW, IFS, IFL, ZW, ZS, ZL, THETAM, THETAS
DATA (ICONE=0)
DATA (ZLAST=0.)

C--------------------------------------------------------------------------
C COMPUTE THE RADIUS OF THE BASELINE BODY AT ANGLE PHI=PHI(M)

C--------------------------------------------------------------------------
3 IF (Z .GT. ZBAR) GO TO 29
   IF (Z .LE. Z8G(M)) GO TO 25
   B=QM(1)-Z/BM .8PHI=BM=0.
   RZ2=1./BM**2*(1.-(1.-Z)**2)
   BM=SQRT(RZ2-Z**2)
   GO TO 1000

C *** VERSION 3 OF TRANSITION REGION FOR HENT CONE ***

C--------------------------------------------------------------------------
25 IF (Z .EQ. ZLAST) GO TO 42
   BETA=(COSBN*(Z-1.)*BSN*BCN/AUGN
   DELTA=(SINBN*(Z-1.)*COSBN*BSN*DELII*(COSRN**2-BSN**2))/AUGN
   GO TO 41
   IF (Z .LT. ZEN) GO TO 43
   IF (Z .EQ. ZLAST) GO TO 44
   BM=BM+TANNS*Z
   COSPHIII=DTLZ-Z*COTOV2/BTMP
   GO TO 45

C--------------------------------------------------------------------------
44 IF (COSPHIII .LT. COSPHIII) GO TO 43
   RZ=(RZ+TANN)
   BM=RPHIII=BZOMI*8PHI=0.
   GO TO 1000

C--------------------------------------------------------------------------
43 S2=2*PHII(M)**2
   SCR=EPSO*52+1.
   SCA=COSPHIII*DELTA
   R=(ZM(SQRT(SCA*COSPHIII)*DELTA)**2+DELTA)**2/SCR
   ZPHIM=BZW(8*DELTA/DELTA*DELTA+RZ/DELTA*DELTA)*8PHI=0.
   GO TO 1000

C--------------------------------------------------------------------------
29 IF (Z .LT. ZLAST) GO TO 190
   BETA=1.*SCR
   GO TO 1000

C--------------------------------------------------------------------------
190 AP.
NSWC/WOL/TR 77-28

RP-1[(M) = PTM = HM*SINPH1(M)*(DELTA-RM*EPSQ*COSPH(M))/TCR
BODY 5

9ZPHI=(RPHI-TUCSPH1(M)*(FELTABSZM/8m4DELTAPL-BM*SINPHI(M)*(EPS0*BODY 5)

19Z*OPIM-FTPESP(PHMSNH()B*~PIM)/TCR
BODY 64

RPPI(ET*RHM0*OPI4/R+MCSH~)kHMSPlM~4
BODY 6

l)4*PG(M(2CSH~)*)-.RHMSNH()CSH~))
BODY 6

TCP BODY 64

8ZZ=(-8ZMO().@FPSQP4BM*S2.AMOSCP*2.@DELTAP*COSPHI(m)).BFTA*
BODY 55

IHTAPP,8ETAP.*?2-ELTAP**2-DELTAPP(DFLTAB4*COSPIIMfl-Bu.*2
BODY 66

2-EPSQPP-S2,2.)/TCR
BODY 67

GO
TO 1000
BODY 68

29 CONTINUE
BODY 69

IF(TPND.EQ.n) GO TO 30
BODY 70

IF(LT.ZMN) GO TO 30
BODY 71

ZMN=0
BODY 72

B=M=DELZ*ZMN*TANCO
BODY 73

BRA=COS(AONE (NCONE) *PI/180.)
BODY 74

ZCEN=BRAND+BRAR
BODY 75

ZBASE=ZCEN+PRND
BODY 76

GO TO 100
BODY 77

CONTINUE

IF(Z.LT.ZFLARE) GO TO 35
BODY 78

IF(IEFL.EQ.n) GO TO 33
BODY 79

IEFL=O
BODY 80

PCODELZ+TANCO*ZFLAPE
BODY 81

TNFLR=TAN THETAFL*RAD) /RC
BODY 82

TNGLR=TAN CTHETAGL*PAD) /RC
BODY 83

RC2=RCOPC
BODY 84

GZSQ=TNGLP**2
BODY 85

SFZSQ=TNFLP**?
BODY 86

33 IF (Z =EQ. 7LAST) GO TO 35
BODY 87

GLZ=1.+ (Z-ZFLARE)*TNGLR
BODY 88

G5Q=GLZ*GLZ $ 0GZ=GLZ*TNGLR
BODY 89

FLZ= 1.+ (Z-ZFLARE)*TNFLR
BODY 90

FSQ=FLZ*FLZ $FFZ=FLZ*TNFLR
BODY 91

FZ=(Z-ZCONE (1CONE)) 50.40.40
BODY 92

40 AONE=DELZ+TANCO*ZCONE (1CONE)
BODY 93

[ICONE=[ICONE+1
BODY 94

TANCO=TAN(AONE (ICONE) *PI/180.)
BODY 95

DELZ=AONE-TANCO*ZCONE (ICONE)-1)
BODY 96

IF(ZCONE (ICONE).LE.ZWW) GO TO 31
BODY 97

ZWU=ZWW
BODY 98

BWU=DELZ*ZWIJ4TANCO
BODY 99

TANW=TAN CTHETAW (1)*AD)
BODY 100

ZWW=1.E8
BODY 101

31 IF(ZCONE (ICONE).LE.ZLL ) GO TO 32
BODY 102

ZLJ=ZLL
BODY 103

32 IF(ZCONE (ICONE).LE.ZSS) GO TO 35
BODY 104

ZSU=ZSS
BODY 105

ZSS=I.EO8
BODY 106

BSUwDELZ.ZSUATANCO
BODY 107

TANS=TAN CTtiETAS (1) *AD)
BODY 108

GO TO 35
BODY 109

50 IF(Z.TZPHI)26.27
BODY 110

9RA=ACEN+SORT(RAND**2-(Z-ZCEN)**2)
GO TO 100
BODY 111

191
GO TO 200
210 IF(PIH.GT.PIN2-1.E-05) GOTO 300
\[ P \text{=} (B(u) + (Z_7 - u) \times \tan(w) / \cos(p)) \]
IF(RW.GE.RB) GO TO 300
\[ \text{IPTR} = 4 \] \[ \text{SRH} = \text{RW} \]
300 \[ \text{BM} = \text{RB} \]

C
IPTR NOW CONTAINS A POINTER INDICATING WHICH OF THE ABOVE COMPUTED DISTANCES IS MINIMAL.

C IF IPTR = 1 THEN THE MINIMAL DISTANCE IS TO THE BASELINE BODY.

C IF IPTR = 2 THEN THE MINIMAL DISTANCE IS TO THE SIDE-CUT PLANE.

C IF IPTR = 3 THEN THE MINIMAL DISTANCE IS TO THE LEE-CUT PLANE.

C IF IPTR = 4 THEN THE MINIMAL DISTANCE IS TO THE WIND-CUT PLANE.

C THE ROUTINE THEN TRANSFERS TO THE APPROPRIATE SECTION OF THE CODE FOR COMPUTATION OF DERIVATIVES.

C GO TO (1900+400,500+600)+IPTR

C SIDE CUT
C CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE DERIVATIVES.

400 IF(IFS.IS.LE.2) GO TO 460
IF(ICS.EQ.0) GO TO 430
\[ \text{RS} = \text{RS} \times \cos(PH - PHIS) \]
\[ \text{PHIFS} = \tan(HFS / RS) \]
IF(Z.GT.ZFS)4109420
\[ \text{RSF} = BSFS \]
\[ \text{PHITS} = \tan(HFS / RSF) \]
\[ \text{TANFS} = 0 \]
GO TO 425
420 \[ \text{RSF} = BSFS \times Z_2 - ZFS \] \times \tan FS
\[ \text{PHITS} = \tan(HFS / RSF) \]
425 IF(RSF.GT.RS) GO TO 430
\[ \text{IFS} = 0 \]
\[ \$ \text{ GO TO 460 } \]
430 IF(ABS(PH - PHIS),GE,PHIFS) GO TO 450
\[ \text{IF} (\text{ABS} (\text{PH} - \text{PHIS}), \geq, \text{PHITS}) \text{ GO TO } 450 \]
\[ \text{COSU}\times\cos(\text{PH}-\text{PHIS}) \]
\[ \text{TANU} = \tan(\text{PH} - \text{PHIS}) \]
\[ \text{BM} = \text{BM} \times \text{ABS}(\text{HFS} / \text{SINU}) \]
\[ \text{GO TO 800 } \]
450 \[ \text{SINU} = \sin(\text{PH} - \text{PHIS}) \]
\[ \$ \text{COSU}\times\cos(\text{PH}-\text{PHIS}) \]
\[ \text{BM} = \text{BM} \times \text{ABS}(\text{HFS} / \text{SINU}) \]
\[ \text{GO TO 700 } \]
460 \[ \text{COSU}\times\cos(\text{PH}-\text{PHIS}) \]
\[ \text{TANU} = \tan(\text{PH} - \text{PHIS}) / \cosU \]
\[ \text{BM} = \text{BM} \times \text{ABS}(\text{HFS} / \text{SINU}) \]
\[ \text{GO TO 800 } \]

C LEE CUT
C CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE DERIVATIVES.

500 IF(IFL.IL.LE.2) GO TO 570

193
IF (ICL.EQ.0) GO TO 540
ICL=0
RL=RL*COSPHI(M)
PHI=ATAN(HFL/RL)
IF(Z.LE.ZFL) GO TO 530
RLF=9FLF
PHIL=ATAN(HFL/RLF)
TANFL=0
GO TO 535
530 HLF=8FLF*(Z-ZFL)+TANFL
PHIL=ATAN(HFL/RLF)
535 IF (RLF.GT.RL) GO TO 540
IFL=0
GO TO 570
540 IF (ABS(PHI).GE.PHIFL) GO TO 570
IF (ABS(PHI).GT.PHIL) GO TO 560
COSU=COSPHI(M)
TANU=SINPHI(M)/COSU
B(M)=RB=RLF/COSU
BZ(M)=TANFL/COSU
GO TO 800
560 SINU=SINPHI(M)
COSU=COSPHI(M)
A(M)=RB=RLF/SINU
GO TO 760
570 BZ(M)=TANFL/COSPHI(M)
TANU=SINPHI(M)/COSPHI(M)
GO TO 800
C
c
WIND C
CHECK FOR FLAP AND COMPUTE QUANTITIES NEEDED TO CALCULATE THE
DERIVATIVES.
C
c
600 IF (FW*.LE.2) GO TO 710
IF (Z .EQ. ZLAST) GO TO 630
PHITW=ATAN(HFW/RW)
IF(Z.LE.ZFW) GO TO 620
RWF=BFWF
PHITW=ATAN(HFW/RWF)
TANFW=0
GO TO 625
620 RWF=BFWF*(Z-ZFW)*TANFW
PHITW=ATAN(HFW/RWF)
625 IF (RWF.GT.RA) GO TO 630
IFW=0
GO TO 710
630 IF (ABS(PHI).GE.PHIFW) GO TO 710
IF (ABS(PHI).GT.PHITW) GO TO 650
COSU=COSPHI(M)
TANU=SINPHI(M)/COSU
BZ(M)=TANFW/COSU
B(M)=RB=RWF/COSPHI(M)
GO TO 800
650 SINU=SINPHI(M)
COSU=COSPHI(M)
B(M)=RB=RLF/SINU
GO TO 229
BODY 230
BODY 231
BODY 232
BODY 233
BODY 234
BODY 235
BODY 236
BODY 237
BODY 238
BODY 239
BODY 240
BODY 241
BODY 242
BODY 243
BODY 244
BODY 245
BODY 246
BODY 247
BODY 248
BODY 249
BODY 250
BODY 251
BODY 252
BODY 253
BODY 254
BODY 255
BODY 256
BODY 257
BODY 258
BODY 259
BODY 260
BODY 261
BODY 262
BODY 263
BODY 264
BODY 265
BODY 266
BODY 267
BODY 268
BODY 269
BODY 270
BODY 271
BODY 272
BODY 273
BODY 274
BODY 275
BODY 276
BODY 277
BODY 278
BODY 279
BODY 280
BODY 281
BODY 282
BODY 283
BODY 284
BODY 285
C COMPUTE DERIVATIVES ON THE SIDE OF THE FLAP.

700 BZ(M)=BZ=M7PHI=0.
    BPHI(M)=-H*CO6/SINU
    BPMPHI=H*{(1.0/SINU)*SINU-1.)
    GO TO 1000

710 BZ(M)=TANU/COSPHI(M)
    TANU=SINPHI(M)/COSPHI(M)
    GO TO 1000

C COMPUTE THE DERIVATIVES ON CUT OR TOP OF FLAP.

800 BPHI(M)=TANU*RB
    RZ=0.
    BZPHI=RB2.*BPHI(M)*TANU
    GO TO 1000

C COMPUTE THE DERIVATIVES ON BASELINE BODY.

900 IF(Z.LE.ZRND) GO TO 905
    RZ=RA-RCEN
    BZ=M=BZ(M)=(7CEN-Z)/RZ
    BZZ=-(I..BZM.BZM)/RZ
    BZPHI=BPHPHI=BPHI(M)=0.
    RETURN

905 IF(Z.LT.ZFLARE) GO TO 910
    BZ=M=BZ(M)=RC2'(FFZCOSSQ.GGZSIJSQ)/RB
    BZZ=(RC2e(FZSQCOSSG.GZSG*SINSQ)-BZMBZ)/R8
    BZPHI=BPHPHI=(RC2*(GSO-FSO)
    BPHPHI=(RC2*(GSO-FSO)
    ZLAT=Z
    RETURN

C ***END OF ROUTINE***

C THIS PART OF THE ROUTINE READS IN DATA DEFINING THE BODY
C AND COMPUTES VARIOUS PARAMETERS.

C NCONE DETERMINES THE NUMBER OF CONIC SECTIONS
C NRND DETERMINES WHETHER THE END OF THE BODY IS ROUNDED.
C (1=0001 IF ROUNDED AND 0 OTHERWISE)
C IEFL DETERMINES WHETHER THERE IS AN ELLIPTIC FLARE ON THE
C END OF THE BODY. (1=IFL IF THERE IS AN ELLIPTIC FLARE
C AND IS 0 OTHERWISE.)
C NEXT, FOR EACH CONIC SECTION, INPUT THE ANGLE OF THE CONE AND THE
C Z VALUE WHERE IT ENDS.
C IF THE BODY IS ROUNDED, INPUT THE Z LOCATION OF THE BEGINNING OF
C THE ROUND AND THE RADIUS OF THE ROUNING.
C IF THE AFTERBODY IS FLARED, INPUT THE Z LOCATION OF THE BEGINNING OF
C THE FLARE AND THE ANGLE OF EXPANSION OF THE PHI=0 AND 180 DEGREE

C
AXIS AND THE ANGLE OF EXPANSION FOR THE PHI=0 DEGREE AXIS.

NW IS THE NUMBER OF SECTION OF THE WIND CUT
NS IS THE NUMBER OF SECTION OF THE SIDE CUT
NL IS THE NUMBER OF SECTION OF THE LEE CUT
IFW IS ONE IF THERE IS A WIND FLAP 0 OTHERWISE
IFS IS ONE IF THERE IS A SIDE FLAP 0 OTHERWISE
IFL IS ONE IF THERE IS A LEE FLAP 0 OTHERWISE

NOTE THAT ALL FLAPS BEGIN ON THE SECOND PART OF THE CORRESPONDING CUT.

NEXT, FOR EACH CUT SECTION INPUT THE Z LOCATION OF THE BEGINNING OF THAT SECTION AND THE ANGLE OF THAT SECTION

FINALLY, FOR EACH EXISTING FLAP INPUT THE HALF-WIDTH OF THE FLAP ALONG THE Z AXIS, AND THE FLAP ANGLE.

PI=PI/180.
NCONE=1. ZNND=IZFRE=1.E08
CONE=ACONE(1).
ALNS=ALNS*RAD
TANNS=BSN/SIN(THETABN)
TANNS=BSN/SIN(THETABN/2.)
DUM2=COS2=CN02*SN02/DUM)
DUM2=COS2=CN02*SN02/DUM)
DUM2=COS2=CN02*SN02
ZEN=ZEN+DUM2*(DUM2*COS2-1.)*TANNS * ZBAR=TANNS * TANG2+CNO2*SN02
OTIL2=DUM2*ZEN*COTANG
AUGN=(COS2+BSN+SN02-ZBAR*BSN)*ZBAR (BSN*SN02)**2
EPS0=EPS0=EPS0*0.
DELAPP=0.

196
CONTINUE
TANCO=SINCOSCOS
DEL=BAR-TANCO*ZBAR
ZLIN+1=1,F08
ZSN+1=1,F08
ZNW=1,F08
IF(NW.NE.0) [W1]
IF(NS.NE.0) [S1]
IF(NL.NE.0) [L1]
PHI=PHISRAD
TANFW=TAN(FAW*PAD)
TANFS=TAN(FAS*PAD)
TANFL=TAN(FAL*RAD)
ZLL=ZL(F1)
ZWW=ZW(L1)
ZSS=ZS(L1)
IF(NW.NE.0) GD TO 3020
IF(NS.NE.0) GD TO 3040
IF(NL.NE.0) GD TO 3060
RETURN
2000 FORMAT(15)
2001 FORMAT(2F10.4)
2002 FORMAT(3F10.4)
ENTRY BODY
C*************************
C THIS PART OF THE ROUTINE PRINTS OUT INFORMATION ABOUT THE BODY.
C*************************
IVERSON=3
WRITE(6,3000)IVERSON
IF(IBN.EQ.1) GD TO 3020
WRITE(6,3005)ZBAR,BAR
GO TO 3021
3020 WRITE(6,3100)ALNS/RAD,THETABN/RAD,ZEN
3100 FORMAT(1X,*FORE BODY IS A SPHERICALLY BLUNTED CONE OF *,F10.4,1X,1 DEGREES BENT AT AN ANGLE OF *,F10.4/11X,**THE BENT CONE ENDS AT*, BODY
3110 FORMAT(1X,*THE BODY IS SMOOTHED TO THE AFT BODY BEGINNING AT*, BODY
3120 FORMAT(1X,*THE AFT BODIES IS A MULTIPLE CONIC WITH*, BODY
3021 CONTINUE
WRITE(6,3010)(ACONE(I),ZCONE(I),I=1,NCONE)
IF(IRND.EQ.1) WRITE(6,3025)RRDD,ZRND
IF(IEFL.EQ.1) WRITE(6,3030)ZFLARE,THETAFL,THETAGL
IF(NW.EQ.0) GD TO 3060
WRITE(6,3035)
WRITE(6,3040)(THETAW(I),ZW(I),I=1,1NW)
IF(IFW.EQ.1) WRITE(6,3050)HFW,ZFW,FAW
3060 IF(NS.EQ.0) GD TO 3090
WRITE(6,3040)(THETAL(I),ZL(I),I=1,1NL)
IF(IFL.EQ.1) WRITE(6,3050)HFL,ZFL,FAIL
RETURN
3090 IF(NL.EQ.0) RETURN
WRITE(6,3095)
WRITE(6,3090)(THETAL(I),ZL(I),I=1,1NL)
IF(IFL.EQ.1) WRITE(6,3050)HFL,ZFL,FAIL
RETURN
3000 FORMAT(1H0,*PROGRAM BODY=*6X,*VERSION=*14)
3005 FORMAT(I1X,**BODY IS SPHERICALLY BLUNTED AND SPHERE ENDS AT Z=*** BODY 456
  1 E15.7,2X,** WITH B=E15.7/I1X,** AFT BODY IS A MULTIPLE CONIC WITH BODY 457
  2 **) BODY 458
3010 FORMAT(I6X,**ANGLE**F10.4** UP TO **F10.4) BODY 459
3025 FORMAT(I1X,**THE REAR OF THE BODY IS ROUNDED WITH RADIUS**F10.4/) BODY 460
  1 I1X,** THE ROUNING BEGINS AT** F10.4) BODY 461
3030 FORMAT(I1X,**THERE IS AN ELLIPTIC FLARE BEGINNING AT** F10.4/) BODY 462
  1 I1X,** THE WIND-LEE AXIS EXPANDS WITH ANGLE**, F10.4,** AND THE SIDE BODY 463
  2 AXIS EXPANDS WITH ANGLE**, F10.4) BODY 464
3035 FORMAT(I1X,**THERE IS A WIND CUT OF**) BODY 465
3040 FORMAT(I6X,**ANGLE**F10.4** BEGINNING AT**F10.4) BODY 466
3050 FORMAT(I20X,**WITH A FLAP OF HALF-WIDTH**,F10.4,** LENGTH ALONG Z-AXIS BODY 467
  1 *,F10.4**, AT=F10.4,** DEGREES**) BODY 468
3070 FORMAT(I1X,**THERE IS A SIDE CUT CENTERED AT LONGITUDE**,F10.4** OF BODY 469
  1 **) BODY 470
3095 FORMAT(I1X,** THERE IS A LEE CUT OF**) BODY 471
END BODY 472
SUBROUTINE FIELD

FIELD PRINTS THE DATA AT SOME FIX

FIELD 2

FIELD 3

FIELD 4

FIELD 5

NEWCOM 1

NEWCOM 2

NEWCOM 3

NEWCOM 4

CO3CSS 32

NEWCOM 5

NEWCOM 6

NEWCOM 7

NEWCOM 8

NEWCOM 9

NEWCOM 10

NEWCOM 11

NEWCOM 12

NEWCOM 13

NEWCOM 14

NEWCOM 15

NEWCOM 16

NEWCOM 17

NEWCOM 18

NEWCOM 19

NEWCOM 20

NEWCOM 21

NEWCOM 22

NEWCOM 23

NEWCOM 24

NEWCOM 25

NEWCOM 26

NEWCOM 27

NEWCOM 28

NEWCOM 29

NEWCOM 30

NEWCOM 31

NEWCOM 32

NEWCOM 33

NEWCOM 34

NEWCOM 35

NEWCOM 36

NEWCOM 37

NEWCOM 38

NEWCOM 39

NEWCOM 40

NEWCOM 41

NEWCOM 42

NEWCOM 43

NEWCOM 44

NEWCOM 45

NEWCOM 46

NEWCOM 47

NEWCOM 48

NEWCOM 49

NEWCOM 50

NEWCOM 51

NEWCOM 52

NEWCOM 53

NEWCOM 54

NEWCOM 55

NEWCOM 56

NEWCOM 57

NEWCOM 58

NEWCOM 59

NEWCOM 60

NEWCOM 61

NEWCOM 62

NEWCOM 63

NEWCOM 64

NEWCOM 65

NEWCOM 66

NEWCOM 67

NEWCOM 68

NEWCOM 69

NEWCOM 70

NEWCOM 71

NEWCOM 72

NEWCOM 73

NEWCOM 74

NEWCOM 75

NEWCOM 76

NEWCOM 77

NEWCOM 78

NEWCOM 79

NEWCOM 80

NEWCOM 81

NEWCOM 82

NEWCOM 83

NEWCOM 84

NEWCOM 85

NEWCOM 86

NEWCOM 87

NEWCOM 88

NEWCOM 89

NEWCOM 90

NEWCOM 91

NEWCOM 92

NEWCOM 93

NEWCOM 94

NEWCOM 95

NEWCOM 96

NEWCOM 97

NEWCOM 98

NEWCOM 99

NEWCOM 100

RETURN

3000 FORMAT(1I1,*MACH NO IS*,1PE15.7,5X,*ANGLE OF SIDE SLIP IS*,1PE15.7) FIELD 30

3100 FORMAT(1H0,*PLANЕ E*,14.3X,*ANGLE IS*,F7.2,* DEGREES*) FIELD 31

3110 FORMAT(1H0,*50X,*WINDWARD PLANE*) FIELD 32

3120 FORMAT(1H0,*50X,*LEWWARD PLANE*) FIELD 33

3140 FORMAT(1H0,*1PE14.6) FIELD 34

3666 FORMAT(1H0,*STATION IS*,15.4X,*Z IS*,1PE15.7,5X,*R IS*,1PE15.7,4X,*A IS*,1PE15.7,7,5X) FIELD 35

3667 FORMAT(1H0,*8Z IS*,1PE15.7,4X,*PHI IS*,1PE15.7,7,5X) FIELD 36

3700 FORMAT(1H0,*1PE15.7,4X,*CZ IS*,1PE15.7,4X,*CPHI IS*,1PE15.7,7,5X) FIELD 37

3710 FORMAT(1H0,*1PE15.7,4X,*C13X IS*,1PE15.7,4X) FIELD 38

3800 FORMAT(1I1,*THE AXIS IS SHIFTED UP*,F10.4,* UNITS*) FIELD 39

END
SUHRoutine OUT
OUT OUTPUTS THE WALL PRESSURES AND THE FORCES AND MOMENTS.
THIS ROUTINE IS EXECUTED AFTER CALCULATIONS ARE COMPLETED.

COMMON NC, MC, K, PNF, DINF, PH10(K), D2YAW, PI, RAD
COMMON YZ(3), YPH(11), C(25), CZ(25), CPMH(1), P(20, 25)
COMMON D(20, 25), V(20, 25), W(20, 25), ASG(20, 25)
COMMON CU(.4, 20, 25), CUP (.4, 20, 25)

COMMON /COUT/ ACH, ATTAYAW, ZENDET, XINDEF, VINF, SINF
1 1NTARGET, TARGETZ(100)
 COMMON /CBODY/ Z + B2Z, BPHM1, ZPBPM1, TANCO, DELZ
1 COMMON /CINTEG/ FNFYFA, P4XMYMZ, FNZFYZ, FAZ, MXZ, MYZ, MZZ
1 COMMON /CD3CSS/ ACH, ATTA, YAW, ZEND, XINDEF, VINF, SINF

REAL MX, MY, MZ, X, Y, Z
DIMENSION P8(1)
EQUIVALENCE (P8, P)
NAMELIST /OIJTRO/ ZREFA, REFZ, C, Zin, M2, M1, P14, M1 = 14

IF (ZREF.EQ. 0.) ZREF = ZEND * ZIN
IF (AREF.EQ. 0.) GOTO 10

REWIND 1
S NPTS = 0
S MCK = MC
READ (16) NC, MC, ACH, ATTAYAW, ACH, GANNA, PNF, DINF, PH10, K, Z
A 1NGAS = TEST + RR
1 1NFYFA, P4XMYMZ, FNZFYZ, FAZ, MXZ, MYZ, MZZ
2 PHIM(DUM, DUM, DUM, P(M), D, M = 1, MC)
3 P(DUM, DUM, DUM, P(M), D, M = 1, MC)
IF (ECF(16)) .GT. 50 GO TO 30
50 M2 = M0(MC, M1, P14) IF (M2 .LT. M1) GO TO 30
IF (MC .NE. MCS) NPTS = 1
IF (MOD(NPTS - 1, 38) .NE. 0) GOTO 75
IF (IPC0 .EQ. 0) WRITE (6, 3020)
3000 FORMAT (1H1, 10X, *MACH NO. = *F8.3, *S
1 1NGL OF ATTACK = *F8.3, *S
3 IF (IPC0 .EQ. 0) WRITE (6, 3020)
3020 FORMAT (1H1, 10X, *SURFACE PRESSURE RATIO*),
1 C0EFFICIENT),
DO 60 M = M1, M2
60 PHIM(M) = PHIM(M, P14)
WRITE (6, 3030) (PHIM(M), M = M1, M2)
3030 FORMAT (1H1, 5X, AMZ, Z0 + 15FB, 1)
WRITE (6, 3040)
3040 FORMAT (1H1, 5X, AMZ, Z0 + 15FB, 1)
75 DO 62 M = M1, M2
62 PHIM(M) = PHIM(M, P14)
NSWC/WOL/TR 77-28

202
3300 FORMAT(1H*9X,15X,18Hforce coefficients, 12X*)
1 2X,29HMoment coefficients about Z = 1PF15.6, 2X*
2 19HCenters of pressure)
3320 FORMAT(1H*5X,4H2X,10X,2HMCN, 3X, 10X, 2HCAN, 3X, 10X, 2HCY, 3X*)
1 9X,3HCMN, 3X, 9X, 3HCAN, 3X, 9X, 3HCAN, 3X, 9X, 3HCY, 3X, 9X, 3HCY, 3X)
3330 FORMAT(1H*9.3,1PE15.5)
3340 FORMAT(1H*28X,2HZ derivatives of force and moment coefficients* )
3350 FORMAT(1H*5X,2Z,20X,9X,2HCNZ, 13X, 2HCNZ, 13X, 2HCNZ, 13X, 2HCNZ)
1 *CMNZ*, 12X, *CMNZ*, 12X*
3360 FORMAT(1H*9.3,1PE16.5)
END
SUBROUTINE SAVE(Ex,EnRun,RAPO)

SAVE PRINTS OUT THE FIELD DATA FOR THE LAST IERRPR STEPS. IF THE PROGRAM BOMBS BECAUSE OF AN ERROR CONDITION, ALSO THE WALL Pressures, Forces and Moments ARE PRINTED.

COMMON NC,M,C,K,PINF,DINF,PHIO,YAW,P1,T,RAD
COMMON YZ(3),YPHI(3),C(25),CZ(25),CPHI(25),R(120,25)
COMMON D(120,25),P(120,25),U(120,25),V(120,25),W(120,25),ASO(120,25)
COMMON CY(4,20,25),CUP(4,20,25)

NEWCOMMON

COMMON /CSAVE/ EZABZ2,RpPHI,2ZPHI+TANCO+FLZ
COMMON /CBODY/ ZARZ2,RPwPHI,BPwPHI+TANCO+FLZ
COMMON /CBODY/ ZARZ2,RPwPHI,BPwPHI+TANCO+FLZ

NEWCOMMON C

*** END OF PLANK COMMON ***
COMMON /CSAVE/ IERRPR,MAS
COMMON /CSAVE/ IERRPR,MAS

DIMENSION EX(17)

ENDFILE 16

IF (IERPR .LE. 0) GO TO 75

DO 25 I=1,IERRPR

BACKSPACE 16

DO 50 I=1,IERRPR

READ (16) NC,M,C,K,PINF,DINF,PHIO,YAW,P1,T,RAD

A *NGAS,NTET,ARX

C

1 *FUZ,2Z,FZ,MYZ,MZ,2Z,FZ,MYZ,MZ

2 *F(HI)+C(M)+CZ(M)+CPHI(M)+N=1,MC

3 *(R(N,M)+U(N,M)+V(N,M)+W(N,M)+P(N,M)+D(N,M)+K=1,MC)+N=1,NC

DO 40 N=1,MAS

CALL BODY(M)

CALL FIELD

CALL OUT

STOP

END
SUBROUTINE TRANFO

TRANFO DEFINES QUANTITIES NEEDED BY SUBROUTINE TRANF WHEN THE USER READS IN THE SF(X,Y,Z) DATA POINTS

COMMON /CTRANF/ NSFD, SFD(20), SFXXD(20), SFXXD(20)

READ (5,2000) (SFD(N)+N=1,NSFD)

SFD0=NSFD-1

DX=1./FLOAT(NSFD)

TWO=1./(2.*DX) $ DX=1./DX**2

DO 40 N=SFD+1

SFXXD(N)=2.*SFXXD(N-1)-SFXXD(N-2)

40 CONTINUE

SFXXD(SFD)=2.*SFXXD(SFD-1)-SFXXD(SFD-2)

WRITE (6,3300)

FORMAT(1H14X*NSFD,N1SFXD,N1SF),(1H14X*NSFD,SFXD,N1SFXXD)

DO 125 N=1,NSFD

WRITE (6,3400)

FORMAT(1H15,1P6E20.6)

RETURN

END
SUBROUTINE TRANGD

TRANGO DEFINES QUANTITIES NEEDED BY SUBROUTINE

TRANG WHEN THE USER READS IN THE PHI VALUES

COMMON Q,CM,M,K,PINF,INF,PHI(1),YAN,YAN,P,RAN
COMMON Y(3),YPH(3),C(25),CZ(25),CPHI(25),R(20,25)
COMMON D(20,25),P(20,25),U(20,25),W(20,25),ASG(20,25)
COMMON CU(4,20,25),CUP(4,20,25)

*** END OF RANK COMMON ***

COMMON /CTRANG/,NSGD(50),SGD(50),SGD(25),SGDY(25),SGYD(25)

1 ,GYMY,GYMDY,GYPDY,GYPDY

NEWCOMMON

2 NEWCOMMON

3 NEWCOMMON

4 NEWCOMMON

5 NEWCOMMON

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

**NSWC/WOL/TR 77-28**

READ (5,200) (SGD(M),M=1,NSGD)

200 FORMAT(KI10.0)

NSGD=NSGD-1

DY=1./FLOAT(NSGD)

TWOY=1./12.*DY $ DYS=1./DY**2

DO 25 M=1,NSGD

25 SGD(M)=SGD(M)/SGD(NSGD)

DO 50 M=2,NSGD

SGYD(M)=(SGYD(M-1)-SGD(M-1))*TWOY

50 SGYD(M)=(SGY(M-1)-2.*SGD(M-1)+SGD(M-1))*DYS

IF (PHIO.LE.2.*PI-1.E-6) GO TO 75

NOTE THAT SGD(Y+1)=SGD(Y)*1 FOR NON-SYMMETRIC PROBLEM

SGYD IS ASSUMED LINEAR ON (-DY,2DY) AND (1-2DY,1+DY)

GO TO 100

NOTE THAT FOR SYMMETRIC PROBLEM (PHIO=360)

75 SGYD(1)=2.*SGYD(2)-SGYD(3)

SGYD(2)=2.*SGYD(1)-SGYD(2)

SGYD(3)=2.*SGYD(2)-SGYD(3)

100 WRITE (6,1330)

1330 FORMAT (1H1,4X,3E10.6)

13350 FORMAT (1H1,4X,4I,15X,*PHI*,18X,*SG*,17X,*SGY*,16X,*SGYY*)

** NSGD=NSGD-1 S 15X,*GYM*,*GYMD*,*GYPDY*,*GYPDY**

WRITE (6,3350) M,GYMY,GYMDY,GYPDY

135 FORMAT (I4)

3350 FORMAT (1H1,4X,4I,15X,*PHI*,18X,*SG*,17X,*SGY*,16X,*SGYY*)

WRITE (6,3350) M,GYMY,GYMDY,GYPDY

RETURN

END

C
SUBROUTINE DMPSORT(NAME,KNT,Z,K,N,N,value)
C
NAME IS THE NAME OF THE ROUTINE
C
KNT IS THE NUMBER FROM WHICH NAME WAS CALLED
C
Z IS THE Z VALUE
C
K IS THE STATION NO.
C
N IS THE PLANE NO.
C
N IS THE RADIAL POINT NO.
C
VALUE IS THE ARGUMENT OF SORT ROOT
C
WRITE (14,3000) NAME,KNT,VALUE,Z,K,N,N
3000 FORMAT(1H19.1N ROUTINE **A10,** AT CALL NO.**13,**,
1     *NEGATIVE SORT ROOT OF**1PE15.6*,
2     ** FOR Z**1PE15.6**315)
CALL SAVE(DUM,DUM,DUM)
STOP
END
NSWC/WOL/TR 77-28

DISTRIBUTION

Copies

Director
U.S. Naval Research Laboratory
Washington, D.C. 20390
Library
Code 6503

ICASE, NASA Langley
MS 132C
Hampton, Virginia 23665
Dr. James Ortega

NASA
Langley Research Center
Langley Station
Hampton, Virginia 23665
MS/185 Technical Library
Aero & Space Mech Div.
Dennis Bushnell
Ivan Beckwith
R. Trimpi
Julius Harris

NASA
Lewis Research Center
21000 Brookpark Road
Cleveland, Ohio 44135
Library 60-3
Chief, Wind Tunnel & Flight Div.

NASA
George C. Marshall Space Flight Center
Huntsville, Alabama 35812
Mr. T. Reed, R-AERO-AU
Mr. W. K. Dahn, BB31

NASA
600 Independence Avenue, S. W.
Washington, D. C. 20546
F. C. Schwenk, Director,
Research (Code RR)

NASA
P.O. Box 33
College Park, Maryland 20740
Headquarters, Arnold Engineering
Development Center
Arnold Air Force Station
Tennessee 37389
Library/Documents
R. W. Henzel, TD
CAPT C. Tirres/DYR
CAPT K. Kushman

von Karman Gas Dynamics Facility
ARO, Inc.
Arnold Air Force Station
Tennessee 37389
Dr. J. D. Whitfield
Dr. J. Adams

Commanding Officer
Harry Diamond Laboratories
Washington, D.C. 20438
Library

Commanding General
U.S. Army Missile Command
Redstone Arsenal
Alabama 35809
AMSMI-RR
Chief, Document Section
AMSMI-RDK, R. A. Deep
AMSMI-RDK, T. R. Street

Department of the Army
Office of the Chief of
Research and Development
ABMDA, The Pentagon
Washington, D.C. 20350

Commanding Officer
Picatinny Arsenal
Dover, New Jersey 07801
SMUPA-VC-3 (A. A. Loeb)

Commander (ADL)
Naval Air Development Center
Johnsville, Pennsylvania 18974
Dr. R. K. Lobb
DISTRIBUTION

Copies

NASA Ames Research Center
Moffett Field, California  94035
   Dr. M. Horstman
   F. Kutler
   J. Rakich
   R. MacCormack
   Library

AFFDL
Wright Patterson Air Force Base
Ohio  45433
   Mr. Melvin Buck
   D. J. Harney
   W. Hankey
   J. Shang
   K. Stetson
   Art Lewis

Wright Aeronautical Laboratories
Wright-Patterson Air Force Base
Dayton, Ohio  45433
   Technical Library

Air Force Materials Laboratory
Wright Patterson Air Force Base
Dayton, Ohio  45433
   Dr. William Kessler

Naval Research Laboratory
Washington, D.C.
   Dr. J. Boris

ONR Branch Office/Pasadena
1030 East Green Street
Pasadena, California  91101
   Dr. Richard Lau

Aerospace Engineering Program
University of Alabama
P.O. Box 6307
University of Alabama  35486
   Prof. W. K. Rey, Chm.

AME Department
University of Arizona
Tucson, Arizona  85721
   Dr. L. B. Scott
Case Western Reserve University  
Division of Fluid, Thermal and Aerospace Engineering  
Cleveland, Ohio 44106  
Dr. Eli Reshotko

The Catholic University  
Washington, D.C. 20017  
Dr. C. C. Chang

University of Cincinnati  
Cincinnati, Ohio 45221  
Department of Aerospace Engineering  
Dr. Arnold Polak  
Dr. Michael Werle

Department of Aerospace Engineering Sciences  
University of Colorado  
Boulder, Colorado 80302

Cornell University  
Graduate School of Aero. Engineering  
Ithaca, New York 14850  
Prof. A. R. George

University of Delaware  
Mechanical and Aeronautical Engineering Dept.  
Newark, Delaware 19711  
Dr. James E. Danberg

Georgia Institute of Technology  
225 North Avenue, N. W.  
Atlanta, Georgia 30332  
Dr. Arnold L. Ducoffe

Technical Reports Collection  
Gordon McKay Library  
Harvard University  
Div. of Engineering and Applied Physics  
Pierce Hall, Oxford Street  
Cambridge, Massachusetts 02138  
Dr. George Carrier
NSWC/WOL/TR 77-28

DISTRIBUTION

Copies

Dept. of Aero-Astro Engineering
Ohio State University
2036 Neil Avenue
Columbus, Ohio 43210
   Engineering Library

The Pennsylvania State Univ.
University Park, Pennsylvania 18602
   Dept. of Aero. Engr.
   Hammond Bldg.
   Library, Documents Section

Bevier Engr. Library
126 Benedum Hall
University of Pittsburgh
Pittsburgh, Pennsylvania 15261

Princeton University
D-214 Engr. Quadrangle
Princeton, New Jersey 08540
   Dr. I. E. Vas
   Dr. W. D. Hayes

Purdue University
School of Aeronautical and
   Engineering Sciences
Lafayette, Indiana 47907
   Dr. B. Reese, Head, Dept. of
   Aero. and Astro.
   Dr. C. P. Kentzer

Rensselaer Polytechnic Institute
Troy, New York 12181
   Dept. of Aeronautical and
   Engineering and Astronautics

Department of Mechanical
   Industrial and Aerospace Engr.
Rutgers - The State University
New Brunswick, New Jersey 08903
   Dr. R. H. Page

Stanford University
Stanford, CA 94305
   Librarian, Dept. of Aeronautics
   and Astronautics
   Prof. J. Oliger, Dept. of Computer Sciences
Stevens Institute of Technology  
Hoboken, New Jersey 07030  
Mechanical Engineering Dept.  
Library

The University of Texas at Austin  
Applied Research Laboratories  
P.O. Box 8029  
Austin, Texas 78712  
Director  
Engr. S.B. 114B/Dr. Friedrich

University of Toledo  
2801 W. Bancroft  
Toledo, Ohio 43606  
Dept. of Aero Engineering

University of Virginia  
School of Engineering and Applied Science  
Charlottesville, Virginia 22091  
Dr. I. D. Jacobson

University of Washington  
Seattle, Washington 98105  
Engineering Library  
Dept. of Aeronautics and Astronautics

West Virginia University  
Morgantown  
West Virginia 26506  
Library

Federal Reports Center  
University of Wisconsin  
Mechanical Engineering Building  
Madison, Wisconsin 53706  
S. Reilly

Los Alamos Scientific Laboratory  
P.O. Box 1663  
Los Alamos, New Mexico 87544  
Reports Library

Institute for Defense Analyses  
400 Army-Navy Drive  
Arlington, Virginia 22202  
Classified Library
Kaman Sciences Corporation
P.O. Box 7463
Colorado Springs, Colorado 80933
Library
Mr. F. Barbera
Mr. D. Foxwell

Kaman Sciences Corporation
Avidyne Division
83 Second Avenue
Burlington, Massachusetts 01803
Dr. J. R. Ruetenik

Rockwell International
B-1 Division
Technical Information Center
(BA08) International Airport
Los Angeles, California 90009

Rockwell International Corporation
Technical Information Center
4300 E. Fifth Avenue
Columbus, Ohio 43216

M.I.T. Lincoln Laboratory
P.O. Box 73
Lexington, Massachusetts 02173
Library A-082
Dr. A. B. Wardlaw

The RAND Corporation
1700 Main Street
Santa Monica, California 90406
Library - D

Aerojet Electrosystems Co.
1100 W. Hollyvale Avenue
Azusa, California 91702
Engineering Library

The Boeing Company
P.O. Box 3999
Seattle, Washington 98124
87-67

United Aircraft
Research Laboratories
East Hartford
Connecticut 06108
Dr. William M. Foley
DISTRIBUTION

Calspan Corporation
4455 Genesee Street
Buffalo, New York 14221
Library

Air Force Univ. Library
(SE) 63-578
Maxwell Air Force Base
Alabama 36112

McDonnell Company
P.O. Box 516
St. Louis, Missouri 63166
R. D. Detrich, Dept. 209
Bldg. 33

McDonnell Douglas Astronautics
Company - West
5301 Bolsa Avenue
Huntington Beach, California 92647
368, B4A0
J. S. Murphy, A3
J. Copper
J. Xerikos
W. Shaw

Fairchild Hiller
Republic Aviation Division
Farmingdale, New York 11735
Engineering Library

General Applied Science Labs. Inc.
Merrick and Stewart Avenues
Westbury, Long Island
New York 11590
Dr. F. Lane
L. M. Nucci

General Electric Company
R&D Lab. (Comb. Bldg.)
Schenectady, New York 12301
Dr. H. T. Nagamatsu
NSWC/WOL/TR 77-28

DISTRIBUTION

Northrop Norair
3901 West Broadway
Hawthorne, California 90250
Tech. Info. 3360-32

Government Documents
The Foundren Library
Rice Institute
P.O. Box 1892
Houston, Texas 77001

Grumman Aircraft Engineering Corporation
Bethpage, Long Island
New York 11714
Dr. R. E. Melnik

Marquardt Aircraft Corp.
16555 Saticoy Street
Van Nuys, California 91409
Library

ARDE Associates
P.O. Box 286
580 Winters Avenue
Paramus, New Jersey 07652
Librarian

Aerophysics Company
3500 Connecticut Ave., N. W.
Washington, D.C. 20003
Mr. G. D. Boehler

Aeronautical Research Associates of Princeton
50 Washington Road
Princeton, New Jersey 08540
Dr. C. duP. Donaldson

General Research Corporation
5383 Hollister Avenue
P.O. Box 3587
Santa Barbara, California 93105
Technical Information Office
DISTRIBUTION

Department of Mathematics
University of California at Berkeley
Berkeley, California 94720
  Prof. A. Chorin

Science Applications Inc.
101 Continental Bldg.
El Segundo Blvd.
El Segundo, California 90245
  L. Cassel

University of Tennessee
Department of Mathematics
Knoxville, Tennessee 37916
  Prof. Max Gunzburger