A THEORETICAL METHOD
FOR THE ANALYSIS AND DESIGN
OF AXISYMMETRIC BODIES

T. D. Beatty

Prepared by
MCDONNELL DOUGLAS AIRCRAFT CORPORATION
Long Beach, Calif. 90801
for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1975
A theoretical method is presented for the computation of the flow field about an axisymmetric body operating in a viscous, incompressible fluid. A potential flow method is used to determine the inviscid flow field. These results yield the boundary conditions for the boundary layer solutions. Boundary layer effects in the forces of displacement thickness and empirically modeled separation streamlines are accounted for in subsequent potential flow solutions. This procedure is repeated until the solutions converge. An empirical method is used to determine base drag allowing configuration drag to be computed.
A theoretical method is presented for the computation of the flow field about an axisymmetric body operating in a viscous incompressible fluid. This approach combines a smoothing routine, a potential flow method based on a surface source distribution, and a finite-difference boundary-layer method to accomplish the analysis. An empirical method used for modeling separated flow is shown to work reasonably well for cases of extreme flow separation. Results obtained by this method are presented which show very good agreement with experimental data. Suggestions are made for extending this method both to include a better model for separated flow and to calculate the "viscous" flow about axisymmetric bodies at angle of attack. A detailed instruction manual for inputing data to the computer program is given in Appendix A. Appendix B contains the necessary information to place this program on to a computer. This appendix also contains a complete description of output parameters from the computer program, as well as basic flow charts of some of the major subroutines. Appendix C contains a complete listing of the computer program for operation on either a CDC or an IBM computer.
<table>
<thead>
<tr>
<th>TABLE OF CONTENTS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
<td>iii</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>DEFINITION OF SYMBOLS</td>
<td>3</td>
</tr>
<tr>
<td>TECHNICAL DISCUSSION</td>
<td>6</td>
</tr>
<tr>
<td>Geometry Definition</td>
<td>6</td>
</tr>
<tr>
<td>Point distribution</td>
<td>6</td>
</tr>
<tr>
<td>Smoothness of input coordinates</td>
<td>6</td>
</tr>
<tr>
<td>Potential Flow Method</td>
<td>7</td>
</tr>
<tr>
<td>Boundary Layer Method</td>
<td>9</td>
</tr>
<tr>
<td>Basic boundary layer equations</td>
<td>9</td>
</tr>
<tr>
<td>Eddy-viscosity equations</td>
<td>12</td>
</tr>
<tr>
<td>Low Reynolds number effects</td>
<td>14</td>
</tr>
<tr>
<td>Transverse curvature</td>
<td>14</td>
</tr>
<tr>
<td>Transition region effect</td>
<td>15</td>
</tr>
<tr>
<td>Boundary layer transition location</td>
<td>16</td>
</tr>
<tr>
<td>Calculation Procedure</td>
<td>17</td>
</tr>
<tr>
<td>EXPERIMENTAL CORRELATIONS</td>
<td>22</td>
</tr>
<tr>
<td>CONCLUDING REMARKS</td>
<td>24</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>26</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>49</td>
</tr>
<tr>
<td>APPENDIX C</td>
<td>75</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>253</td>
</tr>
<tr>
<td>ILLUSTRATIONS</td>
<td>258</td>
</tr>
</tbody>
</table>
A THEORETICAL METHOD
FOR THE ANALYSIS AND DESIGN
OF AXISYMMETRIC BODIES

by T. D. Beatty
McDonnell Douglas Aircraft Corporation

One of the ultimate goals in aerodynamics is the achievement of the ability to obtain the real fluid flow field about an arbitrary three-dimensional configuration by theoretical calculation rather than by resorting to expensive and time-consuming wind tunnel tests. The exact treatment of this problem requires the solution of the full Navier-Stokes equations, which is currently not practical. However, a good approximation to this real flow can be obtained by displacing the surface boundaries of the original body to account for viscosity as shown by Thwaites in Reference (1).

This technique of displacing the boundary surface to obtain a viscous solution has been used in two-dimensional flows quite successfully, as shown in References 2, 3, and 4. The extension of this approach to three-dimensional flow requires that appropriate computational routines be available to calculate the potential and viscous flow parameters. A potential flow routine which can calculate the flow about arbitrary three-dimensional bodies is available (Reference 5), although the comparable three-dimensional boundary layer method is not currently in the state of the art. At the present time, the general three-dimensional problem cannot be solved. However, both an axisymmetric potential flow method and an axisymmetric boundary layer method which can calculate the inviscid and viscous flow field about a body of revolution at zero degrees angle of attack are currently available.

Because of its simple nature and its common appearance in fluid dynamics, it was decided that a body of revolution would be a good starting point for the development of a three-dimensional method for calculating inviscid and viscous flow fields.

The axisymmetric potential flow routine (References 6 & 7), used in the present method was developed at the Douglas Aircraft Company under the guidance of A. M. O. Smith and has proven over the years to be an extremely versatile
and accurate method, as well as the only purely axisymmetric potential flow method, generally available in industry today. This method has been well disseminated throughout industry; only a brief discussion will, therefore, be presented in a following section.

The boundary layer method presented in this report (Reference 8) is a finite difference technique which uses an eddy-viscosity concept to replace the Reynolds shear stress term. Since this method is relatively new and has been modified extensively since Reference 8 was reported, a detailed description will be presented.

The capability of the present method to determine the viscous flow about axisymmetric bodies is shown by correlations between the calculated results and experimental data.

Recommendations are presented for extending the present method to the calculation of the flow field about axisymmetric bodies at angle of attack.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Damping length or frontal area, wherever applicable</td>
</tr>
<tr>
<td>A'</td>
<td>Damping constants</td>
</tr>
<tr>
<td>C_F</td>
<td>Total skin friction coefficient</td>
</tr>
<tr>
<td>C_p</td>
<td>Pressure coefficient</td>
</tr>
<tr>
<td>c</td>
<td>Chord</td>
</tr>
<tr>
<td>c_f</td>
<td>Local skin friction coefficient $\tau_w/(\frac{1}{2})\rho u_e^2$</td>
</tr>
<tr>
<td>D</td>
<td>Maximum diameter</td>
</tr>
<tr>
<td>f</td>
<td>Dimensionless stream function</td>
</tr>
<tr>
<td>G</td>
<td>Spot formation parameter</td>
</tr>
<tr>
<td>H</td>
<td>Shape factor, $\theta/\delta$</td>
</tr>
<tr>
<td>K_l</td>
<td>Mixing-length constant</td>
</tr>
<tr>
<td>k</td>
<td>Power to determine 2-D or axisymmetric flow</td>
</tr>
<tr>
<td>L</td>
<td>Reference body length</td>
</tr>
<tr>
<td>\ell</td>
<td>Mixing length</td>
</tr>
<tr>
<td>P'</td>
<td>Pressure gradient parameter</td>
</tr>
<tr>
<td>R_C</td>
<td>Chord Reynolds number, $u_\infty c/\nu$</td>
</tr>
<tr>
<td>R_D</td>
<td>Diameter Reynolds number, $u_\infty D/\nu$</td>
</tr>
<tr>
<td>R_x</td>
<td>Local Reynolds number, $u_e x/\nu$</td>
</tr>
<tr>
<td>R_\theta</td>
<td>Momentum thickness Reynolds number, $U_e \theta/\nu$</td>
</tr>
<tr>
<td>r</td>
<td>Radial distance from axis of revolution</td>
</tr>
<tr>
<td>r_0</td>
<td>Local radius of body of revolution</td>
</tr>
<tr>
<td>T</td>
<td>Absolute temperature, °K or °R.</td>
</tr>
<tr>
<td>t</td>
<td>Transverse curvature term</td>
</tr>
<tr>
<td>U_\infty</td>
<td>Free stream velocity</td>
</tr>
<tr>
<td>u_r</td>
<td>Friction velocity, $\sqrt{\frac{\tau_w}{\delta}}$</td>
</tr>
<tr>
<td>u</td>
<td>x component of velocity</td>
</tr>
</tbody>
</table>
Velocity at edge of boundary layer

$u_e$

$\nu$ - component of velocity

$x$ - Distance along surface measured from leading edge or from stagnation point

$y$ - Distance normal to the surface of the body

$\alpha$ - Angle between normal to the surface $y$ and the radius $r$

$\alpha$ - Constant in outer eddy viscosity equation

$\beta$ - Dimensionless velocity gradient term, $\beta = (2\xi/u_e)(du_e/d\xi)$

$\gamma_{Tr}$ - Transitional parameter

$\delta$ - Boundary layer thickness

$\delta^*$ - Boundary layer displacement thickness

$\epsilon$ - Eddy viscosity

$\epsilon^+$ - Ratio of eddy viscosity to kinematic viscosity, $\epsilon/\nu$

$\eta$ - Transformed $y$-coordinate

$\theta$ - Momentum Thickness

$\nu$ - Dynamic viscosity

$\nu$ - Kinematic viscosity

$\xi$ - Transformed $x$-coordinate

$\rho$ - Density

$\tau$ - Shear stress

**SUBSCRIPTS**

c - Switching point between the inner and outer eddy viscosity formulas

e - Outer edge of boundary layer

i - Inner region

l - Laminar
o Outer Region
T Turbulent
Tr Transition
w Wall
\infty Free-stream conditions

Primes denote differentiation with respect to \eta.
The geometry input to the Douglas Neumann Potential Flow Program must satisfy two primary requirements: the coordinates must be distributed properly and the surface curvature must be smooth. These requirements are easily achieved on an analytical body shape, since the input coordinates may be calculated exactly for any prescribed distribution. However, some method of determining accurate input coordinates for an arbitrary axisymmetric body is necessary, since the body may not always be amenable to exact analytical definition. The approach adopted in the following method is to assume that the coordinates are input in the proper distribution about the body, but that they are not necessarily smooth. These two requirements will be discussed in some detail in the following sections.

**Point distribution.** - In order to obtain a high degree of accuracy in defining a pressure distribution when using the Douglas Neumann Potential Flow program, surface coordinates should be concentrated in regions of high surface curvature where rapid changes in the surface pressures would be expected. Since the total number of points per body is fixed, the distribution of these points about the body contour becomes extremely important. The Neumann program uses the input coordinates to create linear segments between points, thus approximating the body by a series of Frusums of Cones. The basis distribution required is then quite simple: more points and thus smaller segment sizes in regions of high curvature and less points and thus larger segment sizes in the other areas of the body. The basic guidelines to follow to insure proper point distribution are simply that the surface lengths of adjacent elements should not change by more than twenty to thirty percent and the maximum length of any segment should not exceed either five percent of the body chord or fifty percent of the local body thickness.

**Smoothness of input coordinates.** - The Douglas Neumann program, or any similar potential flow method, is sensitive to the derivative of the surface slopes, or the curvature of the surface. The surface defined by the input coordinates must therefore have smooth first and second derivatives. The approach used in the
present method to smooth these coordinates, is a five point smoothing routine, which assumes that the input coordinates are smooth and continuous to graphical accuracy, i.e., points are chosen from a small graph (approximately a 10 inch chord). The output points from this routine will be moved very slightly to smooth the derivatives, but this movement will be negligible as far as the body shape is concerned. The equations used to accomplish this smoothing are as follows:

\[
\bar{x}_j = \frac{1}{16} \left\{ -x_{j-2} + 4x_{j-1} + 10x_j + 4x_{j+1} - x_{j+2} \right\} \tag{1a}
\]

\[
\bar{y}_j = \frac{1}{16} \left\{ -y_{j-2} + 4y_{j-1} + 10y_j + 4y_{j+1} - y_{j+2} \right\} \tag{1b}
\]

where \( x_j \) and \( y_j \) are the unsmoothed input coordinates

and \( \bar{x}_j \) and \( \bar{y}_j \) are the smoothed coordinates.

Potential Flow Method

The Douglas Neumann method, (References 6 and 7) is very general in that it can calculate the potential flow about virtually any body. There is no restriction, for example, to slender bodies; in fact, the "body" in question need not be a single body but may be an ensemble of bodies. In principle, the calculated solution may be made as accurate as desired by suitably refining the numerical procedure; accordingly, the so-called Neumann method is designated an exact method in this sense.

The Neumann method is based on the use of a distribution of source density over the body surface. Applying the condition of zero normal velocity on the body surface yields an integral equation for the source distribution. Specifically, the equation is a Fredholm integral equation of the second kind over the body surface. Once this has been solved for the source distribution, all flow quantities of interest, i.e., velocity, pressure, etc., can be calculated by rapid straightforward procedures. To implement this method on a computer, the body surface is approximated by a large number of small surface segments, over each of which the source density is assumed constant. The
integral equation is replaced by a set of linear algebraic equations for the values of the source density on the segments. Input to the computer program consists of the coordinates of a set of points defining the body surface; these points are then used to determine the surface segments for approximating the body. There is no assumption made that the body can be analytically represented.

The usefulness of potential flow with its neglect of viscosity and compressibility is due to the fact that it is a good approximation to real flow under a wide variety of circumstances. With regard to viscosity, the program obtains useful results except in regions of catastrophic separation. To verify the usefulness of potential flow as a predictor of real flow, results calculated by the Neumann program have been compared with experimental data. Several collections of comparisons have been made. Reference 9 was a very complete collection but is now rather old. Reference 10 is a more recent collection that shows a smaller number of comparisons. In the calculation of the viscous flow about axisymmetric bodies it is necessary to add the boundary layer displacement thickness to the body as will be shown in a subsequent section. This results in an "open" trailing edge body. This "open" body can be evaluated by the Neumann program without any difficulty even though the boundary surface does not close. Reference 11 presents an explanation of this phenomenon which proceeds as follows: for a closed body the integral of the source density over the body is zero; for an "open trailing edge body, this integral is not zero, and a streamtube leaves the trailing edge of the open body which proceeds downstream and approaches infinity parallel to \( \hat{u}_\infty \) as a constant cross section streamtube. Thus, the flow that is calculated may be thought of as that about a semi-infinite body consisting of the open body and an extension defined by this streamtube. The shape of the extension is unknown but is presumably unique, having both zero normal velocity and zero source density.

The potential flow program has many useful options available which do not pertain directly to the present development. The details of these options are described in References 12 through 17.
Boundary Layer Method

Basic boundary layer equations. — The calculation of the viscous flow over an axisymmetric body involves the solution of the laminar and turbulent flow equations. For laminar flows, the problem is strictly mathematical because the governing differential equations can be written exactly. For turbulent flows on the other hand, an exact solution of the governing equations is not possible. Consequently, in order to proceed at all, one must rely on a certain degree of empiricism. In the past, most of the work in this area has concentrated on so-called momentum and/or energy integral methods as a means of evaluating the viscous flow parameters. Thus, the exact mathematical solution to the problems of the turbulent flow was bypassed, leading to fast and simple methods with varying degrees of accuracy. These methods usually rely quite heavily on empirical correlations and generally are restricted to a limited range of flow conditions.

The Douglas Boundary-Layer Method (Reference 8), eliminates many of the disadvantages of the integral methods by proceeding to solve the full partial-differential equations governing the flow, thereby, being classified as a differential method. For two-dimensional and axisymmetric incompressible flows, turbulent boundary-layer equations contain terms involving time means of fluctuating velocity components known as Reynolds stress terms. At present the exact relationship between these terms and the mean velocity distribution in the boundary layer still remains unknown. In the present method, a relation based on the eddy-viscosity concept is used giving highly satisfactory results for a variety of flow conditions.

If the normal-stress terms are neglected, the incompressible turbulent boundary-layer equations for two-dimensional and axisymmetric flows can be written as in Reference 8:

Continuity
\[ \frac{\partial}{\partial x} \left[ r u \right] + \frac{\partial}{\partial y} \left[ r v \right] = 0 \] (2)
Momentum

\[ u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = u_e \frac{du}{dx} + \frac{1}{\rho_\infty} \frac{1}{k} \frac{\partial}{\partial y} \left( k \tau \right) \]  

(3)

where

\[ \tau = \tau_k + \tau_t \]

with

\[ \tau_k = u_\infty \frac{\partial u}{\partial y} \quad \text{(For laminar flow only)} \]  

(4)

\[ \tau_t = - \rho_\infty u'u'v' \quad \text{(Additional term due to turbulent flow)} \]

and

\[ u'u'v' = \text{Reynolds shear stress term} \]

\[ k = 0 \quad \text{for two-dimensional flow} \]

\[ k = 1 \quad \text{for axisymmetric flow} \]

The basic notation and coordinate scheme are shown in Figure 1, where \( u_\infty \) is a reference velocity and \( u_e(x) \) is the velocity just outside the boundary layer. The coordinate system is a curvilinear one in which \( x \) is the distance along the surface measured from the stagnation point or leading edge, and \( y \) is measured normal to the surface. Within the boundary layer, the velocity components in the \( x \)- and \( y \)-directions are \( u \) and \( v \), respectively. The body radius is \( r_0 \).

The boundary conditions for equation (3) are

\[ u(x,0) = 0 \]  

(5a)

\[ v(x,0) = 0 \]  

(5b)

\[ \lim_{y \to \infty} u(x,y) = u_e(x) \]  

(5c)

Before equations (2) and (3) can be solved, they must be transformed to a coordinate system which removes the singularity at \( x = 0 \) and stretches the coordinate normal to the flow direction. First, these equations are placed in an almost two-dimensional form by the Probstein-Elliott transformation (Reference 18):

\[ d\overline{x} = \left[ \frac{r_0(x)}{L} \right]^{2k} dx \]  

(6)

\[ d\overline{y} = \left[ \frac{r(x,y)}{L} \right]^k dy \]  

(7)
where \( r_0(x) \) is the body radius and \( r(x,y) \) is a radius which accounts for the transverse curvature effect which will be subsequently discussed.

A stream function \( \psi \) is defined that satisfies the continuity equation (2):

\[
\frac{\partial \psi}{\partial y} = r'u \quad \frac{\partial \psi}{\partial x} = -r'v \quad \psi = \frac{\psi}{L}
\]  

(8)

The resulting equations are transformed by the Levy-Lees transformation (Reference 19) in order to remove the singularity at \( x = 0 \) and stretch the coordinates in the \( \bar{x} \) and \( \bar{y} \) directions. The Levy-Lees transformations are:

\[
d\bar{x} = \rho_0 u_0 u_e \frac{dx}{\bar{x}}
\]

\[
d\bar{y} = \frac{\rho_0 u_0}{(2\xi)^2} \frac{dy}{\bar{y}}
\]

(9a)

(9b)

A dimensionless stream function, \( \bar{\psi} \), is introduced which is related to \( \psi \) as follows:

\[
\bar{\psi} = (2\xi)^b f(\xi, \eta)
\]

(10)

Combining the Levy-Lees and the Probstein-Elliott transformations given above we have

\[
d\bar{x} = \rho_0 u_0 u_e \left[ \frac{r_0(x)}{L} \right]^{2k} dx
\]

\[
d\bar{y} = \frac{\rho_0 u_e}{(2\xi)^2} \left[ \frac{r(x,y)}{L} \right]^k dy
\]

(11a)

(11b)

Introducing an eddy viscosity term to account for the Reynolds shear stress terms,

\[
\varepsilon \equiv \frac{u'u'}{\bar{u}} \quad \varepsilon^+ \equiv \frac{\varepsilon}{U}
\]

(12)

and a transverse curvature term \( \beta \) along with a pressure parameter term

\[
\beta = \frac{2\varepsilon}{u_e} \frac{du}{d\xi}
\]

(13)

The momentum equation (3) then becomes, with \( f' = u/u_e \),

\[
\left[ 1 + \beta \right]^{2k} (1 + \varepsilon^+) \left[ f'' - f' \right] + f' f'' + \beta \left[ 1 - (f')^2 \right] = 2\xi \left[ f' \frac{\partial f'}{\partial \xi} - f'' \frac{\partial f}{\partial \xi} \right]
\]

(14)
The boundary conditions given by equation (5) become

\[ f(\xi,0) = f_w = 0 \]  
\[ f'(\xi,0) = 0 \]  
\[ \lim_{\eta \to \infty} f'(\xi,\eta) = 1 \]  

The momentum equation is then solved by a very efficient numerical scheme developed by Keller, (Reference 20) and applied to boundary layer calculations by Cebeci and Keller, (References 21 and 22).

**Eddy viscosity equations.** The eddy viscosity concept is used to relate the time-mean fluctuating velocities to a mean velocity distribution as given in equation (12)

\[ \varepsilon = -\frac{u'V'}{\partial u/\partial y} \]  

A two-layer model of the eddy viscosity within the boundary layer will be used as shown in figure 2.

In the inner region of the boundary layer an eddy viscosity model, based on Prandtl's mixing-length theory, is used:

\[ \varepsilon_1 = \ell^2 \left| \frac{\partial u}{\partial y} \right| \]  

where \( \ell \), the mixing length is given by

\[ \ell = K_1 Y \]  

A modified expression for \( \ell \) has been developed by Van Driest (Reference 23) to account for the viscous sublayer close to the wall. This modification is

\[ \ell = K_1 Y \left[ 1 - e^{-Y/A} \right] \]  

where \( A \) is given by

\[ A = A^+ \frac{V_\infty}{N} \left[ \frac{r_w}{\rho_\infty} \right]^{-\frac{1}{2}} \]
and

\[ A^+ = 26.0 \]  \hspace{1cm} (20) 

\[ N = \left[ 1 - 11.8 \, p^+ \right]^{-\frac{1}{2}} \] \hspace{1cm} (21) 

\[ p^+ = \frac{\nu \infty}{u_t^3} \frac{du}{d\xi} \rho \infty u \infty \left( \frac{r_0}{L} \right)^{2k} \] \hspace{1cm} (22) 

\[ u_t = \left( \frac{r_0}{\rho \infty} \right) \] \hspace{1cm} (23)

Now for axisymmetric flows the value of \( \lambda \) is replaced by

\[ \lambda = 0.4 r_0 \ln \left( \frac{r}{r_0} \right) \left[ 1 - e^{-\frac{r_0}{A} \ln \left( \frac{r}{r_0} \right)} \right] \] \hspace{1cm} (24) 

which is developed in reference 24. If transverse curvature effects are desired then

\[ \frac{r}{r_0} = \frac{r_0 + Y \cos \alpha}{r_0} = 1 + \frac{Y}{r_0} \cos \alpha \]

\[ = 1 + \tau \] \hspace{1cm} (25)

where \( \tau = \frac{Y}{r_0} \cos \alpha \)

then \( \lambda \) becomes

\[ \lambda = 0.4 r_0 \ln \left( 1 + \tau \right) \left[ 1 - e^{-\frac{r_0}{A} \ln (1 + \tau)} \right] \] \hspace{1cm} (26)

The eddy viscosity in the outer region of the boundary layer is given by

\[ \varepsilon_0 = \alpha u \infty \delta^*_k \] \hspace{1cm} (27)

where \( \delta^*_k \) is the boundary layer displacement thickness defined by
\[ \delta^* = \int_{0}^{n_{\infty}} \left[ 1 - \left( \frac{u}{u_e} \right) \right] \, dn \]  

(28)

which in the transformed plane becomes

\[ \delta^*_k = \left[ \frac{u}{r_0} \right]^k \frac{(2\xi)^{k_2}}{\rho \infty u_e} \int_{0}^{n_{\infty}} (1-\xi^2)(1+t)^{-k} \, dn \]  

(29)

where

\[ 1+t = \left[ 1 + \frac{2L \cos \alpha}{r_0^2} \left( \frac{2\xi}{\rho \infty u_e} \int_{0}^{n_{\infty}} dn \right)^{k_2} \right]^{1/2} \]  

(30)

This relationship for \( \varepsilon_0 \) is the same for two-dimensional or axisymmetric flows as shown in Reference 24.

**Low Reynolds number effects.** The calculation of turbulent boundary layers about two-dimensional and axisymmetric bodies must often be done at low Reynolds number, i.e., momentum thickness Reynolds number, \( R_\theta \), less than 6000. Most of the boundary layer methods including the one presented above are based on empirical data which were obtained at high Reynolds numbers. A correction term to account for low Reynolds numbers which was developed by Cebeci (Reference 25) based on prior work by Coles (Reference 26) is, therefore, applied to the outer eddy viscosity by varying the \( \alpha \) in equation (27) with \( R_\theta \) in the following manner.

\[
\begin{align*}
\text{if } R_\theta &< 425 \text{ then } \alpha = (0.0168)(1.55) \quad (31a) \\
\text{if } R_\theta &> 6000 \text{ then } \alpha = 0.0168 \quad (31b) \\
\text{if } 425 &< R_\theta < 6000 \text{ then } \alpha = 0.0168 \left[ \frac{1.55}{1+\Pi} \right] \text{ where} \\
\Pi &= 0.55 \left[ 1 - e^{-0.243\gamma} - 0.298 \gamma \right] \text{ and } \gamma = \left( \frac{R_\theta}{425} \right)^{-1} \quad (31c)
\end{align*}
\]

**Transverse curvature.** In developing the axisymmetric boundary layer equations a radius term is introduced as shown in equations (2) and (3). If the assumption is made that the body radius is very large compared to the boundary layer thickness then the radii in equations (2) and (3) reduce to the local body radius \( r_0 \) and the effect of the transverse (i.e.,
circumferential) curvature in the momentum equation is neglected. If, however, the body radius is small compared to the boundary layer thickness then the effect of the transverse curvature cannot be ignored and \( r \) must be a function of the distance into the boundary layer, \( y \). The relationship between \( y \), \( r_0 \), and \( r \) is given by:

\[
r = r_0 + y \cos \alpha
\]  

(32)

As observed in figure 3, \( \alpha \) is simply the surface slope in the longitudinal direction. i.e.,

\[
\tan \alpha = \frac{\text{dr}}{\text{dx}}
\]  

(33)

For slender cylinders where \( \alpha = 0^\circ \),

\[
r = r_0 + y
\]  

(34)

The inclusion of the transverse curvature terms in the boundary layer equations is shown in References 24 and 27 to substantially improve the accuracy of the calculation of the local skin friction as well as the other viscous parameters.

**Transition region effect.** - The boundary layer method has the capability of calculating transition from laminar flow to turbulent flow in two different ways. The first approach is to use the transition point as a switching point between laminar and turbulent boundary layer calculations. At the transition point the turbulent boundary layer calculations are started by activating the eddy viscosity coefficient. In general, especially at low Reynolds numbers this approach can lead to errors as shown by Cebeci in Reference 28. The second approach which is available uses the intermittancy factor given by Chen and Thyson (Reference 29) to modify the eddy viscosity equations to account for a region of transition. This modification was developed from the point of view of intermittent production of turbulent spots and is a further extension of Emmons' spot theory (Reference 30). The modification to be used is to multiply the inner and outer eddy viscosities equations (16) and (27) by the following parameter:
The effect of this transition region correction can be seen in figure 4 which compares experimental data to theoretical calculations for local skin friction with and without the above correction on a two-dimensional ellipse. This transitional effect will be assumed to be the same for axisymmetric bodies.

**Boundary layer transition location.** - The location of boundary layer transition from laminar to turbulent flow can be either input to the boundary layer method or calculated internally within the program. The approach used to calculate the transition location is one developed for two-dimensional flow by Michel (Reference 31) and later verified by Smith (Reference 32). This method correlates the local momentum thickness Reynolds number, \( R_e \), and the local distance Reynolds number, \( R_x \), as shown in figure 5 which comes from Reference 32. The procedure used is to calculate the values of \( R_x \) and \( R_e \) at each station and to compare them to the curve in figure 5. If the value of \( R_e \) is less than the value of \( R_e_{TR} \) then transition has not been reached but if the value of \( R_e \) is greater than \( R_e_{TR} \) then transition has occurred.

The above method was extended to axisymmetric flow by the use of Mangler's transformation. The parameters \( R_e \) and \( R_x \) are calculated by the axisymmetric boundary layer routine and they are then transformed to two-dimensional values by the following relationships:

\[
\theta_{2-D} = \left( \frac{R_e}{L} \right) \theta_{AXISYMMETRIC} \tag{36a}
\]

\[
x_{2-D} = \int_{0}^{x_{LOCAL}} \left( \frac{R_e}{L} \right)^2 (dX)_{AXISYMMETRIC} \tag{36b}
\]
These values of $\theta_{2-D}$ and $X_{2-D}$ are used to determine values of $R_0$ and $R_x$ which can be used in conjunction with figure 5.

A study of transition location calculation for axisymmetric bodies was recently completed by Kaups (Reference 33). In this study empirical methods due to Granville, Hall, and Gibbons, and the method of Michel presented above were compared to the stability analysis technique of Smith (Reference 32). It was determined that for flows where transition occurred in an adverse pressure gradient all of the above techniques predicted transition fairly accurately. For flows where transition occurred in favorable pressure gradients, only the method of Smith (Reference 32) gave satisfactory results as shown in figure 6 which is taken from Reference 33. The method of Smith, however, requires extremely lengthy computer calculation times which makes it undesirable for the iterative type of calculation presented in this report. Therefore, based on the results of Reference 33, the method of predicting transition in the present program should not be used for flows with very large Reynolds numbers where the transition location might occur in a favorable gradient, but rather the transition point should be input to the program.

**Calculation Procedure**

The viscous flow field about an axisymmetric body is simulated by calculating the inviscid flow about an equivalent "viscous" body which is formed by adding the boundary layer displacement thickness to the original body surface. This technique of defining the inviscid body has been used quite successfully for two-dimensional flows as shown in Reference 2 and has also been used for axisymmetric flows as presented in Reference 34. This equivalent body is formed by combining the previously discussed geometry routine, potential flow method, and boundary layer method under control of the axisymmetric design and analysis method computer program known as ADAM.

Given the desired axisymmetric configuration and flow conditions, the ADAM program utilizes these sections, as shown in figure 7, in the following iterative manner:

1. Precise geometry definition for input into the potential flow program.
2. Calculation of the exact nonlinear potential flow for specified geometry and flow conditions.
3. Calculation of the viscous flow characteristics based on the results of the potential flow program.
4. Addition of boundary-layer displacement thickness to the basic geometry for each element.
5. Recalculation of the pressure distribution utilizing the potential flow program, based on the redefined geometry.
6. Recalculation of viscous flow field based on recalculated pressure distribution from redefined geometry, if desired.
7. Possible iteration of the above scheme; the degree to which this is required is presented in the subsequent discussion on correlations with experimental data.

The above technique must be modified when the boundary layer separates or when the local body radius approaches zero at the trailing edge of the body. When the dimension of the local body radius approaches zero at the trailing edge, the boundary layer equations become invalid since the \(1/r\) term in equation (3) approaches infinity. When this occurs, the boundary layer results are ignored from this point downstream to the trailing edge. The assumption is then made that the boundary layer displacement area at the point where

\[
\delta^* \cos \alpha = r_o \tag{37}
\]

is defined by

\[
DAREA = \pi \left( r_{op} + \delta^* \cos \alpha_p \right)^2 - \left( r_{op} \right)^2 \tag{38}
\]

where \(p\) refers to the point where equation (37) is first satisfied. This displacement area is then considered to remain constant from the point \(p\) to the trailing edge. The new "viscous" body coordinates in this region are then defined by

\[
y_{new} = \left( \frac{\pi r_o^2 + DAREA}{\pi} \right)^{\frac{1}{2}} \tag{39}
\]

The second problem area occurs when the boundary layer separates from
the body creating a separation bubble. This bubble must be accounted for in
the creation of a "viscous" body if the flow about this configuration is to
be predicted accurately. The simplest technique of modeling this separation
bubble is to assume that the flow leaves the surface parallel to the free-
stream direction, producing a cylindrical wake shape as shown in figure 8.
This approach, however, gives decelerations in the flow at the junction of the
body with the cylinder as shown in figure 9, which do not exist in the real
flow field. To minimize this problem, a circular arc is used to fair the
body into the separated cylinder.

This circular arc is defined by passing a circle through the last three
"viscous" body coordinates defined prior to the separation point. The radius
of this circle is then used to create a circular arc which is tangent to the
"viscous" body at the point of separation. The center of this arc is then
defined according to whether the surface slope of the body at separation is
positive or negative.

If the surface slope is positive then the center is taken as the center
of the circle passed through the three points as defined above. This center
is defined by

\[
\begin{align*}
    x_c &= x_{sep} + R \sin \left[ \tan^{-1} \left( \frac{dy}{dx} \right) \right] \\
    y_c &= y_{sep} - R \cos \left[ \tan^{-1} \left( \frac{dy}{dx} \right) \right]
\end{align*}
\]  

where \( R = \) Radius of the circle  
\[ \frac{dy}{dx} = \text{Surface slope at separation} \]  

This arc is then used from the point of separation to either the end
of the body or to the maximum point on the arc, where \( \frac{dy}{dx} = 0 \), as shown
in figure 10a. If the maximum point of the arc occurs before the trailing
edge of the body is reached then a cylinder is defined which extends from the
maximum point of the arc to the trailing edge.

If the surface slope is negative then the circular arc is defined such
that the center is located above the body. The center is then defined by
This arc is then used from the point of separation to either the end of the body or to the minimum point on the arc, where \( \frac{dy}{dx} = 0 \), as shown in figure 10b. If the minimum point of the arc occurs before the trailing edge of the body then a cylinder is defined which extends from the minimum point of the arc to the trailing edge of the body.

The above separated wake model has been derived from intuitive considerations rather than from first principals. It does, however, provide reasonable results, as will be shown in the subsequent discussion.

The base drag coefficient for blunt axisymmetric bodies is calculated using the method of Hoerner, reference 35. This approach is based on the assumption that the flow field behind a blunt base is basically a jet pump, in that, air flowing around the body leaves the trailing edge forming a cylindrical jet which attempts to pump away the stagnated air in the base region. However, since there is no air to replace this stagnated air, the pumping mechanism can only reduce the static pressure acting on the base. The effectiveness of this jet pump mechanism is controlled by the boundary layer thickness at the base since this region of lower momentum flow acts as a buffer between the stagnated air behind the base and the flow in the jet. Since the boundary layer thickness is directly related to the skin friction on the body, \( C_f \), Hoerner used \( C_f \) to correlate with the base drag to develop an empirical approach to determine base drag. Figure 11 shows the correlation obtained by Hoerner for bodies whose base area is the same as the maximum area. This curve is represented by

\[
C_{DBASE} = 0.029 / \sqrt{C_f_{forebody}}
\]

(42)

where the coefficients are based on the base area. Thus, once the skin friction on the forebody has been calculated in the boundary layer programs, then the base drag can be determined by equation 42.

\[
x_c = x_{sep} + R \left( \sin \tan^{-1} \frac{\partial y}{\partial x} \right)
\]

(41a)

\[
y_c = y_{sep} + R \left( \cos \tan^{-1} \frac{\partial y}{\partial x} \right)
\]

(41b)
This equation must be modified for boat-tailed bodies, that is, bodies whose base area is less than their maximum area. The mechanics of the base drag for these configurations do not change, but the calculation must take into account the reduced base area. This effect is taken into account by the following relationship:

\[ C_{D_{\text{BASE}}} = C_{D_{\text{BASE}}} \cdot \left( \frac{d_{\text{BASE}}}{D_{\text{MAX}}} \right)^2 \]  

(42a)

and

\[ C_{f_{B}} = C_{f_{B}} \cdot \left( \frac{D_{\text{MAX}}}{d_{\text{BASE}}} \right)^2 \]  

(BOAT TAIL)

so

\[ C_{D_{\text{BASE}}} = \frac{0.029}{C_{f_{B}}} \cdot \left( \frac{d_{\text{BASE}}}{D_{\text{MAX}}} \right)^3 \]  

(43c)

A comparison of results calculated by the above method in ADAM with experimental force data from reference 35 is presented in figure 12. One of these cases is for a boat-tailed body and the other for a body whose base area is also the maximum area.

The experimental data used for this comparison as well as the configuration used for the analytical calculations are both subject to some discussion. The experimental base drag, taken from Figure 4 of Reference 35, originally came from an old German report which is not readily available. These base drag values were obtained from both force measurements and pressure measurements which unfortunately do not agree. Therefore, since it was felt that the force measurements were the more accurate, they were used in the comparison shown in Figure 12. In addition, no good definition of the configuration tested was available, therefore, the geometry used in the ADAM analysis was taken from the schematics shown in Reference 35. In light of these uncertainties the comparison presented in Figure 12 is fairly good in that even though the levels are different, the trends are the same. It should be noted that this comparison was used only because there is a singular lack of experimental data for blunt based axisymmetric bodies at low subsonic Mach numbers.
Experimental results from three different configurations were selected to establish the extent of validity of the method presented in this report. These geometries consisted of a high fineness ratio body of revolution, and a sphere in both subcritical and supercritical flow regimes. These correlations, while limited to some extent by the scope of the present effort, do represent a wide range of axisymmetric flow conditions.

The body of revolution chosen was tested in the low speed wind tunnel at the Douglas Aircraft Company, (Reference 36), and is shown in figure 13. This model was composed of three sections; an elliptical nose section, a cylindrical control section, and a parabolic afterbody. The calculation done for this configuration used the wind tunnel flow properties, namely, $U_\infty = 71.628$ M/Sec (235 Ft/Sec), $T_\infty = 288.3^\circ K$ (519.0°R) and $R_L = 10.05 \times 10^6$. Boundary layer transition was fixed on the model and in the calculation at $0.03048$ meters (1.2 inches) from the nose. This model was relatively large for the wind tunnel in which it was tested; wall effects, not accounted for in the original data reduction, were present. To correct for this, the model was run in the potential flow program in the presence of the wind tunnel walls as shown in figure 14. The effect of including the walls in the calculation is shown in the inviscid pressure distributions of figure 15. The final results for this configuration are shown in figure 16 where the calculated "viscous" results are compared to experimental data. The inviscid distribution is also shown for reference. In this particular case no separation occurred and so only one iteration, that is, two potential flow solutions and two boundary layer solutions, was necessary. The calculated "viscous" results agree very well with the experimental values except in the region of the nose. This discrepancy is not due to the calculation method, but rather is due to the model being too long for the wind tunnel test section resulting in the nose being in a different static pressure field than the rest of the body. The overall effect of viscosity on this configuration is seen to be small except in the region of the trailing edge. The body is so slender in this region that the boundary layer equations are no longer valid so the technique described in the calculation procedure was used to modify the viscous body. The results show a pressure oscillation
in this modified region which is due to an unsmooth curvature distribution. However, the level of these pressures agree quite well with the experimental values.

The second case considered was that of a sphere in the supercritical flow regime, i.e., \( R_D = 1 \times 10^6 \). Since the boundary layer transition was forced to occur at an \( X/D = 0.65 \), there were regions of both laminar and turbulent flow present. The experimental data for this case were taken from references 37 and 38. The freestream velocity assumed for this case was \( 47.85 \, \text{M/Sec} \) (157 \, \text{Ft/Sec}). Figure 17 shows the sphere with the "viscous" body superimposed and figure 18 presents a comparison between the calculated "viscous" solution and experimental data. Note that while the calculated pressure distribution is in reasonably good agreement with the experimental values, the calculated separation point is 0.07 diameters further downstream than the experimentally measured value. The inviscid and "viscous" solutions for the local skin friction coefficient, \( C_f \), are presented in figure 19. The "viscous" solution shown is the fourth iteration, i.e., the fifth potential flow solution, and appears to be the best solution possible for this configuration with the technique being used in the present method to simulate flow separation.

The last correlation to be presented is for the flow about a sphere in the subcritical regime, i.e., \( R_D = 1 \times 10^5 \), which is a purely laminar case. The experimental data is again taken from Reference 37. The freestream velocity for this case was assumed to be \( 4.785 \, \text{M/Sec} \) (15.7 \, \text{Ft/Sec}). The calculated "viscous" body is shown in figure 20 while a comparison of the "viscous" pressure distribution to experimental data is shown in figure 21. The calculated "viscous" pressures are in close agreement with the experimental values with some slight over-prediction in the separated region. The calculated separation point is only 0.03 diameters further downstream than the experimental value which is excellent considering the large effect that viscosity has on this configuration. Figure 22 presents the inviscid and "viscous" solutions for the local skin friction coefficient for this case.
A method has been presented for the computation of the viscous flow field about axisymmetric bodies at zero angle of attack in incompressible flow. This computing program requires only the specified body geometry and desired flow conditions as input. The appropriate theory has been discussed and correlations between theoretical and experimental results presented.

The flow field about axisymmetric bodies at zero angle of attack with no flow separation is well defined and can be computed accurately by the present method. When flow separation occurs, the flow field is no longer amenable to analytical treatment. Currently, methods do not exist to calculate the flow field within a separated region; it is therefore necessary to resort to empirical methods to account for flow separation. Since there is almost a complete lack of experimental data concerning the behavior of separated regions, any empirical methods must necessarily be somewhat crude. The most sophisticated model for separation currently available is due to Jacob (References 39 and 40) and is strictly for two-dimensional airfoils. An unsuccessful attempt was made in Reference 41 to adapt Jacob's approach to axisymmetric configurations. The conclusions of Reference 41 indicated that the assumed boundary conditions needed to be modified if this approach was to be used for axisymmetric flow. It is proposed that the Douglas-Neumann program be used to pursue this approach at modeling separation. This potential flow program is ideal for attempting to use Jacob's technique since it already has the ability to specify a non-uniform flow distribution over all or part of a configuration; therefore, only suitable boundary conditions would have to be added to the program. It is felt that this approach can be successful in modeling separation if care is taken in developing the distribution of non-uniform velocity as well as specifying the proper boundary considerations.

The further extension of this model to the calculation of flow about axisymmetric bodies at angle of attack is also possible. The potential flow routine contained in the present method has the capability of predicting the flow field about non-lifting bodies at angle of attack by combining the streamflow and the crossflow solutions. The boundary layer analysis would require the replacement of the routine in the present method by a three-dimensional
technique, which is currently not available. However, it is felt that a good approximation to the boundary layer calculations can be made by the small crossflow program of Reference 42.

One area of primary concern in extending the method to include an angle of attack capability is the determination of the separation line about the body. The present method of predicting separation for two-dimensional bodies and for axisymmetric bodies at zero angle of attack is to find the location where the skin friction goes to zero. It has been shown in several studies, including those reported in References 43 and 44, that this condition does not apply in three-dimensional flows because the skin friction along a separation line is not necessarily zero. Therefore, some method of determining the separation line for axisymmetric bodies at angle of attack must be developed. It is proposed that the present method could be extended to calculate the "viscous" flow about axisymmetric bodies at angle of attack when no flow separation is present. This method could then be used to assist in the development of a procedure for determining the separation line location. Once the location of the separation line is known then a model could be developed for analyzing the viscous flow about the separated body. The development of such procedures is not a simple task and considerable effort would have to be expended; but the reward for accomplishing this task is an advance in the ability to calculate the real flow about arbitrary three-dimensional bodies which is our ultimate goal.
This part of the report contains the necessary information to input data to the ADAM computer program. The input data is broken into three sections: smoothing, potential flow, and viscous flow. These sections can be used together in the iterative fashion described in the main text, or the potential flow and viscous flow sections may be used independently. A detailed card-by-card description of all input quantities is given followed by a set of input forms which can be used to facilitate the loading of the input data into the program.
Input Instructions

The Adam program requires one system control card followed by the required sets of data cards for each program option to be executed. The sets of data furnished must be in the same order as the options are specified on the system control card. If an iteration is desired the system control card is repeated along with the necessary other data cards.

The general scheme used in describing the input data is shown below:

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column — Column indicates the starting position on the card for each data field.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code — The &quot;code&quot; gives the FORTRAN name used in the read statement by the program.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Routine — &quot;Routine&quot; indicates the subroutine where the data is read.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format — The parameter &quot;FORMAT&quot; which is given right under the routine name, indicates the FORTRAN format of the data read statement field. The parameter I5 would indicate that the parameter is an integer in a field that is 5 columns wide. Integers should be punched on the right side of the field (right justified). The parameter F10.0 would indicate a fixed point number punched with a decimal point (i.e., -12.354). The number may be punched anywhere in the field indicated irrespective of the decimal point location indicated by the format. The parameter E12.6 would indicate a floating point number punched with a decimal point (i.e., 5.0 x 10^6). The number must be punched to the right of the field in the manner 5.0E+06.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Explanation — The description of the input data is given under "explanation".
### SUSTEN CONTROL DATA CARD

(This card must be the first card in the data deck)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>IGEOM</td>
<td>MAIN I1</td>
<td>Smoothing option flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  no smoothing is desired</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  smoothing is desired</td>
</tr>
<tr>
<td>8</td>
<td>INEUM</td>
<td>MAIN I1</td>
<td>Potential flow option flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  No potential flow solution is desired</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  Potential flow solution is desired</td>
</tr>
<tr>
<td>12</td>
<td>IBOUND</td>
<td>MAIN I1</td>
<td>Boundary layer option flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  No boundary layer solution is desired</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  Boundary layer solution is desired</td>
</tr>
<tr>
<td>16</td>
<td>ITER</td>
<td>MAIN I1</td>
<td>&quot;Viscous&quot; body formation flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  No &quot;viscous&quot; body is formed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  &quot;Viscous&quot; body is formed</td>
</tr>
<tr>
<td>17-20</td>
<td>IFINSH</td>
<td>MAIN I4</td>
<td>Termination Flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  Another case expected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=9999 Program will stop after exercising all options specified above</td>
</tr>
</tbody>
</table>

28
SMOOTHING SECTION

These cards required if IGEOM = 1 on system control card. This section is used to smooth body geometry data before it is input to the potential flow program.

### Smoothing Control Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>NPTS</td>
<td>SMOOTH I3</td>
<td>Number of input data points for this configuration. NPTS must be ≤ 100</td>
</tr>
<tr>
<td>8</td>
<td>ITAPE</td>
<td>SMOOTH I1</td>
<td>Data source flag ＜0: Data input on unit 5 (card input) ＜0: Data input on unit 1. This is used for a case where a &quot;viscous&quot; body generated by the iteration procedure is being read.</td>
</tr>
</tbody>
</table>

#### Geometry Data Input Cards

These cards are input only if ITAPE = 0.

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-coordinate cards</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-10</td>
<td>x(1)</td>
<td>SMOOTH 6F10.0</td>
<td>x-coordinates starting at the leading edge and proceeding along the upper surface to the trailing edge. Input 6 x-values on each card. The numbers of x-values must be equal to NPTS.</td>
</tr>
<tr>
<td>11-20</td>
<td>x(2)</td>
<td>SMOOTH 6F10.0</td>
<td></td>
</tr>
<tr>
<td>21-30</td>
<td>x(3)</td>
<td>SMOOTH 6F10.0</td>
<td></td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| y-coordinate cards | | | |
| 1-10   | y(1) | SMOOTH 6F10.0  | y-coordinates to correspond to the above x-locations. y-values must be positive. Input 6 values per card. |
| 11-20  | y(2) | SMOOTH 6F10.0  | |
| 21-30  | y(3) | SMOOTH 6F10.0  | |
| etc.   |      |                | |
POTENTIAL FLOW SECTION

These cards required if INEUM = 1 on system control card. The input geometry for this program may be obtained from the geometry storage unit (10) as generated by the smoothing section, or it may be input directly on unit 5. Thus, this program may be operated as a separate entry if so desired. The program saves the geometry data element midpoints with the corresponding pressure coefficients on unit 3 for input to the boundary layer routine and it saves the basic non-dimensional input Neumann coordinates on unit 1 for use if a 'visous' body is desired.

Title Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HEDR</td>
<td>PART1 10A6</td>
<td>Title of case. May be any characters input in the first 60 columns of card;</td>
</tr>
<tr>
<td>63</td>
<td>CASE</td>
<td>PART1 I6</td>
<td>Case number</td>
</tr>
<tr>
<td>77</td>
<td>PSF</td>
<td>PART1 I6</td>
<td>Additional identifier for this case.</td>
</tr>
</tbody>
</table>

Flag Card

Card columns 1-30 when punched with any non-zero integer, activate flags that indicate the following:

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NB</td>
<td>PART1 I1</td>
<td>The number of bodies input. Normally set equal to 1. 1 ≤ NB ≤ 5</td>
</tr>
<tr>
<td>2</td>
<td>NNU</td>
<td>PART1 I1</td>
<td>The number of non-uniform onset flows. Normally set equal to 0.</td>
</tr>
<tr>
<td>3</td>
<td>FLGO3</td>
<td>PART1 I1</td>
<td>Axisymmetric flow flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 No axisymmetric stream-flow solution calculated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 Axisymmetric streamflow solution is calculated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Normally set equal to 1</td>
</tr>
<tr>
<td>Column</td>
<td>Code</td>
<td>Routine Format</td>
<td>Explanation</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>4</td>
<td>FLG04</td>
<td>PART1 I1</td>
<td>Cross flow flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  No cross flow solution is calculated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  Cross flow solution is calculated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Normally set equal to 0</td>
</tr>
<tr>
<td>5</td>
<td>FLG05</td>
<td>PART1 I1</td>
<td>Off-body point flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  No off body points input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  Off body points are input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This flag allows the velocity at points off the body surface to be determined.</td>
</tr>
<tr>
<td>6</td>
<td>FLG06</td>
<td>PART1 I1</td>
<td>Basic data formation flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  A full case will be done</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  The basic data, i.e., midpoints, normals, etc. will be formed and printed. No velocities will be calculated.</td>
</tr>
<tr>
<td>7</td>
<td>FLG07</td>
<td>PART1 I1</td>
<td>Ellipse generator option</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  Body coordinates will be input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  An ellipse is generated using data input later. No body coordinates are input</td>
</tr>
<tr>
<td>8</td>
<td>FLG08</td>
<td>PART1 I1</td>
<td>Matrix print flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  Coefficient matrices are not printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  Coefficient matrices will be printed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Normally set equal to 0</td>
</tr>
<tr>
<td>11</td>
<td>FLG11</td>
<td>PART1 I1</td>
<td>Perturbation velocity flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  Normal case</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  No onset flow used. Only perturbation velocities are calculated.</td>
</tr>
<tr>
<td>12</td>
<td>FLG12</td>
<td>PART1 I1</td>
<td>Potential matrix solution *</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0  Normal case</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1  A potential matrix is solved</td>
</tr>
</tbody>
</table>
Flag Card (Continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| 13     | FLG13 | PART1 I1      | Matrix solution flag  
|        |       |               | =0  No matrix solution done  
|        |       |               | =1  Matrix solution performed  
|        |       |               | Normally set equal to 1. |
| 14     | FLG14 | PART1 I1      | Prescribed tangential velocity flag *  
|        |       |               | =0  Normal case  
|        |       |               | =1  Tangential velocities are specified |
| 15     | FLG15 | PART1 I1      | Strip ring vorticity flag  
|        |       |               | =0  Normal case  
|        |       |               | =1  A vorticity distribution is formulated. |
| 16     | FLG16 | PART1 I1      | Axisymmetric uniform flow flag  
|        |       |               | =0  Normal case  
|        |       |               | =1  Axisymmetric uniform flow solution is omitted  
|        |       |               | Normally set equal to 0. |
| 17     | FLG17 | PART1 I1      | Crossflow uniform flow flag  
|        |       |               | =0  Normal case  
|        |       |               | =1  Crossflow uniform flow solution is omitted. |
|        |       |               | Since FLG04 is normally = 0 then so is FLG17 normally set equal to 0. |
| 18     | FLG18 | PART1 I1      | Surface vorticity flag *  
|        |       |               | =0  Normal case  
|        |       |               | =1  Surface vorticity is generated. |
| 19     | FLG19 | PART1 I1      | Prescribed vorticity Flag *  
|        |       |               | =0  Normal case  
|        |       |               | =1  A prescribed vorticity is input |
| 20     | FLG20 | PART1 I1      | Total vorticity flag *  
|        |       |               | =0  Normal case  
<p>|        |       |               | =1  Total vorticity calculated |</p>
<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
</table>
| 21     | FLG21 | PART1 II       | Extra crossflow flag *
|        |       |                | =0 Normal case |
|        |       |                | =1 Extra crossflow option used |
| 22     | FLG22 | PART1 II       | Generated boundary condition flag *
|        |       |                | =0 Normal case |
|        |       |                | =1 Boundary conditions generated |
| 23     | FLG23 | PART1 II       | Ring wing option flag *
|        |       |                | =0 Normal case |
|        |       |                | =1 Ring wing option used |
| 28-29  | NIN   | PART1 I2       | Tape input flag |
|        |       |                | =0, 10 Data input on unit 10 from smoothing program |
|        |       |                | =5 Data input from unit 5 (card input) |
| 30     | ITER  | PART1 II       | Iteration tape flag |
|        |       |                | =0 x/c, y/c transformed data saved on unit 15 |
|        |       |                | =1 x/c, y/c transformed data not saved. |

This flag is necessary because for a "viscous" body to be formed, the coordinates of the original unmodified body must be saved. Therefore, for the first case set ITER = 0. For subsequent iterations we do not want to use the modified bodies to form new bodies so set ITER = 1.

* These flags are for special options which are discussed in the main text. They are never used for a normal axisymmetric calculation. Therefore, set them equal to zero.
### Chord Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CHORD</td>
<td>PART1</td>
<td>F10.0</td>
<td>Reference chord length used to non-dimensionalize x and y coordinates</td>
</tr>
<tr>
<td>11</td>
<td>MN</td>
<td>PART1</td>
<td>F10.0</td>
<td>Mach number (MN &lt; 1.0) used to approximate effect of compressibility (Gothert's rule)</td>
</tr>
<tr>
<td>21</td>
<td>TCNST</td>
<td>PART1</td>
<td>F10.0</td>
<td>This is a constant which is used for the value of the tangential velocity if this option is desired.</td>
</tr>
</tbody>
</table>

### Body Transformation Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>NN</td>
<td>BASIC1</td>
<td>I3</td>
<td>The number of input points on this body. NN ≤ 100</td>
</tr>
<tr>
<td>11</td>
<td>MX</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>A factor used to multiply all x-coordinates. MX is assumed equal to 1 if no value is input.</td>
</tr>
<tr>
<td>21</td>
<td>MY</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>A factor used to multiply all y-coordinates. MY is assumed equal to 1 if no value is input.</td>
</tr>
<tr>
<td>31</td>
<td>THETA</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>An angle (in degrees) through which all points of a body are to be rotated about the origin in the clockwise direction.</td>
</tr>
<tr>
<td>41</td>
<td>ADDX</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>A constant to be added to all x-coordinates</td>
</tr>
<tr>
<td>51</td>
<td>ADDY</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>A constant to be added to all y-coordinates</td>
</tr>
</tbody>
</table>
Body Control Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>BDN</td>
<td>BASIC1</td>
<td>II</td>
<td>Body sequence number. This program will handle up to 5 bodies.</td>
</tr>
<tr>
<td>20</td>
<td>SUBKS</td>
<td>BASIC1</td>
<td>II</td>
<td>Subcase Flag.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>=0 Normal case</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>=1 Use unmodified coordinates of the previous case.</td>
</tr>
<tr>
<td>30</td>
<td>NLF</td>
<td>BASIC1</td>
<td>II</td>
<td>Non-lifting flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>=0 Body is non-lifting (normal case)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>=1 Body is lifting (this is used in special option)</td>
</tr>
<tr>
<td>31</td>
<td>XE</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>Value of major semi-axis for use by ellipse generation option.</td>
</tr>
<tr>
<td>41</td>
<td>YE</td>
<td>BASIC1</td>
<td>F10.0</td>
<td>Value of minor semi-axis for use by ellipse generation option</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Note: if XE = YE a sphere will be formed.</td>
</tr>
</tbody>
</table>

Geometry Data Cards

The body geometry data cards are included only if the input parameters NIN = 5 and FLG07 = 0 on the flag card. If NIN = 0 or 10 then the data is read from unit 10. If NIN = 5 and BDN = 0, then the following cards contain the x-y coordinates of off-body points instead of x-y geometry data. The number of either geometry data point or off-body points must be equal to NN.

x-Coordinate cards (six values per card)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TX1(1)</td>
<td>BASIC1</td>
<td>6F10.0</td>
<td>x-coordinates of body input from leading to trailing edge.</td>
</tr>
<tr>
<td>11</td>
<td>TX1(2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>TX1(3)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### y-Coordinate cards (six values per card)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TY1(1)</td>
<td>BASIC1</td>
<td>y-Coordinates of body which correspond to the x-values above. y values must be positive.</td>
</tr>
<tr>
<td>11</td>
<td>TY1(2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>TY1(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>etc.</td>
</tr>
</tbody>
</table>

**NOTE:** Each body input, including the off body points, requires the body transformation card, the body control card, and may also require the geometry data cards depending on the input flags. This is the stopping place for a normal axisymmetric case. The following cards are input only if one of the special options is required.

### Tangential Velocity Data (six values per card)

These cards are input only if FLG14 ≠ 0 and TCNST = 0.0

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TG(1)</td>
<td>BASIC1</td>
<td>Specified tangential velocities at element midpoints.</td>
</tr>
<tr>
<td>11</td>
<td>TG(2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>TG(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>etc.</td>
</tr>
</tbody>
</table>

### Non-uniform Flow Cards (six values per card)

These cards are input only if NNU ≠ 0.

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>NUM</td>
<td>BASIC2</td>
<td>Non-uniform flow identification number.</td>
</tr>
</tbody>
</table>
Non-uniform Flow Cards (Continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>MSF</td>
<td>BASIC2</td>
<td>If MSF = 0 the flow velocities $N_0, T_0$ will be used for the axisymmetric case only. If MSF = 1 the flow velocities $N_0, T_0$ will be used for the cross flow case only. If MSF &gt; 1 the flow velocities will be used for both axisymmetric and cross flow cases.</td>
</tr>
<tr>
<td>21</td>
<td>TYPE</td>
<td>BASIC2</td>
<td>Flag which specifies the type of input flow velocities at each mid-point. If TYPE &gt; 0.0, the velocities are input as $x$ &amp; $y$ components. If TYPE = 0.0 the velocities are input as normal &amp; tangential components. If TYPE &lt; 0.0 the automatic generation of the flow due to a rotating body is used.</td>
</tr>
<tr>
<td>31</td>
<td>FG</td>
<td>BASIC2</td>
<td>Constant used by the flow generator. Type must be less than 0.0.</td>
</tr>
</tbody>
</table>

The following cards are input only if NNU ≠ 0 and TYPE ≠ -1.0.

**Normal velocity cards** (six values per card)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NO(1)</td>
<td>BASIC2</td>
<td>This is either the $x$ or normal velocity component depending on the value of type above. These values must be in sequence with the coordinate data. If the $x$ component is input it is defined as positive to the right. If the normal velocity is input it is positive if it is to the interior of the body. NN-1 values are input.</td>
</tr>
</tbody>
</table>
## Tangential Velocity Cards (six values per card)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine</th>
<th>Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>T0(1)</td>
<td>BASIC2</td>
<td>6F10.0</td>
<td>This is either the y or tangential velocity component depending on the value of type above. These values must correspond to the NO values above. If the y component is input it is defined as positive if it is orientated upwards. If the tangential velocity is input it is positive if the flow field is to the left of the vector representing the tangential velocity.</td>
</tr>
</tbody>
</table>
VISCOUS FLOW SECTION

These cards required if IBOUND = 1.

BOUNDARY LAYER PROGRAM

The geometry and pressure distribution data required by this program may be input directly on cards (Unit 5), or read from the data save unit (Unit 3) as generated by the Neumann program.

HEADER CARD

This card is supplied purely for description purposes.

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-60</td>
<td>TITLE</td>
<td>INPT 15A4</td>
<td>Description of input</td>
</tr>
<tr>
<td>61</td>
<td>CASE</td>
<td>INPT A4</td>
<td>Case number</td>
</tr>
</tbody>
</table>

Flag Control Card

This card contains flags which control the type of flow to be considered and the form of the input.

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NXT</td>
<td>INPT I4</td>
<td>The number of the x-station where the flow goes turbulent measured from the stagnation point (i.e., the leading edge for axisymmetric bodies at zero angle of attack) if transition is to be calculated by the program set NXT to be one greater than the number of points input. Transition flag</td>
</tr>
<tr>
<td>5</td>
<td>LG16</td>
<td>INPT I1</td>
<td>=0 Boundary layer transition point is input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 Boundary layer transition point is computed. Set NXT to be greater than number of points input.</td>
</tr>
</tbody>
</table>
### Flag Control Card (Continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>LG17</td>
<td>INPT I1</td>
<td>Transition control flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 Transition is instantaneous</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 Transition is gradual (transitional region used)</td>
</tr>
<tr>
<td>7</td>
<td>LG18</td>
<td>INPT I1</td>
<td>Transverse curvature flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 No transverse curvature correction used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 Transverse curvature corrections applied.</td>
</tr>
<tr>
<td>8</td>
<td>LG32</td>
<td>INPT I1</td>
<td>Print control flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 Print using long format (with velocity profiles)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 Use short printout (no velocity profiles)</td>
</tr>
<tr>
<td>9</td>
<td>LG26</td>
<td>INPT I1</td>
<td>Velocity input control flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=2 Velocity ratio ( \frac{U_e}{U_o} ) is input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=3 Pressure coefficient ( c_p ) is input.</td>
</tr>
<tr>
<td>10</td>
<td>LG40</td>
<td>INPT I1</td>
<td>Unit input flag for geometry and velocity data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 Data read from unit 3 as generated by the potential flow program.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>≠0 Data read from cards (unit 5)</td>
</tr>
<tr>
<td>11</td>
<td>LG41</td>
<td>INPT I1</td>
<td>System of units FLAG</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0 English system of units</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1 International system of units</td>
</tr>
</tbody>
</table>

### Flow Condition Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TI</td>
<td>INPT F10.0</td>
<td>Reference static temperature used to compute the reference fluid properties. If TI is input as zero then TI is set equal to either 288.33°K or 519°R depending on FLAG LG41</td>
</tr>
</tbody>
</table>
### Flow Condition Card (Continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>RMI</td>
<td>INPT F10.0</td>
<td>Reference or free-stream Mach number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0.0 $\mathbf{M}_\infty$ is input next</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>≠0.0 $\mathbf{M}_\infty$ input as zero above.</td>
</tr>
<tr>
<td>21</td>
<td>UI</td>
<td>INPT F10.0</td>
<td>Reference or free-stream velocity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0.0 $\mathbf{M}_\infty$ is input above</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>≠0.0 $\mathbf{M}_\infty$ input as zero above.</td>
</tr>
<tr>
<td>31</td>
<td>FK</td>
<td>INPT F10.0</td>
<td>Flow index</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=0.0 2-D flow assumed</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>=1.0 Axisymmetric flow assumed</td>
</tr>
<tr>
<td>41</td>
<td>RL</td>
<td>INPT F10.0</td>
<td>Chord or reference length</td>
</tr>
<tr>
<td>51</td>
<td>RI</td>
<td>INPT E12.0</td>
<td>Reynolds number/foot</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\frac{R}{\ell} = \frac{U_\infty}{v}$</td>
</tr>
</tbody>
</table>

If $\text{CHORD} = 1.0$ then $\text{RI}$ must be Reynolds number based on $\text{CHORD}$.

NOTE: The input of either Mach number or freestream velocity is for convenience only. This program is entirely incompressible.

### Radius Card

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROMAX</td>
<td>INPT F10.0</td>
<td>Maximum radius of body. This is used to obtain frontal area for skin friction calculation.</td>
</tr>
<tr>
<td>11</td>
<td>DETAl</td>
<td>INPT F10.0</td>
<td>Initial step size of boundary layer velocity profile grid. For a case which contains turbulent flow set DETAl = .005.</td>
</tr>
</tbody>
</table>
Radius card (continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>VGP</td>
<td>INPUT F10.0</td>
<td>VGP is the growth factor for the boundary layer velocity profile grid; for cases with turbulent flow set equal to 1.14.</td>
</tr>
</tbody>
</table>

**NOTE:** For laminar cases the boundary layer velocity profile grid may be made constant if VGP = 1.0 is input. However, if this is done the minimum value of DETAI that can be input is approximately .10. This can be calculated if the value of the transformed boundary layer thickness, ETAINF, is known. Then DETAI becomes

\[
\text{DETAI} = \frac{\text{ETAINF}}{100}
\]

Geometry-Pressure Distribution Cards

These cards input only if LG40 ≠ 0.

**Point Number Card**

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NXM</td>
<td>INPT I4</td>
<td>Number of data points to be input. Maximum of 100 points allowed.</td>
</tr>
</tbody>
</table>

**x-Coordinate Data Cards**

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>XS(1)</td>
<td>INPT 6F10.0</td>
<td>x-coordinate points input from leading to trailing edge input 6 points per card. Number of points = NXM</td>
</tr>
<tr>
<td>11</td>
<td>XS(2)</td>
<td>6F10.0</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>XS(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**y-Coordinate Data Cards**

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>YS(1)</td>
<td>INPT 6F10.0</td>
<td>y-coordinate points corresponding to x-coordinates above input 6 points per card.</td>
</tr>
</tbody>
</table>
### y-Coordinate Data Cards (continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>YS(2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>YS(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Pressure Distribution Cards

<table>
<thead>
<tr>
<th>Column</th>
<th>Code</th>
<th>Routine Format</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UE(1)</td>
<td>TNPT 6F10.0</td>
<td>Velocity-pressure-distribution points corresponding to x-points input above input 6 points per card.</td>
</tr>
<tr>
<td>11</td>
<td>UE(2)</td>
<td></td>
<td>If LG26 = 2, ( \frac{u_e}{U_\infty} ) input</td>
</tr>
<tr>
<td>21</td>
<td>UE(3)</td>
<td></td>
<td>LG26 = 3, ( c_p ) input</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ADAM

AXISYMMETRIC DESIGN AND ANALYSIS METHOD

IGEOM INEUM IBOUND ITER IFINSH

cc 4 8 12 16 20

SYSTEM CONTROL DATA CARD

cc 4 IGEOM = 0 NO SMOOTHING REQUIRED = 1 FIVE-POINT SMOOTHING USED
cc 8 INEUM = 0 NO POTENTIAL FLOW = 1 NEUMANN ROUTINE USED
cc 12 IBOUND = 0 NO BOUNDARY LAYER SOLUTION DESIRED = 1 BOUNDARY LAYER SOLUTION WILL BE DONE
cc 16 ITER = 0 NO "VISCOUS" SOLUTION IS DESIRED = 1 A "VISCOUS" BODY WILL BE CREATED
cc 17 IFINSH =0000 ANOTHER CASE IS EXPECTED = 9999 THIS IS THE LAST CASE

Instructions to Keypunch:
Do not punch blank columns

ENGINEER ________________ PHONE __________

DATE ________________ PAGE ___ OF ___
SMOOTHING PROGRAM

SMOOTHING CONTROL CARD

<table>
<thead>
<tr>
<th>NPTS</th>
<th>ITAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>cc</td>
<td>4</td>
</tr>
</tbody>
</table>

Instructions to Keypunch
Do not punch blank columns

ENGINEER________________________ PHONE________________

DATE________________________ PAGE____ OF____
ADAM
BOUNDARY LAYER PROGRAM

FLAG CONTROL CARD

RADIUS CARD

Instructions to Keypunch
Do not punch blank columns

ENGINEER
PHONE

DATE
PAGE OF
APPENDIX B

CONTROL INFORMATION FOR ADAM COMPUTER PROGRAM

This part of the report contains the necessary control information to operate this program on a computer system. This section contains the overlay structure as well as flow charts of the main subroutines including input flow information. Also, the various data sets used between main programs are described.
MAIN PROGRAM

START

READ IGEOM, INEUM, IBOUND, ITER, IFINSH FROM UNIT 5

IF IGEOM = 1
   CALL SMOOTH
   IF INEUM = 1
      CALL NEUMAN
      IF IBOUND = 1
         CALL BOUNDL
      END IF
   END IF
END IF

IF ITER = 1
   CALL ITERAT
   IF IFINSH = 9999
      STOP
   END IF
END IF

STOP
FUNCTIONAL ORGANIZATION OF NEUMANN PROGRAM

- NEUMANN
  - PART1
    - BASIC1
    - BASIC2
    - MATRIX
      - XYZ
        - XYZ1
        - XYZ2
  - PREP
    - SOLVIT
  - PART4
    - AXIS
    - CROSS
    - EXCROSS
      - COMBO
        - NOTS
        - SOLCOM
          - QC
          - HLAMB
            - ELIP
            - ELLC
            - INEL
              - ELINT3
START

REWIND UNITS 3, 4, 8, 9, 10, 11, 12, 13, 15

CALL PART1

IF FLG06 = 0

CALL PREP

CALL PART4

RETURN
BASIC FLOW CHART FOR SUBROUTINE PART1

START

READ FLAG CARD FROM UNIT 5

READ CHORD CARD FROM UNIT 5

CALL BASIC1

IF NNU = 0
    CALL BASIC2
    CALL MATRIX

RETURN
BASIC FLOW CHART FOR SUBROUTINE BASIC1

START

L=1

READ BODY TRANSFORMATION CARD FROM UNIT 5

READ BODY CONTROL CARD FROM UNIT 5

IF SUBKS ≠ 0

IF FLG07 = 0

READ BODY COORDINATES FROM UNIT 13 (SAVED FROM PREVIOUS CASE)

GENERATE ELLIPSE FROM XE AND YE INPUT DATA

READ BODY COORDINATES FROM UNIT NIN

GOTO A
BASIC FLOW CHART FOR SUBROUTINE BASIC1 (CONTINUED)

A

IF MX > 0
  TX1 = MX * TX1

IF MY > 0
  TY1 = MY * TY1

IF THETA > 0
  ROTATE COORDINATES ABOUT (0,0) BY THETA

IF ADDX > 0
  TX1 = TX1 + ADDX

IF ADDY > 0
  TY1 = TY1 + ADDY

IF CHORD > 0
  TX1 = TX1 / CHORD
  TY1 = TY1 / CHORD

IF MN > 0
  TX1 = TX1 / (1-MN^2) * 5

GOTO B
BASIC FLOW CHART FOR SUBROUTINE BASIC1 (CONTINUED)

B

L = L+1

GOTO C

IF L > NB

YES

IF FLG14 ≤ 0

IF TCNST > 0

READ TG(I) FROM UNIT 5

RETURN
BASIC FLOW CHART FOR SUBROUTINE BASIC2

START

L = 1

READ NON-UNIFORM FLOW CARD FROM UNIT 5

IF TYPE = -1

READ NO AND TO FROM UNIT 5

L = L + 1

IF L ≤ NNU

NO

RETURN

YES

CALCULATE NO AND TO USING ROTATING FLOW
START

NX = 0
IGNP = 0
IGTR = 0
NTC = NTC+1
IT = 0

CALL INPT

NX = NX + 1

IGRC = 0

IGOL = 0
IGOT = 0

IF
NX < NXT IGOL = 1
NX ≥ NXT IGOT = 1

IGCV = 0

GOTO A
CALL EINF

IF NX = 1

YES

CALL IVPF
CALL FLPR

NO

IF NX = 1 or XS(NX) = 0

NO

CALL EDVS

YES

IF NX = 1

NO

YES

CALL SHFT
CALL FLPR

IT = 0
LC = LC + 2

145

IF NX = NXT

YES

NO

ITC = 1

150

IT = IT + 1

GOTO B
BASIC FLOW CHART FOR SUBROUTINE BOUNDL (CONTINUED)

E

IF EAG < .02

YES GOTO C

NO

IF IGTR ≤ 1 OR v(1,2) ≤ 0

YES GOTO 150

NO

IGCV=1

CALL EINF

IF NX=1 OR IGCR=0

YES GOTO 150

NO

CALL SHFT
CALL FLPR

GOTO 145
BASIC FLOW CHART FOR SUBROUTINE BOUNDL (CONTINUED)

C

600
LC = LC + 1

IF
IGTR > 1
IGTR = 0

IGCV = 1
CALL EINF

IF
IGRC = 0

YES

CALL SHFT
CALL FLPR

GOTO 145

NO

CALL OTPT

IF
NX = NSM

YES

GOTO 700

NO

IF
IGOT = 1

YES

GOTO 100

NO

GOTO D

GOTO
BASIC FLOW CHART FOR SUBROUTINE BOUNDL (CONTINUED)

D

IF
NX > 1 &
LG16 = 0

CALL TRNS

IF ICTR > 1
IGRO = 1

IF IGTR > 1
GOTO 120

GOTO 100

64
BASIC FLOW CHART FOR SUBROUTINE BOUNDL (CONTINUED)

700

IF

NX=0

YES

NO

IOUT = 1
CALL OUPT

RETURN
BASIC FLOW CHART FOR SUBROUTINE INPT

START

READ TITLE CARD FROM UNIT 5

READ CONTROL CARD FROM UNIT 5

READ FLOW CONDITION CARD FROM UNIT 5

READ RADIUS CARD FROM UNIT 5

IF LG40 = 0

READ COORDINATE DATA FROM UNIT 5

CALCULATE SURFACE DISTANCE, FLUID PROPERTIES, AND INITIALIZE PERTINANT DATA

RETURN

READ COORDINATE DATA FROM UNIT 3
The following is a description of the output symbols from the various sections of the ADAM computer program:

**Neumann Potential Flow Subprogram**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDED MASS</td>
<td>$\Sigma 2\pi \cdot \Phi_i \cdot V_n \cdot ds$</td>
</tr>
<tr>
<td>AJK</td>
<td>Influence coefficients $W_{jk}$ resolved parallel to the outward normal of the element. Output only if FLG08 = 1.</td>
</tr>
<tr>
<td>ADDX</td>
<td>Constant which is added to all X-coordinates of a particular body. Value printed out for each body.</td>
</tr>
<tr>
<td>ADDY</td>
<td>Constant which is added to all Y-coordinates of a particular body. Value printed out for each body.</td>
</tr>
<tr>
<td>BJK</td>
<td>Influence coefficient $W_{jk}$ resolved parallel to the tangent direction of the element.</td>
</tr>
<tr>
<td>BODIES</td>
<td>Number of bodies in system, same as NB input on flag card.</td>
</tr>
<tr>
<td>BODY NO.</td>
<td>Number of this particular body. This parameter input on body control card.</td>
</tr>
<tr>
<td>CHORD</td>
<td>The reference chord for the system.</td>
</tr>
<tr>
<td>COSA</td>
<td>The cosine of DALPHA</td>
</tr>
<tr>
<td>CP</td>
<td>The pressure coefficient on a body element.</td>
</tr>
<tr>
<td>DALPHA</td>
<td>The change in angle between consecutive elements of a body. (degrees)</td>
</tr>
<tr>
<td>DELTAS</td>
<td>The length of a body element.</td>
</tr>
<tr>
<td>MACH NO.</td>
<td>Mach number used in Gothert's transformation.</td>
</tr>
<tr>
<td>MX</td>
<td>The factor by which all X-coordinates are multiplied for one body. Input on body transformation card.</td>
</tr>
<tr>
<td>MY</td>
<td>The factor by which all Y-coordinates are multiplied for one body. Input on body transformation card.</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>N</td>
<td>Velocity normal to a surface element, this is a measure of how well the boundary condition of zero normal velocity is satisfied.</td>
</tr>
<tr>
<td>NN</td>
<td>The number of geometry data points for a given body. This is input on the body transformation card.</td>
</tr>
<tr>
<td>NNU</td>
<td>The number of non-uniform onset flows to be considered.</td>
</tr>
<tr>
<td>PHI</td>
<td>Value of potential on each surface element.</td>
</tr>
<tr>
<td>PSF NO.</td>
<td>Identification for this case.</td>
</tr>
<tr>
<td>SIGMA</td>
<td>Source density on each surface element.</td>
</tr>
<tr>
<td>SINA</td>
<td>Sin of DALPHA</td>
</tr>
<tr>
<td>SUM(T) DELTA(S)</td>
<td>This is the summation of T multiplied by Deltas up to each element midpoint.</td>
</tr>
<tr>
<td>SUMDS</td>
<td>Summation of Deltas, surface distance around the body.</td>
</tr>
<tr>
<td>TCNST</td>
<td>Constant value of tangential velocity used in special option.</td>
</tr>
<tr>
<td>THETA</td>
<td>The angle through which a body is to be rotated about the origin in a clockwise direction.</td>
</tr>
<tr>
<td>TI</td>
<td>The velocity at each midpoint.</td>
</tr>
<tr>
<td>VOLUME</td>
<td>The volume of the body being analyzed. (calculated by Neumann)</td>
</tr>
<tr>
<td>X</td>
<td>The input X-coordinate defining the body surface, or off-body X-coordinates.</td>
</tr>
<tr>
<td>XE</td>
<td>The value of semi-major axis used in ellipse generation option.</td>
</tr>
<tr>
<td>Y</td>
<td>The input Y-coordinates defining the body surface, or off-body Y-coordinates.</td>
</tr>
<tr>
<td>YE</td>
<td>The value of semi-minor axis used in ellipse generation option.</td>
</tr>
</tbody>
</table>
Output of this routine consists of CASE DATA and STATION DATA inputs as well as the computed STATION DATA. Body geometry data, flags and counters, and reference quantities are printed out under the heading of CASE DATA. Values of parameters at the outer edge of the boundary layer as well as the boundary condition inputs are printed out under the heading of STATION DATA. These are followed by iteration results, velocity profiles for each x-station (if FLG32 = 0), and a summary of the computed boundary-layer parameters as functions of streamwise or x-distance.

Error messages generated by the program are printed out at the end of the STATION DATA printout if they are generated by input errors. Other error messages are issued at different locations in the profile printout if errors are detected during the computations.

### SYMBOL DESCRIPTION

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPH1</td>
<td>Local body slope $dy/dx$.</td>
</tr>
<tr>
<td>ALPH2</td>
<td>Not used in this program.</td>
</tr>
<tr>
<td>BETA</td>
<td>$\beta = (2\xi/u_e)(du_e/d\xi)$</td>
</tr>
<tr>
<td>C</td>
<td>CHORD</td>
</tr>
<tr>
<td>CDBASE</td>
<td>Base drag coefficient.</td>
</tr>
<tr>
<td>CF</td>
<td>$c_f = \tau_w/(1/2 u_e^2)$, value of local skin friction coefficient.</td>
</tr>
<tr>
<td>CFA</td>
<td>Total integrated skin friction to each point.</td>
</tr>
<tr>
<td>$C_p$</td>
<td>Pressure coefficient</td>
</tr>
<tr>
<td>DELS</td>
<td>Boundary layer displacement thickness.</td>
</tr>
<tr>
<td>DELVW</td>
<td>Delta $V(1,2)$ used in iteration for $V(1,2)$.</td>
</tr>
<tr>
<td>EPS</td>
<td>$\epsilon^+$, eddy viscosity parameter for outer region.</td>
</tr>
<tr>
<td>EPS1</td>
<td>$\epsilon_1$, eddy viscosity parameter for inner region.</td>
</tr>
<tr>
<td>ETA</td>
<td>$\eta$, non-dimensionalized boundary layer thickness to each point in the boundary layer.</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ETAE</td>
<td>( n_\infty ) value of ( n ) which corresponds to ( \delta ).</td>
</tr>
<tr>
<td>ETAINF</td>
<td>Non-dimensional boundary layer thickness used as maximum value in forming numerical solution grid.</td>
</tr>
<tr>
<td>F,FP,FPP</td>
<td>( f, f', f'' ), respectively.</td>
</tr>
<tr>
<td>FPPW</td>
<td>( f'' ) at the wall.</td>
</tr>
<tr>
<td>FPW</td>
<td>( f' = U/U_e ) at the wall.</td>
</tr>
<tr>
<td>FW</td>
<td>( f_w ), this is the transformed stream function at the wall.</td>
</tr>
</tbody>
</table>
|          | \[
|          | f_w = \frac{1}{(2\xi)^{1/2}} \int_0^\xi \frac{V_w}{\mu_e U_e} d\xi \]
<p>| GW       | Not used in this method.                                                   |
| GPW      | Not used in this method.                                                   |
| H        | Boundary layer form factor, ( H = \delta/\theta )                       |
| HE       | Enthalpy                                                                   |
| HL       | Initial step size, same as DETAIL in input.                                |
| IMAK     | Number of points taken through the boundary layer.                         |
| K        | Initial step size of the variable grid system.                             |
| KK       | Variable grid parameter chosen internally.                                |
| ME       | Local Mach number.                                                         |
| MUE      | Local dynamic viscosity, ( \mu_e ) at edge of boundary layer.            |
| MREF     | Free stream Mach number.                                                   |
| MUREF    | Free stream dynamic viscosity, ( \mu_\infty ).                           |
| PE       | Pressure at edge of boundary layer, ( P_e ).                             |
| PRO      | Laminar Prandtl number.                                                    |
| QW       | Not used in this program.                                                  |
| REY      | Reynolds number based on reference conditions (see input)                  |</p>
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHOREF</td>
<td>Freestream reference density.</td>
</tr>
<tr>
<td>R_o/L</td>
<td>Axisymmetric radius.</td>
</tr>
<tr>
<td>RR</td>
<td>Not used by this program.</td>
</tr>
<tr>
<td>RTHETA</td>
<td>Momentum thickness Reynolds number, ( R_\theta ).</td>
</tr>
</tbody>
</table>
| R_x      | Reynolds number based on local conditions. \[
|          | \( R_x = \frac{u_x}{\nu} \)                                               |
| S        | Surface distance.                                                           |
| S/C      | Nondimensionalized surface distance.                                         |
| SHORTP   | Flag which tells program to print velocity profiles. Same as FLG32 in input.|
| SQUIG    | Transformed x-coordinate, \( \xi \) \[
<p>|          | ( \xi = \int_{0}^{X} \rho e^\mu u_x \left( e^{\frac{X}{L}} \right)^{2k} dx ) |
| ST       | Not used in this program.                                                   |
| SWEEP    | Not used in this program.                                                   |
| TE       | Temperature through boundary layer. Not needed for this program.            |
| THETA    | Momentum thickness, ( \theta ).                                           |
| TREF     | Reference temperature, ( T_{\infty} ).                                    |
| TRFLAG   | Flag which determines transition (input).                                   |
| TRINT    | Flag which determines instantaneous transition or use of transitional region option (input). |
| TW       | Temperature at the wall. Not used in this program.                          |
| TVC      | Transverse curvature flag (input).                                          |
| UE       | Velocity at edge of boundary layer.                                         |
| UPLUS    | Non-dimensionlized velocity in the boundary layer.                          |</p>
<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X-coordinate</td>
</tr>
<tr>
<td>X/C</td>
<td>Non-dimensionalized x-coordinate</td>
</tr>
<tr>
<td>XI</td>
<td>Transformed x-coordinate - same as SQUIG</td>
</tr>
<tr>
<td>Y</td>
<td>Y-coordinate</td>
</tr>
<tr>
<td>Y/C</td>
<td>Non-dimensionalized y-coordinate</td>
</tr>
<tr>
<td>YPLUS</td>
<td>Non-dimensionalized y-coordinate in boundary layer.</td>
</tr>
<tr>
<td>VREF</td>
<td>Reference velocity (input)</td>
</tr>
</tbody>
</table>

**Iteration Subprogram**

XNEW and YNEW. These are the coordinates of the equivalent viscous body. The original coordinates modified by the addition of the boundary layer displacement thickness $\delta^*$. 
DESCRIPTION OF STORAGE UNITS

The following is a description of all disk storage units used in ADAM:

<table>
<thead>
<tr>
<th>TAPE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAPE1</td>
<td>This unit used in subroutine SOLVIT as a scratch unit and also used to transfer the &quot;viscous&quot; coordinates from subroutine iterat to subroutine smooth or subroutine BASIC1.</td>
</tr>
<tr>
<td>TAPE2</td>
<td>This unit used in subroutine SOLVIT as a scratch unit and also used to transfer the boundary layer displacement thickness's from subroutine OTPT to subroutine ITERAT.</td>
</tr>
<tr>
<td>TAPE3</td>
<td>This unit used to store source densities in subroutine SOLVIT and used to transfer body geometry and pressures from subroutine AXIS to subroutine INPT.</td>
</tr>
</tbody>
</table>
| TAPE4  | \(\sin \alpha, \cos \alpha, TCNST, TG(I), \cos R^2, 2|\sin \alpha \cos \alpha|, N_o, T_o, V_N, T_T, A(J), B(J), \text{etc.}\) 
Used exclusively in Neumann subprogram to store and transfer data. |
| TAPE5  | This tape used for card input. |
| TAPE6  | This tape used for printed output. |
| TAPE8  | This tape used to store extra cross flow matrices, EC, ECY, ECZ, in subroutine MATRIX. |
| TAPE9  | This tape used to store axisymmetric flow matrices AS, AY, AZ, in subroutine MATRIX. |
| TAPE10 | This tape used to store cross flow matrices CX, CY, CZ in subroutine matrix and also used to transfer smoothed
<table>
<thead>
<tr>
<th>TAPE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAPE10 (Continued)</td>
<td>coordinate data from subroutine smooth to subroutine BASIC1.</td>
</tr>
<tr>
<td>TAPE11</td>
<td>This tape used as a scratch unit in subroutine SOLVIT.</td>
</tr>
<tr>
<td>TAPE12</td>
<td>This tape used to store transformed parameters $X_1, Y_1, X_2, Y_2,$ and $\Delta S_i$.</td>
</tr>
<tr>
<td>TAPE13</td>
<td>This tape used to store untransformed coordinates $(TX_1, TY_1)$ for use in SUBCASE option.</td>
</tr>
<tr>
<td>TAPE15</td>
<td>This tape used to store transformed coordinates, $(X_1, Y_1)$ for use in subroutine ITERAT.</td>
</tr>
</tbody>
</table>
APPENDIX C

PROGRAM LISTINGS

This part of the report contains the source card listings for the axisymmetric design and analysis method (ADAM) computer program. This program may be run either on a CDC or an IBM computer. The listing as presented here is for the CDC version of the program. This program has been run on the CDC 6600 computer. The program is written in FORTRAN for the CDC run compiler and has been run under the scope 3.1 and 3.4 operating systems. In this listing all cards that are peculiar to the CDC version of FORTRAN are identified by a C in card column 80. All cards that are peculiar to the IBM FORTRAN IV compiler are identified by an I in card column 80 and a C in card column 1. In other words, the code for both CDC and IBM machines is in the deck but the IBM cards are made inactive by converting them to comment statements (C in card column 1). Since all of the machine dependent cards are identified by an I or C in card column 80 it is a simple matter to convert the deck from one version to the other with a small conversion program. When converting from CDC to IBM code this conversion program reads and copies each card to a storage unit. If a card has a C in card column 80, then a C is written into card column 1 to make the CDC peculiar card inactive. If a card has an I in card column 80, then the C is removed from card column 1 and the card image written to the storage unit as an active FORTRAN statement. The conversion from IBM back to CDC is made in a similar manner. The conversion program to convert the deck from CDC to IBM FORTRAN is listed below (for use on an IBM machine):

```fortran
DIMENSION DATA(22)
DATA CB,CC,CI/H, 1HC,1HI/
REWIND 19
DO 100 I=1,20000
   READ (5,20,END=300) DATA
20 FORMAT (1A1,19A4,1A2,1A1)
   IF (DATA(22) .EQ. CI) DATA(1) = CB
   IF (DATA(22) .EQ. CC) DATA(1) = CC
   WRITE (19,20) DATA
100 CONTINUE
300 STOP
END
```
This program places the new deck with IBM cards made active, and CDC cards inactive, on to unit 19. The references to unit 19 above can be changed to unit 7 to punch the deck out.
OVERLAY(AXSY,0,0)
PROGRAM MAIN(INPUT=201,OUTPUT,TAPES=INPUT,TAPE6=OUTPUT,
1 TAPE1=201,TAPE2=201,TAPE3=201,TAPE4=201,TAPE8=201,
2 TAPE9=201,TAPE10=201,TAPE11=201,TAPE17=201,TAPE13=201,TAPE15=201)
C
BBAC MAIN PROGRAM

THIS IS THE AXISYMMETRIC DESIGN AND ANALYSIS METHOD, ADAM.
COMPUTER PROGRAM. THIS COMPUTER PROGRAM WILL CALCULATE THE
AERODYNAMIC FORCES ACTING ON AN AXISYMMETRIC BODY OPERATING
IN A VISCOS INCOMPRESSIBLE FLOW FIELD

1 FORMAT(5I4)
2 READ(5,1) IGEOM,INEUM,IBOUND,ITER,IFINISH

THESE FOUR FLAGS DETERMINE WHICH ROUTINES WILL BE USED

IGEOM CONTROLS THE GEOMETRY DEFINITION
IF IGEOM = 0 NO SMOOTHING IS USED
IF IGEOM = 1 SMOOTHING IS USED

INEUM INDICATES WHETHER OR NOT A POTENTIAL FLOW SOLUTION WILL
BE GENERATED
IF INEUM = 0 NO POTENTIAL FLOW SOLUTION IS USED
IF INEUM = 1 A POTENTIAL FLOW SOLUTION IS USED

IBOUND INDICATES WHETHER OR NOT A BOUNDARY LAYER SOLUTION IS
DESIRED
IF IBOUND = 0 NO BOUNDARY LAYER SOLUTION IS NEEDED
IF IBOUND = 1 A BOUNDARY LAYER SOLUTION IS NEEDED

ITER CONTROLS THE ITERATION CYCLE
IF ITER = 0 NO ITERATION IS NEEDED
IF ITER = 1 AN ITERATION IS NEEDED
IGEOM = IGEOM + 1
INFUM = INFUM + 1
ITBOUND = ITBOUND + 1
ITFR = ITFR + 1

C GO TO (30,20), IGEUM

C 20 CALL SMOOTH
20 CALL OVERLAY(4HAXSY,6,0,6HRECALL)

C 30 GO TO (60,50), INEUM

C 50 CALL NEUMAN

C 60 GO TO (90,80), ITBOUND

C 80 CALL BOUNDL
80 CALL OVERLAY(4HAXSY,5,0,6HRECALL)

C 90 GO TO (120,110), ITER

C 110 CALL ITERAT
110 CALL OVERLAY(4HAXSY,7,0,6HRECALL)

C 120 IF (IFINSH .NE. 9999) GO TO 2

C STOP
200 CONTINUE
FND
**DOUGLAS NEUMANN POTENTIAL FLOW PROGRAM**

**CALCULATION OF POTENTIAL FLOW ABOUT BODIES OF REVOLUTION HAVING FLOWS PARALLEL AND PERPENDICULAR TO THE AXIS OF REVOLUTION.**

**MAIN PROGRAM**

```plaintext
COMMON /IPSF/ PSF
COMMON /NBSAVE /NROLD, NIN
COMMON HEIR(10), CASE, NB, NNU
1 ,FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
2 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
3 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
4 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
5 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27
COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NANC(5), TYPEC(5), NFL(11), IFC, NSIGEC,
3 TYPEC(5), NUNEC(5)
COMMON/ITERF/ ITER
DOUBLE PRECISION HEIR, CASE
INTEGER FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
1 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
2 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
3 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
4 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27
REAL MN
NROLD = 0
```

REWIND 3
REWIND 4
REWIND 8
REWIND 9
REWIND 10
REWIND 11
10 REWIND 12
REWIND 13
REWIND 15
C CALL PART1
CALL OVERLAY (4HAXSY, 1, 0, 6HRECALL )
TF(FLG06, NE, 0) GO TO 50
C 30 CALL PREP
30 CALL OVERLAY (4HAXSY, 3, 0, 6HRECALL )
C 40 CALL PART4
40 CALL OVERLAY (4HAXSY, 4, 0, 6HRECALL )
50 RETURN

SUBROUTINE INS1 ( ARG, TABLE, NTPT, NLQ, NFR )

INS1 - SINGLE LINEAR OR QUADRATIC INTERPOLATION
ONE OR TWO FUNCTIONS OF ONE VARIABLE

THIS SUBROUTINE WILL INTERPOLATE FOR EITHER
1) F(X) FROM A TABLE OF X VRS F(X), OR
2) F(X) AND G(X) FROM A TABLE OF X VRS F(X), G(X).
THE TABLE MAY HAVE UNEQUAL SPACING IN X, EITHER LINEAR
OR QUADRATIC LAGRANGIAN INTERPOLATION MAY BE USED.

ARG   = INPUT   = X ARGUMENT

TABLE  = INPUT  = IS A LINEAR ARRAY. THE FIRST WORD IS AN
        INTEGER CODE (EITHER INTEGER FORM OR REAL*4
        FORM), IF THIS CODE IS POSITIVE, THE CODE
        SPECIFIES THE NUMBER OF X,F(X) PAIRS
        IMMEDIATELY FOLLOWING THE CODE IN SUCCESSIVE
        WORDS, IF THE CODE IS NEGATIVE,
        ABSOLUTE VALUE OF THE CODE SPECIFIES THE
        NUMBER OF X,F(X),G(X) TRIPLETS IMMEDIATELY
        FOLLOWING THE CODE IN SUCCESSIVE WORDS.
        THE X-VALUES MUST BE IN ASCENDING ORDER,
        EXCEPT FOR THE CODE, THE INPUT TABLE VALUES
        MUST BE IN REAL*4 FORM

NTPT  = OUTPUT = INTERPOLATED VALUE OF F(X) IF TABLE(1) IS
        POSITIVE
        = A TWO WORD ARRAY (IF TABLE(1) IS NEGATIVE)
        CONTAINING THE INTERPOLATED VALUES
        FOR F(X) AND G(X)

NLQ   = INPUT   = INTERPOLATION FLAG (INTEGER)
        = 1 FOR LINEAR INTERPOLATION
        = 2 FOR QUADRATIC INTERPOLATION
C NER = OUTPUT = ERROR CODE (INTEGER)
C = 1 INTERPOLATION SUCCESSFUL
C = 2 BELOW TABLE, MINIMUM VALUE FURNISHED
C = 3 ABOVE TABLE, MAXIMUM VALUE FURNISHED
C = 4 NOT ENOUGH ENTRIES = NO ANSWER
C = 5 X-VALUES NOT IN ASCENDING ORDER = NO ANSWERS

C DIMENSION DTPT(2), TABLE(1)
C EQUIVALENCE ( NOENTR, A )

C A = ABS( TABLE(1) )
M = 2
IF ( TABLE(1) .LT. 0.0 ) M = 3
MM1 = M - 1
MP1 = M + 1
IF ( A .GT. 0.99E0 ) NOENTR = A + 0.5E0
J = 3

C CHECK FOR NUMBER OF ENTRIES
C IF ( NLQ .NE. 1 ) GO TO 20
  IF ( NOENTR .GE. 2 ) GO TO 30
  10 NER = 4
  GO TO 140
  20 IF ( NOENTR .LT. 3 ) GO TO 10

C CHECK FOR ARGUMENT LESS THAN OR EQUAL TO LOW LIMIT
C 30 IF ( ARG - TABLE(2) ) .LT. 40,50,60
  40 NER = 2
  GO TO 130
  50 NER = 1
GO TO 130

C SEARCH FOR ENTRY WITHIN TABLE

60 NOS = M * NOFNTX
   IST = 2 + M
   DO 90 I = IST, NOS, M
   J = I + 1
   IF ( TABLE(I) = TABLE(I-M) ) 70, 70, 80
   70 NER = 5
   GO TO 140
   80 IF ( TABLE(I) = ARG ) 90, 50, 100
   90 CONTINUE
   NER = 3
   GO TO 130

C SEARCH SUCCESSFUL, TEST INTERPOLATION TYPE

100 IF ( NLO .GT. 1 ) GO TO 110

C USE ALINEAR INTERPOLATION

C NER = 1
   NTPT (1) = TABLE(I+1) * ( ARG - TABLE(I-M) ) / ( TABLE(I) -
   TABLE(I-M) ) + TABLE(I-MM1) * ( ARG - TABLE(I) ) /
   ( TABLE(I-M) - TABLE(I) )
   IF ( M .LE. 3 ) GO TO 140
   NTPT (2) = TABLF(I+2) * ( ARG - TABLE(I+7) ) / ( TABLE(I) -
   TABLE(I-7) ) + TABLF(I-1) * ( ARG - TABLF(I) ) /
   ( TABLE(I-M) - TABLE(I) )
   GO TO 140

C USE QUADRATIC INTERPOLATION

110 I = I - M
IF ( I .GE. 2*M ) GO TO 120
I = I + M

120 XA1 = ARG - TABLE(I)
XA0 = ARG - TABLE(I-M)
XAP = ARG - TABLE(I+M)
X01 = TABLE(I=M) = TABLE(I)
X0P = TABLE(I=M) = TABLE(I+M)
X1P = TABLE(I) - TABLE(I+M)
NER = 1

PTPT (1) = TABLE(I=M) * (XA1 / X01) * (XA2 / X02)
1 TABLE(I+1) * (XA0 / X01) * (XA2 / X12)
2 TABLE(I+M) * (XA0 / X02) * (X1A / X12)

IF ( M .NE. 3 ) GO TO 140

PTPT (2) = TABLE(I-1) * (XA1 / X01) * (XA2 / X02)
1 TABLE(I+1) * (XA0 / X01) * (X1A / X12)
2 TABLE(I+5) * (XA0 / X02) * (X1A / X12)

GO TO 140

C FRRMR EXIT = SET OUTPUT VALUE
C

130 PTPT (1) = TABLE(J)
IF ( M .LE. 3 ) PTPT (2) = TABLE(J+1)
C
C NORMAL EXIT
C

140 RETURN
END
FUNCTION ARSIN(X)
C THIS ROUTINE IS REQUIRED BECAUSE OF DIFFERENCES BETWEEN CRAY AND IBM
C FORTRAN,
  ARSIN = ASIN(X)
RETURN
END
OVERLAY(AXSY,1,0)
PROGRAM PART1
C
C
SUBROUTINE PART1

C

* CONTROL FOR BASIC DATA AND FORM MATRIX

COMMON / NBSAVE / WBOLD, NIN
COMMON / RNMGND / VA(100,2), VR(100,2), VAN(100), VAT(100)
COMMON / ECF / ECX(100), ECY(100), ECZ(100)
COMMON / D, D3, XMJX, YMYJ, XMJJPY, YMYJPY, S
COMMON HEADR(10), CASE, NB, NNU

1 ,FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
2  ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
3   ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
4    ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
5     ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1  NER1, NER2, NMA, NSIGA, NSIGC,
2    NUNC(5), TYPEC(5), NLF(11), IFC, NSIGEC,
3     TYPEC(5), NUNEC(5)

C

DOUBLE PRECISION HEADR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1  ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
2   ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
3    ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
4     ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

COMMON / IPSF/ PSF
REAL MN

C

COMMON / CL / X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1   SINA(100), COSA(100), XP(100), YP(100)
2   ,XWAKE(11), YWAKE(11)
COMMON / TL / TX1(100), TY1(100), NG(100), TG(100), ALFA(100),
1   BDRS(100), DALT(100), CHORD, TCNST, DUMMY(1315)
COMMON / ITERF / ITER
C * START
C * READ INPUT DATA
100 READ (5,4) HDR, CASE, PSF, NA, NNU, FLG03, FLG04, FLG05, FLG06,
   FLG07, FLG08, FLG09, FLG10, FLG11, FLG12, FLG13, FLG14,
   FLG15, FLG16, FLG17, FLG18, FLG19, FLG20, FLG21, FLG22,
   FLG23, FLG24, FLG25, FLG26, FLG27, NIN, ITER
C*** **TRIANGULARIZATION OF THE MATRIX (SOLVIT) IS THE DEFAULT SOLUTION
IF(FLG09.EQ.0.AND.,FLG10.EQ.0)FLG13=1
C*** **FLG22 IS GENERATED (RESEP) BOUNDARY CONDITIONS
C*** **FLG21 IS EXTRA CROSS FLOW
1 IF (FLG22.LE.0) GO TO 5
   FLG21 = 1
   FLG03 = 1
   FLG04 = 1
C*** **IF FLAG 18 IS NOT EQUAL TO FLAG 14 YOU MUST USE DIRECT MATRIX
5 IF (FLG18.NE.FLG14) GO TO 2
   IF (FLG21.LE.0) GO TO 3
   FLG12 = 1
2 FLG13 = 1
   FLG09 = 0
   FLG10 = 0
3 CONTINUE
   IF( NBOLD .EQ. 0 ) NBOLD = NB
C*** **CARDS (UNIT 5) ARE THE DEFAULT METHOD OF INPUT
4 FORMAT (10A6, 2X A6, 8X A4/ 2711, I2, I1)
5 FORMAT (5,8) CHORD, MN, TCNST
8 FORMAT (3F10,0)
C*** **THE DEFAULT CHORD LENGTH IS 1.0
C IF(CHORD,GT.,-1.0E-5.AND.,CHORD,LT.,1.0E-5)CHORD=1.0
12 FORMAT (1M1 25X, 26HDODGAL AIRCRAFT COMPANY /)
28X, 21HLONG BEACH DIVISION

6X, 43HPROGRAM E0DA = AXISSYMMETRIC AND CROSSFLOW

11X, 29H**** CASE CONTROL DATA ****

6X, 10A6, 4X, 10HCASE NO. A6

6X 9HBODIES =13/ 6X 9HNUM =13/ 6X 9HCWORD =F12,7/

6X 9HMACH NO. =F12,8/ 6X 9HTCNST =F12,7/

6X 9HPSF NO. = A4///

TF (FLG03,GT,0) WRITE (6,16)

16 FORMAT (13X 21HSURFACE OF REVOLUTION )

TF (FLG04,GT,0) WRITE (6,20)

20 FORMAT (13X 9HCROSSFLOW)

IF (FLG05,GT,0) WRITE (6,24)

24 FORMAT (13X 15HOFF=BODY POINTS )

IF (FLG06,GT,0) WRITE (6,28)

28 FORMAT (13X 15HBSAC DATA ONLY )

IF (FLG07,GT,0) WRITE (6,32)

32 FORMAT (13X 17HELLIPSE GENERATOR )

IF (FLG08,GT,0) WRITE (6,36)

36 FORMAT (13X 14HPRINT MATRICES )

IF (FLG09,GT,0) WRITE (6,40)

40 FORMAT (13X 10HOLD SEIDEL )

IF (FLG10,GT,0) WRITE (6,44)

44 FORMAT (13X 31HMODIFIED SEIDEL MATRIX SOLUTION)

IF (FLG11,GT,0) WRITE (6,48)

48 FORMAT (13X 18HPERTURBATIONS ONLY )

IF (FLG12,GT,0) WRITE (6,52)

52 FORMAT (13X 22HSOLVE POTENTIAL MATRIX )

IF (FLG13,GT,0) WRITE (6,56)

56 FORMAT (13X 47HMATRIX SOLUTION BY TRIANGULARIZATION (SOLVIT))

IF (FLG14,GT,0) WRITE (6,53)

53 FORMAT ( 13X 30HPRESCRIBED TANGENTIAL VELOCITY )

IF (FLG18,GT,0) WRITE (6,69)

69 FORMAT ( 15X 22H WITH SURFACE VORTICITY )

IF (FLG15,GT,0) WRITE (6,54)

54 FORMAT ( 13X 12HSTRIP VORTEX )
IF \( \text{FLG16, GT, 0} \) WRITE (6, 64)  
64 FORMAT (13X 40HOMIT AXI-SYMMETRIC UNIFORM FLOW SOLUTION )  
IF \( \text{FLG17, GT, 0} \) WRITE (6, 68)  
68 FORMAT (13X 36HOMIT CROSSFLOW UNIFORM FLOW SOLUTION )  
IF \( \text{FLG19, GT, 0} \) WRITE (6, 72)  
72 FORMAT (13X 30HOMIT PRESCRIBED VORTICITY)  
IF \( \text{FLG20, GT, 0} \) WRITE(6, 74)  
74 FORMAT(13X 24HOMIT TOTAL VORTICITY )  
IF \( \text{FLG21, GT, 0} \) WRITE(6, 76)  
76 FORMAT ( 13X 18HOMIT EXTRA CROSS FLOW )  
IF \( \text{FLG22, GT, 0} \) WRITE(6, 78)  
78 FORMAT(13X 82HOMIT GENERATED BOUNDARY CONDITIONS FOR 3 AXI-SYMMETRIC, 1 CROSS, AND 1 EXTRA CROSS FLOW, )  
IF \( \text{FLG23, LT, 0} \) GO TO 81  
WRITE(6, 79)  
79 FORMAT(13X 16HOMIT WING OPTION )  
FLG03 = 1  
FLG13 = 1  
FLG15 = 1  
FLG19 = 1  
81 IF \( \text{FLG19, GT, 0} \)FLG18 = 1  
IF \( \text{FLG22, GT, 0, AND, NB, NE, 2} \) GO TO 82  
GO TO 84  
82 WRITE(6, 83)  
83 FORMAT (128HOMIT WHEN GENERATED RESSEP BOUNDARY CONDITIONS ARE USED, NUMBER OF BODIES MUST BE EXACTLY TWO, YOU CONFUSED, EXECUTION TERMINATING.)  
STOP  
84 IF \( \text{FLG22, GT, 0, AND, NNU, GT, 0} \)GO TO 86  
GO TO 88  
86 WRITE (6, 87)  
87 FORMAT (98HOMIT GENERATED RESSEP BOUNDARY CONDITIONS CANNOT HAVE NON-UNIFORM FLOW INPUT, EXECUTION TERMINATING.)  
88 CONTINUE  
WRITE ( 6,75 ) NIN
IF (FLG18.GE.0 .OR. FLG14.GT.0) GO TO 125
75 FORMAT (13X,5R,INPUT TAPE NO. FOR COORDINATES AND NON-UNIFORM FLOW)
1W ONLY = , 15 )
WRITE (6,60)
70 FORMAT (1H0//63H FLG14 MUST BE USED WITH FLG18 OR FLG19, EXECUTION
1N TERMINATED, )
STOP
125 IF (NNU.LE.0 .OR. FLG14.LE.0) GO TO 130
WRITE (6,60)
60 FORMAT (1H0//49H COLUMNS 2 AND 14 OF FLAG CARD ARE BOTH NON-ZERO,
A / 43H ILLEGAL COMBINATION, EXECUTION TERMINATED. )
STOP
C * READ DATA AND SETUP FOR UNIFORM FLOW
130 CALL BASIC1
C*** ***NSIGA AND NSIGC ULTIMATELY BECOME THE NUMBER OR RIGHT HAND SIDES
C*** ***IN AXISYMMETRIC FLOW AND CROSS FLOW RESPECTIVELY
133 NSIGA=0
 IF (FLG03.GT.0 .AND. FLG16.LE.0) NSIGA=1
NSIGC=0
 IF (FLG04.GT.0 .AND. FLG17.LE.0) NSIGC=1
 IF (FLG22.GT.0) GO TO 136
DO 135 I = 1,5
NUNA(I) = 123456
135 TYPEA(I) = 100
 IF (FLG23.GT.0) GO TO 141
GO TO 138
C*** ***PREPARE NUNA AND TYPEA FOR NON-UNIFORM AXISYMMETRIC FLOW, GENER
C*** ***(RESEP) BOUNDARY CONDITIONS
136 DO 137 I = 1,3
NUNA(I) = I
137 TYPEA(I) = 100
GO TO 138
C
C *** RING WING OPTION
C *** STRIP VORTEX FLOWS ALREADY HAVE NUNA(I) = 123456.
C *** MAKE PRESCRIBED VORTICITY FLOWS NUNA(J) = TO THEIR FLOW NO. J
C
141 ICNT = 0
   DO 142 I = 1, NB
      IF(NLF(I), GT, 0) GO TO 142
      ICNT = ICNT + 1
   142 CONTINUE
C
C *** ICNT IS THE NUMBER OF LIFTING BODIES
C *** NUMBER OF FLOWS IS 2 * ICNT + 1
C
   NFLOWS = 2 * ICNT + 1
   ICNTP2 = ICNT + 2
   DO 143 I = ICNTP2, NFLOWS
      143 NUNA(I) = I
   138 CONTINUE
C*** ***IF FLGO2 (NON-UNIFORM FLOW) IS NOT CHECKED INITIALLY, THE FLOW
C*** ***OF CONTROL WILL NEVER REACH BASIC2
   IF (NNU) 140, 150, 140
C
   CALL BASIC2
   150 CONTINUE
   160 REWIND 4
      IF (NSIGA, LE, 5) GO TO 180
   170 WRITE(6, 172)
   172 FORMAT (1H1, 75HAXI=SYMMETRIC OR CROSSFLOW NON-UNIFORM FLOWS EXCEED
      A 5, EXECUTION TERMINATED )
      STOP
   180 IF (NSIGC, GT, 5) GO TO 170
      IF (FLG15, LE, 0, OR, FLGO2, GT, 0) GO TO 200
      WRITE (6, 190)
   190 FORMAT (64H1STRIP RING VORTEX OPTION MUST USE SURFACE OF REVOLUTION
      IN OPTION, / 22H EXECUTION TERMINATED. )
      STOP

PAR1 176
PAR1 177
PAR1 178
PAR1 179
PAR1 180
PAR1 181
PAR1 182
PAR1 183
PAR1 184
PAR1 185
PAR1 186
PAR1 187
PAR1 188
PAR1 189
PAR1 190
PAR1 191
PAR1 192
PAR1 193
PAR1 194
PAR1 195
PAR1 196
PAR1 197
PAR1 198
PAR1 199
PAR1 200
PAR1 201
PAR1 202
PAR1 203
PAR1 204
PAR1 205
PAR1 206
PAR1 207
PAR1 208
PAR1 209
PAR1 210
200 IF (FLG15,LE,0) GO TO 230
   J = 0
   DO 210 I = 1, NB
210 IF (MLF(I),LE,0) J=J+1
   IF (NSIGA + J,LE, S ) GO TO 230
   WRITE (6,220)
220 FORMAT (6AM1GENERATED STRIP VORTEX UNSY FLOWS (ONE FOR EACH LIFTING BODY) PLUS/34H INPUT NON-UNIFORM FLOWS EXCEED S, 2 PHASE EXECUTION TERMINATED, )
   STOP
230 IF (FLG06,NE,0) GO TO 235
   CALL MATRIX
C 235 RETURN
235 CONTINUE
END
SURROUTINE BASIC1

* READ DATA AND SETUP FOR UNIFORM FLOW

COMMON /NBSAVE/NROLD,NIN

COMMON /NEDR(10),CASE,ND,NNU

1, FLG03, FLG04, FLG05, FLG06, FLG07
2, FLG08, FLG09, FLG10, FLG11, FLG12
3, FLG13, FLG14, FLG15, FLG16, FLG17
4, FLG18, FLG19, FLG20, FLG21, FLG22
5, FLG23, FLG24, FLG25, FLG26, FLG27

COMMON NT, NDC(11), MN, NUNA(5), TYPEA(5), NUNC(5), TYPFEC(5), NLF(11), IEC, NSIGEC

1, NER1, NER2, NMA, NSIGA, NSIGC
2, NUNC(5), TYPFEC(5), NLF(11), IEC, NSIGEC
3, TYPFEC(5), NUNEC(5)

DOUBLE PRECISION NEDR, CASE

INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1, FLG08, FLG09, FLG10, FLG11, FLG12
2, FLG13, FLG14, FLG15, FLG16, FLG17
3, FLG18, FLG19, FLG20, FLG21, FLG22
4, FLG23, FLG24, FLG25, FLG26, FLG27

DIMENSION COSSQR(100), RNS(100)

REAL MN

COMMON /CL/ X1(100), Y1(100), X2(100), Y2(100), DELS(100), SINA(100), COSA(100), XP(100), YP(100)
1, XWAKE(11), YWAKE(11)

COMMON /TL/ TX1(100), TY1(100), NG(100), TG(100), ALFA(100), RS(100), DAV(100), CHORD, TCNST, DUMMY(1315)

INTEGER BDN, SUBKS

REAL MX, MY, NG

C

* START

NT=0
K=0
K2=NR
IF ( NIN =EQ. 0 ) NIN = 10
IF (FLG05, NE, 0) K2=NB+1
C
* MAJOR LOUP * N. OF BODIES + OFF BODY POINTS
LCNT = 0
DO 1000 L=1,K2
READ (5,15) NN, MX, MY, THETA, ADDX, ADDY
15 FORMAT (5X, I5, 5F10.0)
READ (5,16) RDN, SUBKS, NLF(L), XF, YE
16 FORMAT (3(5X, I5), 2F10.0)
C*** ***ND(L) IS THE NUMBER OF POINTS ON BODY L, OR THE NUMBER OF OFF
C*** ***BODY POINTS FOR L = NB + 1
ND(L)=NN
M=NN-1
IF (SUBKS) 140,150,140
140 IF ( L ,NE, K2 ) GO TO 148
NTIMES = NBOLD + NB
IF ( NTIMES ,LE, 0 ) GO TO 148
DO 145 NSKIPS = 1, NTIMES
145 READ(13) ( TX(I), I=1,NN ), (TY(I), I=1,NN )
148 READ(13) ( TX(I), I=1,NN ), (TY(I), I=1,NN )
GO TO 220
150 IF (RDN,EQ,0) GO TO 200
IF (FLG07) 160,200,160
C
* ELLIPSE GENERATOR FOR X1 AND Y1
160 IF (XE,EQ,0,0) XE=1,
IF (YE,EQ,0,0) YE=1,
EN=M
DGAM=3.141593 /FN
GAM=3.141593
DO 170 I=1,NN
TX(I)=XE*COS(GAM)
TY(I)=YE*SIN(GAM)
170 GAM=GAM+DGAM
GO TO 210
* READ X1 AND Y1 FROM INPUT CARDS

200 DO 204 I=1,NN,6
   READ(NIN,20)TX1(I),TX1(I+1),TX1(I+2),TX1(I+3),TX1(I+4),TX1(I+5)
204 CONTINUE

206 CONTINUE

C

C*** * NB = FLG14 + 1 TO NB ARE PRESCRIBED VORTICITY BODIES

C

IF ( FLG23 ,LE, 0 ,OR, (L,LE, NB-FLG14 ,OR, L ,GT, NB))GO TO 210

C

C*** * IF CONTROL REACHES THIS POINT, RING WING OPTION IS IN EFFECT AND

C*** * L IS A PRESCRIBED VORTICITY BODY

C*** ** LCNT IS THE RELATIVE NUMBER OF THE WAKE BODY STARTING WITH 1

C

LCNT = LCNT + 1
XWAKF(LCNT) = TX1(NN)
YWAKE(LCNT) = TY1(NN)

C

* SAVE X1 AND Y1 FOR SUBCASE

210 WRITE (13) (TX1(I),I=1,NN),(TY1(I),I=1,NN)

C

* BASIC DATA CALC. AND PRINT (UNTRANSFORMED COORDINATES)

220 WRITE (6,24) HEADER, NN, MX, MY, THETA, ADDX, ADDY, XE, YE
24 FORMAT (1H1,'25X',26MDOUGLAS AIRCRAFT COMPANY/',
2     28X 21HLONG BEACH DIVISION // 5X 10A6 //
2     2  8X 4HNN = I4, 15X 4HMX = F13,7, 4X 4HMXY = F13,7 /
2     3  5X 7HTHETA = F13,7, 4X 6HMADDX = F13,7, 2X 6HMADDY =F13,7/
2     4  8X 4HXE = F13,7, 6X 4HYE = F13,7 )

IF (BDN) 240,230,240

230 WRITE (6,28) (I, TX1(I)), TY1(I), I=1,NN
28 FORMAT (1H0, 4X 36HOFF-BODY COORDINATES (UNTRANSFORMED) //
2     1 10X 5HX-OFF 9X 5HY-OFF // (1W I3, 2F14,7))

GO TO 270

240 SUM3=0.0

BAS1
DO 250 I=1,M
T1=TX1(I+1)=TX1(I)
T2=TY1(I+1)=TY1(I)
X2(I)=(TX1(I+1)+TX1(I))/2,
Y2(I)=(TY1(I+1)+TY1(I))/2,
DELS(I)=SQRT(T1*T1+T2*T2)
SUMS=SUMS+DELS(I)
RSDS(I)=SUMS
750 ALFA(I) = ATAN2( T2, T1 )
MA=M-1
DO 260 I=1,MA
260 DALF(I) = ( ALFA(I+1)-ALFA(I) ) * 57.29578
WRITE (6,36) BDN,TX1(I),TY1(I),X2(I),Y2(I),DELS(I),RSDS(I)
36 FORMAT (1H0, 4X, 35HON-BODY COORDINATES (UNTRANSFORMED) /
  1 9H RODY NO., I3/11X2M X 13X1MY 11X7HDELTAS 7X2
  2 5HSUMDS 8X THD ALPHA // 1H 3H1,2F14.7 / 4X 4F14.7)
  1 WRITE (6,40) (I, TX1(I), TY1(I), DALF(I-1), X2(I), Y2(I),
  2 DELS(I), RSDS(I), I=2,M), NN, TX1(NN), TY1(NN)
40 FORMAT (1H, I5, 2F14.7, 2AX4F14.7)
C
270 IF (MX) 280,300,280
280 DO 290 I=1,NN
290 TX1(I)=TX1(I)*MX
300 IF (MY) 310,330,310
310 DO 320 I=1,NN
320 TY1(I)=TY1(I)*MY
330 IF (THETA) 340,360,340
340 THETA = THETA / 57.29578
350 CSTHT = COS(THETA)
360 SNTHT = SIN(THETA)
DO 350 I=1,NN
T1=TX1(I)
TX1(I)=T1*CSTHT+TY1(I)*SNTHT
350 TY1(I)=TY1(I)*CSTHT-T1*SNTHT
360 IF (ADDX) 370,390,370

370 DU 380 I=1,NN
380 TX1(I)=TX1(I)+ADDX
390 IF (ADDY) 400,420,400
400 DO 410 I=1,NN
410 TY1(I)=TY1(I)+ADDY
420 IF (CHORD .EQ. 1.0 .OR. CHORD .EQ. 0.0 ) GO TO 450
430 DO 440 I=1,NN
440 TX1(I)=TX1(I)/CHORD
450 IF (MN) 460,475,460
460 SRM=SQRT(1.0*MN*MN)
470 TX1(I)=TX1(I)/SRM

C * SHIFT X1 AND Y1 TO COMMON /CL/
C*** ***IF BDN = 0.0, OFF BODY POINTS ARE BEING OPERATED ON
475 IF (BDN) 500,480,500
480 DO 490 I=1,NN
490 XP(I)=TX1(I)
490 YP(I)=TY1(I)
500 DO 510 I=1,NN
510 XI(K)=TX1(I)
510 YI(K)=TY1(I)
510 NT=NT+M
500 CONTINUE
1000 CONTINUE
RECORD 13
IF (FLG14.LE.0) GO TO 2000
IF (FLG14.LE.NB) GO TO 1050
WRITE (6,1025)
1025 FORMAT (ASH1,VALUE OF FLG14 EXCEEDS NO. OF BODIES, STOP, )
STOP
1050 IF (FLG14.NE.NB) GO TO 1075
NMA=0

BAS1
GO TO 1150
1075 L = NR-FLG14
    NMA = -L
    DO 1100 I = 1, L
C** *** NMA BECOMES THE NUMBER OF ELEMENTS ON THE 1ST L BODIES (IE THOSE
C** *** NOT HAVING AN INPUT VORTICITY OR VELOCITY)
1100 NMA = NMA * ND(I)
C** *** NR BECOMES THE NUMBER OF ELEMENTS RECEIVING AN INPUT VORTICITY
C** *** NR VELOCITY
1150 NR = NT=NMA
    IF (TCNST.GT.0.) GO TO 2000
    DO 1200 I = 1, NR, 6
    READ (5,20) TG(I),TG(I+1),TG(I+2),TG(I+3),TG(I+4),TG(I+5)
1200 CONTINUE
    C     * CALC. PARAMETERS WITH TRANSFORMED COORDINATES AND
    C     MACH NO. ADJUSTMENT
2000 N1=0
    J1=0
    DO 2500 K=1,NB
    M1=N1+1
    N1=N1+ND(K)+1
    DO 2400 J=M1,N1
    J1=J1+1
    T1=X1(J1+1)-X1(J1)
    T2=Y1(J1+1)-Y1(J1)
    X2(J)=X1(J1+1)+X1(J1))/2,
    Y2(J)=(Y1(J1+1)+Y1(J1))/2,
    DELS(J)=SQRT(T1*T1+T2*T2)
    COSA(J)=T1/DELS(J)
2400 SINA(J)=T2/DELS(J)
2500 J1=J1+1
    C     * SAVE PARAMETERS
    WRITE (12) (X1(I),I=1,J1),(Y1(I),I=1,J1),(X2(I),I=1,N1)
    WRITE (12) (Y2(I),I=1,NT),(DELS(I),I=1,NT)
    REWIND 12
C * SAVE SINA AND COSA ON TAPE 4 FOR CALC. OF MATRIX
C SOLUTION (RIGHT HAND MATRIX)
WRITE (4) (SINA(I), I=1, NT), (COSA(I), I=1, NT)
IF (FLG14) 2600, 2600, 2550
2550 IF (TCNST, GT, 0.0) WRITE(4) (TCNST, I=1, NR)
IF (TCNST, LE, 0.0) WRITE(4) (TG(I), I=1, NR)
2600 IF (FLG22, LE, 0) RETURN
NPR1 = ND(1) - 1
DO 2700 I = 1, NPR1
COSQR(I) = COSA(I)**2
2700 RHS(I) = 2.0 * ABS( SINA(I) * COSA(I) )
WRITE(4) (COSQR(I), I=1, NPB1), (RHS(I), I = 1, NPB1)
RETURN
END
SUBROUTINE BASIC2

* READ DATA AND SETUP FOR NON-UNIFORM FLOWS

COMMON / NBSAVE / NROLD, NIN
COMMON /HEDR(10), CASE, NB, NNU
1 ,FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
2 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
3 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
4 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
5 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNC(5), TYPEC(5), NLF(11), IEC, NSIGEC,
3 TYPFEC(5), NUNEC(5)

DOUBLE PRECISION HEDR, CASE
INTEGER FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
1 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
2 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
3 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
4 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

REAL MN

COMMON /CL/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1 SINA(100), COSA(100), XP(100), YP(100)
2 ,XHAWKE(11), YHAWKE(11)
COMMON /TL/ TX1(100), TY1(100), NG(100), TG(100), ALFA(100),
1 RSDS(100), DALF(100), CHORD, TCN81, DUMMY(1315)

INTEGER BDN
REAL MX ,MY ,NG

* START

* SETS OF NON-UNIFORM FLOW LOOP
NSIGEC = 0
KAE=0
120 DO 130 I = 1, NT
   T1 = NG(I)
   NG(I) = T1 * SINA(I) + TG(I) * COSA(I)
130   TG(I) = T1 * COSA(I) + TG(I) * SINA(I)
C     * WRITE BASIC DATA OUTPUT
140 WRITE (6,150) HDR, MSF, TYPE, FG, NG, NF, (NG(I), I=1, NT)
150 FORMAT (1H1, 25X, 26HDOUGLAS AIRCRAFT COMPANY /  
          1H1, 28X, 21HLONG BEACH DIVISION // 5X 10A6 //  
          2H1, 6X 5HMSF = I4, 10X 6HTYPE = F10.4, 10X 4MFG = F13.7 /  
          3H1, 1H0, 4X, 20HNON = UNIFORM FLOW NO. I6 /  
          4H1, 1H0, 4X, 10HLIST OF NG// (1H 6F14.7))
160 WRITE (6,160) (TG(I), I = 1, NT)
1000 CONTINUE
RETURN
END
SURROUNTE MATRIX

* COMPUTE MATRIX A, R, Z OR X, Y, Z

COMMON HEDR(10), CASE, NB, NNU
1 ,FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
2 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
3 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
4 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
5 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

COMMON NT, NO(11), MN, NUNA(5), TYPEA(5),
1 NER1 ,NER2 ,MNA ,NSIGA ,NSIGC
2 NUNC(5), TYPEC(5), NLF(11), IEC, NSIGEC
3 TYPEEC(5), NUNEC(5)

DOUBLE PRECISION HEDR, CASE
INTEGER FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
1 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
2 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
3 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
4 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27

REAL MN

LOGICAL PF

COMMON /ECS(100), ECS(100), ECS(100)
COMMON /RNGN/ VAC(100,2), VR(100,2), VAN(100), VAT(100)
COMMON /CL/ X1C(100), Y1C(100), X2C(100), Y2C(100), DELS(100),
1 SINA(100), COSA(100), XP(100), WP(100)
2 ,XWAKE(11), YWAKE(11)

COMMON /TL/ A(100), B(100), AX(100), AY(100), AZ(100),
1 CX(100), CY(100), CZ(100), AXV(100), AYV(100),
2 VN(100,5), VT(100,5), BON, IAC,
3 I, J, J1, SJ, D, M,
4 DX, DY, NI, XJ, YJ,
5 XK, EKK, EK, K, PF

103
* START

* INITIALIZE

L1=NT
RDN=0,0
YZRN=0,0

C**** ***TEST TYPE OF FLOW AND SET INDICATORS IAC AND IEC
C**** ***CROSS FLOW ONLY
IAC = -1 IEC = +1

C**** ***AXISYMMETRIC FLOW ONLY
IAC = +1 IEC = 0

C**** ***EXTRA CROSS FLOW ONLY
IAC = 0 IEC = 0

C**** ***CROSS FLOW AND AXISYMMETRIC FLOW
IAC = 0 IEC = +1

C**** ***CROSS FLOW AND EXTRA CROSS FLOW
IAC = -1 IEC = +1

C**** ***AXISYMMETRIC AND EXTRA CROSS FLOW
IAC = +1 IEC = +1

C**** ***AXISYMMETRIC, CROSS, AND EXTRA CROSS
IAC=0 IEC = +1

IF(FLG03)30,10,30
10 IF(FLG04)25,15,25
15 IAC = 0
IIEC = 0
GO TO 55
25 IAC = -1
GO TO 45
30 IF(FLG04)35,40,35
35 IAC = 0
GO TO 45
40 IAC = 1
45 IF(FLG21)50,53,50
50 IEC = +1
GO TO 55
53 IEC = -1
55 ASSIGN 110 TO K1
IF (FLG15,GT,0) ASSIGN 102 TO K1

60 DO 70 I=1,L1
65 VVT(I,J) = 0.
70 IF(I,J) = 0.
VAN(I) = 0.

MATX 036
MATX 037
MATX 038
MATX 039
MATX 040
MATX 041
MATX 042
MATX 043
MATX 044
MATX 045
MATX 046
MATX 047
MATX 048
MATX 049
MATX 050
MATX 051
MATX 052
MATX 053
MATX 054
MATX 055
MATX 056
MATX 057
MATX 058
MATX 059
MATX 060
MATX 061
MATX 062
MATX 063
MATX 064
MATX 065
MATX 066
MATX 067
MATX 068
MATX 069
MATX 070
70 VAT(I)=0.0
C      * I MIDPOINT LOOP
   DO 400 I=1,L1
C      * J ELEMENT LOOP
C   J1 IS THE COORDINATE COUNTER
C   J IS THE ELEMENT COUNTER
   J1=0
   N1=0
   IF (FLG23 .GT. 0) CALL NOTS
   DO 110 K=1,NB
      M1=N1+1
      N1=N1+ND(K)=1
   DO 100 J=M1,N1
      J1=J1+1
   IF (FLG18 .GT. 0 .AND. J .GT. NMA .OR. FLG20 .GT. 0)
      * COMPUTE X,Y,Z MATRICES
      CALL XYZ
   CONTINUE
   DO 107 J=1,L1
      IF (NLF(K),GT,0) GO TO 110
      IF (BON,EQ,0) GO TO 105
      DO 103 J = M1, N1
         VN(I,K) = VN(I,K)+AXV(J)
   103 VT(I,K) = VT(I,K)+AYV(J)
   104 GO TO 110
   105 DO 106 J = M1, N1
         VN(I,K) = VN(I,K)+AXV(J)*SINA(I) - AYV(J)*COSA(I)
   106 VT(I,K) = VT(I,K)+AXV(J)*COSA(I) + AYV(J)*SINA(I)
   110 J1=J1+1
   IF (FLG60 .LE. 0 .OR. FLG15 .LE. 0) GO TO 118
C*** * PRINT STRIP VORTEX MATRICES
C   IF( I .EQ. 1 .AND. BON .EQ. 0 ) WRITE(6,111)
   IF( I .EQ. 1 .AND. BON .EQ. 1 ) WRITE(6,112)
111 FORMAT(1H1,31H STRIP VORTEX MATRICES ON BODY //)
112 FORMAT(1H1,31H STRIP VORTEX MATRICES OFF BODY //)
   WRITE(6,114) ( AXV(J),J=1,NT)
   WRITE(6,115) ( AVV(J),J=1,NT)
114 FORMAT(1H0,5H ROW,14/9H X MATRIX / (6E20.7) )
115 FORMAT(1H0,5H ROW,14/9H Y MATRIX / (6E20.7) )
116 IF (RON)120,210,120
   C     * SAVE X,Y,Z ON TAPE * OFF BODY POINTS
C*** ***SAVE X,Y,Z ON TAPE * OFF BODY POINTS
C*** ***AXISYMMETRIC FLOW * TAPE 9
C*** ***CROSS FLOW * TAPE 10
C*** ***EXTRA CROSS FLOW * TAPE 8
120 IF (IEC,EQ,=1)GO TO 125
   122 WRITE(8) (ECX(J),J=1,NT), (ECY(J),J=1,NT), (ECZ(J),J=1,NT)
      IF (IEC) 125,400,125
125 IF (IAC) 140,130,130
   130 WRITE (9) (AX(J),J=1,NT), (AY(J),J=1,NT), (AZ(J),J=1,NT)
      IF (IAC) 400,140,400
   140 WRITE (10) (CX(J),J=1,NT), (CY(J),J=1,NT), (CZ(J),J=1,NT)
      GO TO 400
C*** ***SAVE ON TAPE * ON BODY
C*** ***AXISYMMETRIC FLOW * TAPE 9
C*** ***CROSS FLOW * TAPE 10
C*** ***EXTRA CROSS FLOW * TAPE 8
C*** ***IEC = #1 MEANS NO EXTRA CROSS FLOW
210 IF (IEC,EQ,=1) GO TO 240
220 DO 230 J = 1,NT
   A(J) = -ECX(J) * SIN(A(I)) + ECY(J) * COS(A(I))
230 B(J) = ECX(J) * COS(A(I)) + ECY(J) * SIN(A(I))
      WRITE (8) (A(J),J=1,NT), (B(J),J=1,NT), (ECZ(J),J=1,NT)
      IF (IEC) 240,400,240
240 IF (IAC) 310,250,250
250 DO 260 J=1,NT
   A(J) = AX(J) * SINA(I) + AY(J) * COSA(I)
260 B(J) = AX(J) * COSA(I) + AY(J) * SINA(I)
WRITE (9) (A(J),J=1,NT),(B(J),J=1,NT),(AZ(J),J=1,NT)
270 IF (TAC) 400,310,400
310 DO 320 J=1,NT
   A(J) = CX(J)*SINA(I)*CY(J)*COSA(I)
320 B(J) = CX(J)*COSA(I)*CY(J)*SINA(I)
WRITE (10) (A(J),J=1,NT),(B(J),J=1,NT),(CZ(J),J=1,NT)
400 CONTINUE
IF (FLG15,LE,0) GO TO 1400
IF (BN0,NF,0) GO TO 1200
C*** ***ON BODY
READ (4)
C
C*** * IF FLG23 .GT. 0 INPUT NNU MUST BE NONE, HENCE NNU = 0 HERE
C
IF (NNU,LE,0) GO TO 600
DO 500 I = 1, NNU
   READ (4) MSF,(A(J),J=1,NT),(B(J),J=1,NT)
500 WRITE (5) MSF,(A(J),J=1,NT),(B(J),J=1,NT)
REWIND 3
REWIND 4
READ (4)
600 N=NSIGA=1
   IF (FLG16,GT,1) N=NSIGA
C*** ***N = 0 MEANS 1 RHS ONLY NO NON-UNIFORM FLOW
C*** * IF FLG23 .GT. 0 INPUT NNU MUST BE NONE, HENCE N = 0 HERE
C
IF (NF,FD,0) GO TO 800
DO 700 I = 1, N
   READ (3) MSF,(A(J),J=1,NT),(B(J),J=1,NT)
700 WRITE(4) MSF,(A(J),J=1,NT),(B(J),J=1,NT)
800 M=0
C*** * SKIP PRESCRIBED VORTEX INPUTS ON 4 So THAT STRIP VORTEX
C*** * SUMMATIONS CAN GO BEHIND IT
C
   IF (FLG23,GT,0) READ(4)
DO 900 J = 1, NB
IF (NLF(J), GT, 0) GO TO 900
NSIGA = NSIGA + 1
NU = NU + 1
WRITE (4) M, (VN(I, J), I = 1, NT), (VT(I, J), I = 1, NT)
900 CONTINUE
C
C*** = SINCE NO NU IS INPUT WITH FLG23 GT 0, NSIGC IS MAX OF 1 AND M
C*** = SHOULD BE 0 IF FLG17 LE 0. DONT USE FLG17 WITH FLG23
C
IF (FLG23, LE, 0) GO TO 975
C
C*** = RING WING OPTION = FORM COLUMN (PARTLY) FOR PRESCRIBED VORTICITY
C*** = RHS
C
IBOD = 0
DO 950 J = 1, NB
IF (NLF(J), GT, 0) GO TO 950
IBOD = IBOD + 1
C
C*** = CONVERT (ON BODY) X,Y TO NORMAL, TANGENTIAL
C
DO 925 I = 1, NT
VAN(I) = VAN(I) + VA(I, IBOD)*SINA(I) + VR(I, IBOD)*COSA(I)
925 VAT(I) = VAT(I) + VA(I, IBOD)*COSA(I) + VR(I, IBOD)*SINA(I)
WRITE (4) (VAN(I), I = 1, NT), (VAT(I), I = 1, NT)
950 CONTINUE
C
M = NSIGC = 1
975 IF (FLG17, GT, 0) M = NSIGC
C
IF (M, LE, 0) GO TO 1100
DO 1000 I = 1, M
READ (3) M$F, (A(J), J = 1, NT), (B(J), J = 1, NT)
1000 WRITE (4) M$F, (A(J), J = 1, NT), (B(J), J = 1, NT)
1100 REWIND 3
GO TO 1400
C*** ***OFF BODY
1200 DO 1300 J = 1, NB
   IF (NFL(J),GT,0) GO TO 1300
   WRITE(4) (VN(I,J), I = 1,L1), (VT(I,J), I = 1,L1)
1300 CONTINUE
   IF (FLG23 .LE. 0) GO TO 1400
   IBOD = 0
   DO 1350 J = 1, NB
      IF (NFL(J),GT,0) GO TO 1350
      IBOD = IBOD + 1
      WRITE(4) (VA(I,IBOD), I = 1,L1), (VR(I,IBOD), I = 1,L1)
1350 CONTINUE
C      * TEST IF OFF BODY COMPLETED
C      * TEST IF OFF BODY
1400 IF (FLG05.EQ.0.OR.BON.NE.0.) GO TO 1600
C      * INITIAL FOR OFF BODY * TMFN REF=ENTER I,J LOOPS
BON=1,
L1=ND(NB+1)
DO 1500 I = 1, L1
   X2(I) = XP(I)
1500 Y2(I) = YP(I)
   GO TO 60
1600 REWIND 9
   REWIND 8
   REWIND 10
   REWIND 4
   RETURN
   END
SUBROUTINE XYZ

* CONTROL FOR X,Y,Z MATRICES COMPUTATION

COMMON /D1, D3, XMXJ, YMYJ, XMXJPI, YMYJPI, 8
COMMON HEDR(10), CASE, NNU, NNU
1 , FLG03, FLG04, FLG05, FLG06, FLG07
2 , FLG08, FLG09, FLG10, FLG11, FLG12
3 , FLG13, FLG14, FLG15, FLG16, FLG17
4 , FLG18, FLG19, FLG20, FLG21, FLG22
5 , FLG23, FLG24, FLG25, FLG26, FLG27
COMMON NT, ND(11), M, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNC(5), TYPEC(5), NLF(11), IEC, NSIGEC,
3 TYPFEC(5), NUNEC(5)

DOUBLE PRECISION HEDR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1 , FLG08, FLG09, FLG10, FLG11, FLG12
2 , FLG13, FLG14, FLG15, FLG16, FLG17
3 , FLG18, FLG19, FLG20, FLG21, FLG22
4 , FLG23, FLG24, FLG25, FLG26, FLG27
REAL MN
LOGICAL PF

COMMON /RNNGWG/ VA(100,2), VR(100,2), VANC(100), VAT(100)
COMMON /CL/ X(100), Y(100), Z(100), YZ(100), DELS(100),
1 SINA(100), COSA(100), XN(100), YP(100)
2 , XWAKE(11), YWAKE(11)
COMMON /TL/ A(100), B(100), AX(100), AV(100), AZ(100),
1 CX(100), CY(100), CZ(100), XV(100), YV(100),
2 VN(100,5), VT(100,5), BQN, IAC,
3 I, J, J1, SJ, DS,
4 DX, DY, DI, XJ, YJ,
5 XK, EKK, EK, K, PF
C    * START
   IF (BUN) 100,10,100
   10 IF (J=1) 110,20,110
C    * J EQUAL I PATH
   20 T1=.5*DELS(J)
      SJ=T1/Y2(J)
      IF (SJ<0.08) 30,30,40
   30 CALL XYZ1
      GO TO 1000
   40 SJ<0.08
      CALL XYZ1
      NI=33
      T2=.08*Y2(J)
      DS=(T1=T2)/32.8
      DX=DS*COSA(J)
      DY=DS*SINA(J)
      XJ=X2(J)+T2*COSA(J)=DX
      YJ=Y2(J)+T2*SINA(J)=DY
      CALL XYZ2
      GO TO 300
C    * INITIAL Y COORDINATE MID-POINT FOR ZERO TEST
   100 YZERO=Y2(I)=0.000001
C    * J NOT EQUAL I PATH
C    * COMPUTE MINIMUM DISTANCE TO I MIDPOINT
   110 J1P1 = J1 + 1
      XMIX = X2(I) = X1(J1)
      YMYJ = Y2(I) = Y1(J1)
      XMIXP1 = X2(I) = X1(J1P1)
      YMYJP1 = Y2(I) = Y1(J1P1)
      D1 = XMIX**2 + YMYJ**2
      D2=(X2(I)=X2(J))**2+(Y2(I)=Y2(J))**2
      D3 = XMIXP1**2 + YMYJP1**2
      S = SQRT( (X1(J1P1) = X1(J1) )**2 + ( Y1(J1P1) = Y1(J1) )**2 )
      IF (D1>D2) 130,150,120
   120 IF (D2>D3) 150,150,140
130 IF (D1=D3) 160, 160, 140
140 DM=SQRT(D3)
   GO TO 170
150 DM=SQRT(D2)
   GO TO 170
160 DM=SQRT(D1)
   C * COMPUTE NO. OF INTERVALS(NI) AND DELTA S (DS)
   C FOR SIMPSON RULE INTEGRATION
170 IF (DM.EQ.0,0) GO TO 200
      NI=5*DELS(J)/DM+0.9
      IF (NI) 180, 180, 190
180 NI=3
      DS=DELS(J)/2.
      GO TO 220
190 NI=NI+NI
      IF (NI=128) 210, 200, 200
200 NI=129
      DS=DELS(J)/128.
      GO TO 220
210 XNI=NI
      DS=DELS(J)/XNI
      NI=NI+1
220 DX=DS*COSA(J)
      DY=DS*SINA(J)
300 XJ=X1(J1)+DX
      YJ=Y1(J1)+DY
      CALL XYZ2
1000 RETURN
   END
SUBROUTINE XYZ1

* COMPUTE X, Y, Z MATRICES FOR SJ LESS THAN OR EQUAL .08

COMMON HEDR(10), CASE, NB, NNU
1     FLG03, FLG04, FLG05, FLG06, FLG07
2     FLG08, FLG09, FLG10, FLG11, FLG12
3     FLG13, FLG14, FLG15, FLG16, FLG17
4     FLG18, FLG19, FLG20, FLG21, FLG22
5     FLG23, FLG24, FLG25, FLG26, FLG27

COMMON NT, ND(11), MN, NUNA(5), TYPEA(5)
1 NER1, NER2, NMA, NSIGA, NSIGC
2 NUNC(5), TYPEC(5), NLF(11), IEC, NSIGEC
3 TYPEC(5), NUNEC(5)

DOUBLE PRECISION HEDR, CASE

INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1 FLG08, FLG09, FLG10, FLG11, FLG12
2 FLG13, FLG14, FLG15, FLG16, FLG17
3 FLG18, FLG19, FLG20, FLG21, FLG22
4 FLG23, FLG24, FLG25, FLG26, FLG27

COMMON /RNGWNG/ VA(100,2), VR(100,2), VAN(100), VAT(100)
COMMON /ECF/ ECX(100), ECY(100), ECZ(100)

REAL MN

LOGICAL PF

COMMON /CL/ X1(100), Y1(100), X2(100), Y2(100), DELS(100)
1 SINA(100), COSA(100), XP(100), YP(100)
2 XWAKE(11), YWAKE(11)

COMMON /TL/ A(100), B(100), AX(100), AY(100), AZ(100)
1 CX(100), CY(100), CZ(100), AXV(100), AYV(100),
2 VN(100,5), VT(100,5), BON, IAC
3 I, J, J1, SJ, DS
4 DX, DY, NI, XJ, YJ
5 XK, FEK, EKK, K, PF
* START

* INITIALIZE

T1=SJ*8J
T2=ALOG(SJ/8.0)
T3=SINA(J)*SINA(J)
T4=T2*T3
T5=.6666667*T3
T6=T5*T3
T7=SJ*SJ
T8=T7*T7
T9=6.283185*COSA(J)
T10=6.283185*SINA(J)
T11=T1*SJ
T14 = .3333333 * (16.0 + 6.0 * T3) + 2.0 * T2
IF (IEC.EQ.*1) GO TO 15
C*** FXTA CROSS FLOW 1ST TERM OF X(I,I), Y(I,I), Z(I,I)
10 ECX(J) = 6.283185 * SINA(J) + 2.0 * SINA(J) * COSA(J) * SJ
ECY(J) = 6.283185 * COSA(J) + SJ * T14
ECZ(J) = -8.0 * (1.666667 + T2) * SJ
IF (IEC) 15,1000,15
15 IF (PF) GO TO 25
IF (IAC) 30,20,20
C*** AXIS FLOW
20 AX(J)=T10+SINA(J)*COSA(J)*(T7+(T4+2.16667)*T11/12.)
T12=T1+T1
AZ(J)=Y2(J)*T8*(1.+T2+T1*(2.+T12+3.*T2*(1.+T12)))/144.
25 IF (IAC) 30,30,100
C*** CROSS FLOW
30 T13=T1/16.
CX(J)=T10+2.*SINA(J)*SJ*COSA(J)*(T13*(T5+T2+T2))
CY(J)=T9+T7*(2.*T4+13*(T1-4.777778 +T3+T6+T2*(3.+7.666667 * T3))
CZ(J)=T8*(1.+T2+T13*(T1111111*T3+T2*(T5-1.)))
100 IF (PF) GO TO 200
IF (FLG15 .LE. 0, OR, NLKEF(K), GT, 0) GO TO 1000
200 AXV(J) = T9 + T7 * (T2 + T3) + T11 * (T2 * (12 * T3 - 9))
1
IF (.NOT., PF) GO TO 1000
AX(J) = AXV(J)
AY(J) = AYV(J)
C
C *** RING WING OPTION PV EFFECTS ON ITSELF NEGLECTS 2*PI TERMS
C
IF (FLG23 .LE. 0) GO TO 1000
AX(J) = AX(J) = T9
AY(J) = AY(J) = T10
AYV(J) = AY(J)
AXV(J) = AX(J)
1000 CONTINUE
RETURN
END
SUBROUTINE XYZ2

* COMPUTE X,Y,Z MATRICES USING SIMPSON RULE INTEGRATION

REAL L2J0
COMMON /D/ R1SQR, R2SQR, XMXJ, YMYJ, XMXJP1, YMYJP1, S
COMMON /MEDR(10), CASE, NNU
1  FLG03, FLG04, FLG05, FLG06, FLG07
2  FLG08, FLG09, FLG10, FLG11, FLG12
3  FLG13, FLG14, FLG15, FLG16, FLG17
4  FLG18, FLG19, FLG20, FLG21, FLG22
5  FLG23, FLG24, FLG25, FLG26, FLG27
COMM NTH, ND(11), MN, NUNA(5), TYPEA(5),
1  NER1, NER2, NMA, NSIGA, NSIGE,
2  NUNC(5), TYPEC(5), NLF(11), IEC, NSIGEC,
3  TYPEC(5), NUNEC(5)

DOUBLE PRECISION MEDR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1  FLG08, FLG09, FLG10, FLG11, FLG12
2  FLG13, FLG14, FLG15, FLG16, FLG17
3  FLG18, FLG19, FLG20, FLG21, FLG22
4  FLG23, FLG24, FLG25, FLG26, FLG27
COMM /ECF/ ECX(100), ECY(100), ECZ(100)
COMM /RNGNG/ VA(100,2), VR(100,2), VAN(100), VAT(100)
DATA NSW /1/

*** RSMALL WILL BE TRUE IF IS .LT. EPS AND THEREFORE SMALL
LOGICAL RSMALL
REAL MN
LOGICAL PF

COMMON /CL/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1  SINA(100), COSA(100), XP(100), YP(100)
2  XWAKE(11), YWAKE(11)
COMMON /TL/ A(100), B(100), AX(100), AY(100), AZ(100),
1  CX(100), CY(100), CZ(100), AXV(100), AXV(100), AYV(100),
2  AZV(100), AZV(100),
2 VN(100, 5), VT(100, 5), BUN, IAC,
3 I, J, J1, S0, OS,
4 DX, DY, N1, XJ, YJ,
5 XK, EK, EKK, K, PF

C
* START

C
* INITIALIZE

EPS = 0.0
ASSIGN 570 TO K1

C*** ***K5 = 80 FOR NON-SMALL ELEMENT AXISYMMETRIC
ASSIGN 80 TO K5

C*** ***K6 = 295 FOR NON SMALL ELEMENT CROSS FLOW
ASSIGN 295 TO K6

IF (F(=E15, LE.0, OR, NLF(K), GT.0) GO TO 15

10 ASSIGN 420 TO K1
15 S2 = .6666667 * DS
S1 = .3333333 * DS
S3 = 8.0/3.0 * S1
S5 = .3333333 * S1
S4 = S2 + S2
T1 = Y2(I) * Y2(I)
ASSIGN 28 TO K2
ASSIGN 410 TO K3
ASSIGN 570 TO K4
IF (.NOT. PF ) GO TO 12
ASSIGN 110 TO K2
ASSIGN 420 TO K3
ASSIGN 560 TO K4

12 IF (I, NE, J), OR, (BON, NE, 0.0) ) GO TO 16

C*** ***J = J ** ON BODY
R = DELS(I) / 2.0
RSMALL = (R / Y2(I)) *LT. EPS
NSW = 2
GO TO 17

16 R = SQRT( AMAX1(R1SQR, R2SQR) )
IF( ABS(Y2(I)) .LT. 10E-30) GO TO 13
RSMALL = ( X / Y2(I)) .LT. EPS
GO TO 17
13 RSMALL = .FALSE.
17 IF( .NOT. RSMALL) GO TO 19
C*** **SMALL ELEMENT -- FORM XIJ2D, YIJ2D, LIJ
C
C*** **K5 = 105 FOR SMALL ELEMENT AXISYMMETRIC
   ASSIGN 105 TO K5
C*** **K6 = 320 FOR SMALL ELEMENT CROSS FLOW
   ASSIGN 320 TO K6
C*** **NSW = 1 FOR I NE J
C*** **NSW = 2 FOR I EQ J 1ST TIME THROUGH
C*** **NSW = 3 FOR I EQ J 2ND TIME THROUGH
   GO TO (14, 21, 22), NSW
C
C*** **I = J 1ST TIME THROUGH
   21 XLEFT = XJ + DX
   YLEFT = YJ + DY
   J1P1 = J + 1
   XRIGHT = X1(J1P1)
   YRIGHT = Y1(J1P1)
C*** **GET NSW READY FOR I = J 2ND TIME THROUGH
   NSW = 3
   GO TO 23
C*** **I = J 2ND TIME THROUGH
   22 XLEFT = X1(J1)
   YLEFT = Y1(J1)
   XRIGHT = XLEFT + 32.0 * DX
   YRIGHT = YLEFT + 32.0 * DY
   NSW = 1
C*** **CALCULATE QUANTITIES WHICH HAVE NOT YET BEEN CALCULATED FOR I=4
   23 XMIX = X2(I) = XLEFT
   YMYJ = Y2(I) = YLEFT
   XMIXP1 = X2(I) = XRIGHT
YMYJP1 = Y2(I) - YRIGHT
RLLEFT = R
RRIGHT = R
S = SQRT((XLEFT - XRIGHT)**2 + (YLEFT - YRIGHT)**2)
GO TO 11
C*** **1 NE J SMALL ELEMENT
14 RLFL = R1SQR
RRIGHT = R2SQR
C*** **2 NOW FORM XIJ2D, YIJ2D, LIJ
11 H = (-XMXJ * SINA(J)) + (YMYJ * CUSAJ(J))
FL1 = (XMXJ * CUSAJ(J)) + (YMYJ * SINA(J))
EL2 = (XMXJP1 * CUSAJ(J)) + (YMYJP1 * SINA(J))
DPHIDX = ALNLOG(RLEFT / RRIGHT)
IF(ARS(H/EL1) .LT. 100.0E-10) GO TO 7
DPHIDY = -2.0 * (ATAN(EL1/H) = ATAN(EL2/H))
GO TO 8
7 DPHIDY = 0.0
IF(EL1 * EL2 .LT. 0.0) DPHIDY = -6.283186
8 XIJ2D = (COSA(J)*DPHIDX) = (SINA(J)*DPHIDY)
YIJ2D = (SINA(J)*DPHIDX) + (COSA(J)*DPHIDY)
LIJ2D = ( (EL1 + EL2) / 4.0 ) * DPHIDX
1 + (S/4.0) * ALNLOG((RLEFT * RRIGHT) / (4096.0 + T2**2))
2 - = 8 = (H/2.0) * DPHIDY
C
* NO. OF INTERVAL LOOP
19 DO 1000 IS=1,NI
XJ=XJ+DX
YJ=YJ+DY
T2=YJ*YJ
T3=X2(I)+XJ
T4=T3+T3
T5=(Y2(I)+YJ)**2
T6=T4+T5
T7=SQRT(T6)
T8=T2+T4
T9A = Y2(I) = YJ

XYZ2 106
XYZ2 107
XYZ2 108
XYZ2 109
XYZ2 110
XYZ2 111
XYZ2 112
XYZ2 113
XYZ2 114
XYZ2 115
XYZ2 116
XYZ2 117
XYZ2 118
XYZ2 119
XYZ2 120
XYZ2 121
XYZ2 122
XYZ2 123
XYZ2 124
XYZ2 125
XYZ2 126
XYZ2 127
XYZ2 128
XYZ2 129
XYZ2 130
XYZ2 131
XYZ2 132
XYZ2 133
XYZ2 134
XYZ2 135
XYZ2 136
XYZ2 137
XYZ2 138
XYZ2 139
XYZ2 140

XYZ2
T9 = T9A**2
T10 = T9 + T4
T10A = SORT(T10)

*** IF DENOM (T8) IS ZERO THEN MAKE T21 FAIL ALL TESTS

IF (ABS(T8) .LT. 10.0E=30) GO TO 29
T21 = SORT(T1 / T8)
GO TO 27

29 T21 = 0, 10

* COMPUTE ELLIPIC INTEGRAL

IF (RSMALL .AND. FLG21 .EQ. 0) GO TO 18
XX = 4.0 * YJ * Y2(I) / T6
CALL ELIP
IF (IEC) 10, 575, 18

18 IF (IAC) 200, 20, 20

* AXIS FLOW

20 IF (RSMALL) GO TO 25
T11 = YJ / T7
IF (T21 .LT. 0.01) GO TO 24
T12 = YJ / Y2(I)
FV2 = (EKK + EEK * (T1 = T8) / T10) / T7
FV3 = Y2(I) / T10 * T3 / T7 * EEK
F1 = FV3 * T12
F2 = FV2 * T12
FV4 = FV2 * T3 / Y2(I)
F3 = T11 * EEK
GO TO 26

24 FV2 = 0.
FV3 = 0.
FV4 = 0.

*** SMALL Y FORMULAS AXISYMMETRIC FLOW

T23 = T1 / T8**2
T24 = 2.0 * T4 = T2
F1 = ( ( 1.570796 * YJ * T3 ) / ( T8**1.5 ) ) *
1 \(1.0 + (0.75 	imes (3.0 	imes T2 + 2.0 \times T4) \times T23)\)

\(F2 = (1.570796 \times YJ \times Y2(I)) \times (T24 / (T8 \times 0.25))\)

\(F3 = 1.570796 \times YJ \times (1.0 + (0.25 \times T23 \times (-T24))) / \text{SQR}(T8)\)

26 GO TO 26

25 \(T32 = T3 / T10A\)

\(T33 = T9A / T10A\)

\(T34 = T33 \times 2\)

\(T35A = T10A / (8.0 \times Y2(I))\)

\(T35 = \text{ALOG}(T35A)\)

\(T36 = T9A \times Y2(I)\)

\(T40 = T10A / Y2(I)\)

\(T37 = (T40 \times 2) \times 0.125\)

\(T38 = 0.250 \times T36 \times T35\)

\(T39 = 0.125 \times T36\)

\(T34A = 2.0 \times T34\)

\(T34B = T34A \times 3.0\)

\(F1 = (-2.0 \times T32 \times ((-T35A + T35) - (0.5 \times T33))\)

\(1 = (T40 / 16.0) \times T34B\) / Y2(I)

\(F2 = ((0.25 \times T36 \times T35) = T34 = 1.0 = (T39 \times T34B) / Y2(I)\)

\(F3 = (T35 \times (T36 \times (0.25 \times T36 \times 2) + T37) \times T36 + T37)\)

C * SIMPSON RULE INTEGRATION

20 IF (IS=1) 30, 30, 40

C * FIRST PASS

30 AXS=F1

AYS=F2

AZS=F3

IA=0

GO TO 110

40 IF (IS=0 \& \& N1) GO TO 75

50 IF (IA) 70, 60, 70

C * EVEN PASS

60 AXS=AXS+4.0*F1

AYS=AYS+4.0*F2

AZS=AZS+4.0*F3
IA=1
GO TO 110
C ** ODD PASS
70 AXS=AXS+F1+F1
AYS=AYS+F2+F2
AZS=AZS+F3+F3
IA=0
GO TO 110
75 GO TO K5, (80,105)
C ** LAST PASS
80 IF (J=I) 100,90,100
90 IF (BON .NE. 0,0) GO TO 100
AX(J)=AX(J)=84*(AXS+F1)
AY(J)=AY(J)=82*(AYS+F2)
AZ(J)=AZ(J)=84*(AZS+F3)
GO TO 110
100 AX(J)=S4*(AXS+F1)
AY(J)=S2*(AYS+F2)
AZ(J)=S4*(AZS+F3)
GO TO 110
C*** **LAST PASS ** SMALL ELEMENT
105 IF (J .NE. I) .OR. (BON .NE. 0,0) )GO TO 107
C*** **I = J ** ON BODY
AX(J) = AX(J) + XIJ2D + (AXS + F1) * S1
AY(J) = AY(J) + YIJ2D + (LIJ2D/ Y2(I) ) + (AYS + F2) * S1
AZ(J) = AZ(J) = (2,0 * LIJ2D) + (AZS + F3) * S1
GO TO 110
C*** **I NE J ** ON OR OFF BODY
107 AX(J) = XIJ2D + (AXS + F1) * S1
AY(J) = YIJ2D + (LIJ2D/ Y2(I) ) + (AYS + F2) * S1
AZ(J) = -2,0 * LIJ2D + (AZS + F3) * S1
110 IF (IAC) 200,200,400
C ** CROSS FLOW
200 IF (RSMALL)GO TO 223
IF (T21 .LT. 0.04)GO TO 220
T12 = T1 + T4
F1 = T3/Y2(I)*(EKK*EEK*T12/T10)/T7
F2 = (EKK*T8*TA+T1*(T4-T2))/T10=EKK*T8)/T1/T7
F3 = T7*(EKK*T12/T6=EEK)/T1
GO TO 230

C*** *** SMALL Y FORMULAS*** CROSS FLOW

220 T25 = T1 / T8**2
T29 = ( 1.570796 * T2 ) / ( T8**1.5 )
T26 = 4.0 * T4 = T2
T31 = T26 * T23
F1 = ( (-4.712369) * T2 * T3 * Y2(I) ) / ( T8**2.5 )
F2 = T29 * ( 1.0 - (1.125 * T31 ) )
F3 = T29 * ( 1.0 - (0.375 * T31 ) )
GO TO 230

C*** *** IAC LT 0 MEANS NO AXISYMMETRIC FLOW

223 IF(IAC)<225,227,227

C*** *** CALCULATE SMALL ELEMENT QUANTITIES THAT DID NOT GET CALCULATED
C*** *** BECAUSE THERE WAS NO AXISYMMETRIC FLOW

225 T32 = T3 / T10A
T33 = T9A / T10A
T34 = T33**2
T35A = T10A / (8.0 * Y2(I) )
T35 = ALOG(T35A)
T36 = T9A/Y2(I)
T40 = T10A / Y2(I)
T37 = (T40**2)*0.125
T38 = 0.250*T36*T35

C*** *** CALCULATE SMALL ELEMENT F1,F2,F3 CROSS FLOW

227 T39A = -5.0 * T38
T40A = T40**2
T36A = T36**2
F1 = (T32 / Y2(I)) * ( (-0.75 * T40 + T35) + T33
1 + (0.125 * T40 * (2.0 * T34 =5.0) )
F2 = ( T38A + T34 + 3.0 + (0.25 * T36 * (T34 = 6.50) ) ) / Y2(I)
F3 = ( ( T36 = (0.375 * T40A) = (0.25 * T36A) ) * T35 =4.0 *T36

XYZ2 246
XYZ2 247
XYZ2 248
XYZ2 249
XYZ2 250
XYZ2 251
XYZ2 252
XYZ2 253
XYZ2 254
XYZ2 255
XYZ2 256
XYZ2 257
XYZ2 258
XYZ2 259
XYZ2 260
XYZ2 261
XYZ2 262
XYZ2 263
XYZ2 264
XYZ2 265
XYZ2 266
XYZ2 267
XYZ2 268
XYZ2 269
XYZ2 270
XYZ2 271
XYZ2 272
XYZ2 273
XYZ2 274
XYZ2 275
XYZ2 276
XYZ2 277
XYZ2 278
XYZ2 279
XYZ2 280
1  -(0.50 * T36A) = T37 ) / Y2(I)

C  * SIMPSON RULE INTEGRATION

230 IF (IS = 1) 740, 740, 250

C  * FIRST PASS

240 CXS=F1
CYS=F2
CZS=F3
IC=0
GO TO 400

250 IF (IS = NI) 260, 290, 260

260 IF (IC) 280, 270, 280

C  * EVEN PASS

270 CXS=CXS+4, *F1
CYS=CYS+4, *F2
CZS=CZS+4, *F3
IC=1
GO TO 400

C  * ODD PASS

280 CXS=CXS+F1+F1
CYS=CYS+F2+F2
CZS=CZS+F3+F3
IC=0
GO TO 400

290 GO TO K6, (295, 320)

C  * LAST PASS

295 IF (J, NE, I) GO TO 310

300 IF (RON, NE, 0,0) GO TO 310

CX(J)=CX(J)+S2*(CXS+F1)
CYS(J)=CYS(J)+S2*(CYS+F2)
CZ(J)=CZ(J)+S2*(CZS+F3)
GO TO 400

310 CX(J)=S2*(CXS+F1)
CYS(J)=S2*(CYS+F2)
CZ(J)=S2*(CZS+F3)
GO TO 400
320 IF (I,NE,J) OR (BON,NE,0,0) GO TO 340
C*** *** LAST PASS SMALL ELEMENT I=J ON BODY
   CX(J) = CX(J) + XIJ2D + (CXS + F1) * S1
   CY(J) = CY(J) + YIJ2D + (LIJ2D / Y2(I)) + (CYS + F2) * S1
   CZ(J) = CZ(J) + (2.0 * LIJ2D / Y2(I)) + (CZS + F3) * S1
GO TO 400
C*** *** I NE J OR ANY OFF BODY
340 CX(J) = XIJ2D + (CXS + F1) * S1
   CY(J) = YIJ2D + (LIJ2D / Y2(I)) + (CYS + F2) * S1
   CZ(J) = -(2.0 * LIJ2D / Y2(I)) + (CZS + F3) * S1
C*** *** K3 = 420 FOR SURFACE VORTICITY PF TRUE
   400 GO TO K3, (410, 420)
C*** *** K1 = 420 FOR STRIP VORTEX
   410 GO TO K1, (570, 420)
C*** *** FLOW OF CONTROL REACHES HERE FOR (PF=TRUE) OR ((FLG15 GT 0 AND
C*** *** NLFL EQ 0 (LIFTING BODY)) AND (I NE J ON BODY OR ANY OFF BODY)
C*** 420 IF (RSMALL) GO TO 542
   FV1 = (T2-T1) / T7 * EEK / T10
   IF (IS,GT,1) GO TO 440
C* FIRST PASS
   AX1 = FV1
   AX2 = FV2
   AV1 = FV3
   AV2 = FV4
   IV = 0
   GO TO 570
C*** 440 IF (IS,EQ,NI) GO TO 500
   IF (IV) 460, 450, 460
C* EVEN PASS
450 AX1 = AX1+4,*FV1
   AX2 = AX2+4,*FV2
   AV1 = AV1+4,*FV3
   AV2 = AV2+4,*FV4
   IV = 1
   GO TO 570
**UNO PASS**

460 AX1 = AX1 + FV1 + FV1
AX2 = AX2 + FV2 + FV2
AY1 = AY1 + FV3 + FV3
AY2 = AY2 + FV4 + FV4
IV = 0
GO TO 570

**LAST PASS**

500 IF (J = 1) 540, 520, 540
520 IF (BONNEO) GO TO 540
AXV(J) = AXV(J) = S4*(AX1 + FV1) = S2*(AX2 + FV2)
AYV(J) = AYV(J) = S4*(AY1 + FV3) + S2*(AY2 + FV4)
GO TO 550
540 AXV(J) = S4*(AX1 + FV1) = S2*(AX2 + FV2)
AYV(J) = S4*(AY1 + FV3) + S2*(AY2 + FV4)
GO TO 550
542 T34C = T34B = 8.0
FV1 = (T38 + (T32*2) = (T39 * T34C) / Y2(I)
FV2 = (T32 / Y2(I)) * ((-0.75 * T40 * T35) + T33
1 + (0.125 * T40 * T34C)
IF (IS_0GT_1) GO TO 544

**FIRST PASS SMALL ELEMENT**

AX1 = FV1
AY1 = FV2
IV = 0
GO TO 570
544 IF (IS_EQ_NI) GO TO 548
IF (IV_EQ_0) GO TO 546

**EVEN PASS SMALL ELEMENT**

AX1 = AX1 + 4.0* FV1
AY1 = AY1 + 4.0* FV2
IV = 1
GO TO 570

**ODD PASS SMALL ELEMENT**

546 AX1 = AX1 + FV1 + FV1
AY1 = AY1 + FV2 + FV2
IV = 0
GO TO 570
C*** ***LAST PASS SMALL ELEMENT
  548 IF ( (I,NE,J) OR, (ROW = NE, 0,0) ) GO TO 549
C*** ***I EQ J OR ANY OFF BODY
  549 AXV(J) = AXV(J) + YIJ2D + (LIJ2D / Y2(I) ) + (AX1 + FV1)*S1
  AYV(J) = AYV(J) + XIJ2D + (AY1 + FV2)*S1
GO TO 550
C*** ***I NE J OR ANY OFF BODY
  549 AXV(J) = -YIJ2D + (LIJ2D / Y2(I) ) + (AX1 + FV1)*S1
  AYV(J) = XIJ2D + (AY1 + FV2)*S1
C*** ***K4 = 560 FOR SURFACE VORTICITY PF TRUE
  550 GO TO K4,(560,570)
C*** ***FLOW OF CONTROL REACHES HERE IF PF IS TRUE
  560 AX(J) = AXV(J)
  AY(J) = AYV(J)
  570 IF (ICE.EQ.,-1) GO TO 1000
  575 IF (T21,LT,0.08)GO TO 595
  580 T20 = SQRT(T2 / (T1 + T4) )
  IF (T20,LT,0.01) GO TO 590
  T13 = YJ * Y2(I)**3
  T14 = T1 + T8
  T15 = T2 + T1
  T16 = T14 * T14
  T17 = T1 * YJ
  T18 = T1 * T1
  T19 = T8 * T8
  F3 = ( T7/T13 ) * ( (=-T14) * EFK + ( (T16 - T15) * EKK) / T6 )
  F1 =(T3 / (T17 *T7) ) * ( (EFK / T10) ) * (T16 = 3.0 * T15) =
  1 * (T14 * EKK) )
  TEMP1 =((B.0*T8**3) = (12.0* T1*T19) + (26.0*T15*T8)
  1 + (2.0*T18*(2.0*T1 =5.0*T2) )* EFK/T10
  TEMP2 = EKK * ( (8.0*T19) + (4.0*T1*T8) - (2.0*T15) = (4.0*T18) )
  F2 = (TEMP1 + TEMP2) / ( T13 * T7)
GO TO 630

C*** ***SMALL YJ FORMULAS * EXTRA CROSS FLOW

590 T25 = YJ**3
T30 = T4 + T1
T27 = T30**3.5
T28 = T25 * Y2(I)
F1 = ( 2.945243 * T25 * T3 * T1 ) / T27
F2 = 9.068584 * T28 * ( 3.0 * T1 - 7.0 * T4 ) / T27
F3 = 1.767146 * T28 / (T30**2.5)
GO TO 630

C*** ***SMALL Y FORMULAS * EXTRA CROSS FLOW

595 T25 = YJ**3
F1 = ( 2.945243 * T25 * T3 * T1 ) / ( T8**3.5 )
F2 = ( (-14.13717) * T25 * Y2(I) ) / (T8**2.5)
F3 = -F2 / 8.0

C*** ***SIMPSON#S RULE

630 IF (IS = 1) 640,640,650

C*** ***FIRST PASS

640 FCXS = F1
ECYS = F2
ECZS = F3
IE = 0
GO TO 1000

650 IF (IS = NI) 660,690,660

660 IF ( IE ) 680,670,680

C*** ***FVEN PASS

670 FCXS = ECXS + 4.0 * F1
ECYS = ECYS + 4.0 * F2
ECZS = ECZS + 4.0 * F3
IE = 1
GO TO 1000

C*** ***ODD PASS

680 ECXS = ECXS + F1 + F1
ECYS = ECYS + F2 + F2
ECZS = ECZS + F3 + F3

XYZ2
IE = 0
GO TO 1000

C*** ***LAST PASS

690 IF (J - I) 710, 700, 710

C*** ***T=J    *   ELEMENTS ON MAIN DIAGONAL

700 IF (BON, NE, 0, 0) GO TO 710
  ECX(J) = ECX(J) + S4 * ( ECXS + F1 )
  ECY(J) = ECY(J) + S5 * ( ECYS + F2 )
  ECZ(J) = ECZ(J) + S3 * ( ECZS + F3 )
GO TO 1000

C*** ***OFF MAIN DIAGONAL OR OF OFF BODY POINTS

710 ECX(J) = -S4 * ( ECXS + F1 )
  ECY(J) = -S5 * ( ECYS + F2 )
  ECZ(J) = S3 * ( ECZS + F3 )

1000 CONTINUE
  RETURN
  END
SUBROUTINE ELIP

* HASTINGS APPROXIMATION FOR ELLIPTIC INTEGRALS

COMMON HEDR(10), CASE, NB, NNU
1 ,FLG03, FLG04, FLG05, FLG06, FLG07
2 ,FLG08, FLG09, FLG10, FLG11, FLG12
3 ,FLG13, FLG14, FLG15, FLG16, FLG17
4 ,FLG18, FLG19, FLG20, FLG21, FLG22
5 ,FLG23, FLG24, FLG25, FLG26, FLG27

COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNEC(5), TYPEC(5), NLFI(11), IEC, NSIGEC,
3 TYPEC(5), NUNEC(5)

DOUBLE PRECISION HEDR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1 ,FLG08, FLG09, FLG10, FLG11, FLG12
2 ,FLG13, FLG14, FLG15, FLG16, FLG17
3 ,FLG18, FLG19, FLG20, FLG21, FLG22
4 ,FLG23, FLG24, FLG25, FLG26, FLG27

REAL MN
LOGICAL PF

COMMON /CL/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1 SINA(100), COSA(100), XP(100), YP(100)
2 XWAKE(11), YWAKE(11)

COMMON /TL/ A(100), B(100), AX(100), AY(100), AZ(100),
1 CX(100), CY(100), CZ(100), AXV(100), AYV(100),
2 VN(100,5), VT(100,5), BUN, IAC,
3 I, J, J1, SJ, DS,
4 DX, DY, NI, XJ, YJ,
5 XK, EEX, EKK, K, PF

* START

FTA = 1, = XK
IF (FTA) 20,20,40
20 WRITE (6,30) ETA
30 FORMAT (1H0,3AH,*** ERL0lK IN SUBROUTINE ELIP * ETA= F15.8)
                  WRITE(6,800) T,XJ,DX,YJ,DY,XZ(I),YZ(I),XK
800 FORMAT(1H1,15,7F15.6)
ETA = 0.000005
40 ELN=ALN(ETA)
        FKK = 1*386294E0 + FTA * (966634E1 + ETA * (3590092E-1 + ETA * (6451196E-1)))
        ELN = 5 + ETA * (1249859E0 + ETA * (18880249E-1 + ETA * (3328355E-1)))
4        FKK = 1*386294E0 + FTA * (966634E1 + ETA * (3590092E-1 + ETA * (6451196E-1)))
1        ELN = (ETA * (9473784E-1 + ETA * (6296601E-1)) + ETA * (4249837E0 + ETA * (9200180F1) + ETA * (2664496F2))))
RETURN
FNN

ELIP 036
ELIP 037
ELIP 038
ELIP 039
ELIP 040
ELIP 041
ELIP 042
ELIP 043
ELIP 044
ELIP 045
ELIP 046
ELIP 047
ELIP 048
ELIP 049
ELIP 050
ELIP 051
ELIP 052
ELIP 053
<table>
<thead>
<tr>
<th>COMMON</th>
<th>S/RMNG/A</th>
<th>VR(100,2)</th>
<th>VAN(100)</th>
<th>VAT(100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>FLG03</td>
<td>FLG04</td>
<td>FLG05</td>
<td>FLG06</td>
</tr>
<tr>
<td></td>
<td>FLG13</td>
<td>FLG14</td>
<td>FLG15</td>
<td>FLG16</td>
</tr>
<tr>
<td></td>
<td>FLG18</td>
<td>FLG19</td>
<td>FLG20</td>
<td>FLG24</td>
</tr>
<tr>
<td></td>
<td>FLG23</td>
<td>FLG24</td>
<td>FLG25</td>
<td>FLG26</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/CASE</th>
<th>MFR(10)</th>
<th>FLG07</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FLG09</td>
<td>FLG12</td>
<td>FLG17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/NFR</th>
<th>MFR(2)</th>
<th>FLG11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FLG08</td>
<td>FLG10</td>
<td>FLG12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/NLFL</th>
<th>NLF(1)</th>
<th>FLG10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FLG13</td>
<td>FLG14</td>
<td>FLG20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/NICE</th>
<th>NICE(1)</th>
<th>FLG15</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FLG18</td>
<td>FLG19</td>
<td>FLG20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/FLG</th>
<th>MFLGC</th>
<th>FLG20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FLG23</td>
<td>FLG24</td>
<td>FLG25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COMMON</th>
<th>/FLG</th>
<th>/FLG</th>
<th>MFLGC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FLG25</td>
<td>FLG26</td>
<td>FLG27</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REAL</th>
<th>MN, KAY</th>
<th>LOGICAL</th>
<th>P</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**FOLLOWING ARE 3 ARITHMETIC FUNCTIONS**

\[
\text{ARCF}(Z, \text{SMALLR}, \text{BIGR}) = \arcsin \left( \frac{Z}{\sqrt{Z^2 + (\text{SMALLR} - \text{BIGR})^2}} \right)
\]

\[
\text{BETAF}(Z, \text{SMALLR}, \text{BIGR}) = \arcsin \left( \frac{Z}{\sqrt{Z^2 + (\text{SMALLR} - \text{BIGR})^2}} \right)
\]

\[
\text{AKAYF}(Z, \text{SMALLR}, \text{BIGR}) = \arcsin \left( \frac{Z}{\sqrt{Z^2 + (\text{SMALLR} + \text{BIGR})^2}} \right)
\]

---

**FOR 100 IBOD = 1, FLG14**

- \(Z = X2(1) = \text{YAKE}(\text{IBOD})\)
- \(\Omega_{\text{GA}} = \text{OMEG}(Z, Y2(1), \text{YAKE}(\text{IBOD}))\)
- \(\text{BETA} = \text{BETAF}(Z, Y2(1), \text{YAKE}(\text{IBOD}))\)
- \(\text{KAY} = \text{AKAYF}(Z, Y2(1), \text{YAKE}(\text{IBOD}))\)
- \(\text{CALL QC}(\Omega_{\text{GA}}, \Omega_{\text{M}}, \Omega)\)

---

**SMALLR IS Y2(I)**

**BIGR IS YAKE(100)**

**SMALLR GT BIGR**

\[
\text{BIGK} = \left( Z \left( \sqrt{Y2(I) \times \text{YAKE}(\text{IBOD})} \right)^2 \right) \times \text{QM} \]

**SMALLR LE BIGR**

\[
\text{BIGK} = 3.141593 + \left( Z \left( \sqrt{Y2(I) \times \text{YAKE}(\text{IBOD})} \right)^2 \right) \times \text{QM} + 1
\]

**NOTE THAT VA AND VR WILL NOT YET BE MULTIPLIED BY \(\text{DGAMMA}/\text{DZ}\)**

**WHICH IS REALLY THE INPUT PRESCRIBED VORICITY**

---

**NOTES**

- 036
- 037
- 038
- 039
- 040
- 041
- 042
- 043
- 044
- 045
- 046
- 047
- 048
- 049
- 050
- 051
- 052
- 053
- 054
- 055
- 056
- 057
- 058
- 059
- 060
- 061
- 062
- 063
- 064
- 065
- 066
- 067
- 068
- 069
- 070
C 40 YALT(1,1) = RIGK / 6.283185
100 YALT(1,1) = SQRT(YWAKE(1BD)) / Y2(I)) ) / 6.283185
RETURN
FND
FUNCTION HLABM (BETA, K)

C THIS SURROUNTE CALCULATES THE HEUMAN'S LAMBDA FUNCTION US BETTA AND K

DOUBLE PRECISION A, F, E

REAL K

DATA TWOPI/0.6366197724/

CALL INEL (FI, EI, PI, BETA, BETA, 1.0 - K * K, 0, 1, 1)

A = 1.0 - K ** 2

CALL ELLC (A, F, E, 1)

CALL ELLC (A, F, F, 2)

HLAMB = TWOPI * (F * EI + (E * F) * FI)

RETURN

END
SUBROUTINE QC(OMEG, QM, Q)
   THIS SUBROUTINE CALCULATES THE LEGENDRE FUNCTIONS OF THE SECOND KIND
   AND HALF ORDER, THE ARGUMENTS ARE:
   OMEG ARGUMENT FOR WHICH LEGENDRE FUNCTIONS WILL BE FOUND
   QM VALUE OF LEGENDRE FUNCTION OF MINUS ONE HALF ORDER
   Q VALUE OF LEGENDRE FUNCTION OF PLUS ONE HALF ORDER
   DOUBLE PRECISION OMEG, ARG, A, F, E, QMD, QD
   OMEG=OMEG
   ARG=2.0/(OMEGD+1.0)
   A=1.0-ARG
   CALL ELLC (A, F, E, 1)
   CALL ELLC (A, F, E, 2)
   QMD=F*ARG**0.5
   QD=E*(2.0*(OMEG+1.0)**0.5+OMEGD)*QMD
   QM=QMD
   Q=QD
   RETURN
END
SUBROUTINE ELLC(A,K,E,I)
C THIS SUBROUTINE CALCULATES THE ASSOCIATED COMPLETE ELLIPTIC INTEGRALS
C OF THE FIRST OR SECOND KIND
C THE ARGUMENTS ARE
C A ARGUMENT (K SQUARED) FOR WHICH EX OR K# WILL BE FOUND
C K VALUE OF ASSOCIATED COMPLETE ELLIPTIC INTEGRAL OF FIRST KIND
C E VALUE OF ASSOCIATED COMPLETE ELLIPTIC INTEGRAL OF SECOND KIND
C I IF EQ 1, COMPUTE K; IF EQ 2, COMPUTE E
C DOUBLE PRECISION K,E,CON(32),A,LN4,CF(29),CL(3),DLOG
C DOUBLE PRECISION CON(32),CF(29),CL(3)
C EQUIVALENCE (CON,CF),(CON(30),CL)
C DATA CF /9,6573590797569016D-2,3,088573486752694D-2,1,4978988178
1704629D=2,9,6587579861753113D-3,1,1208918554464092D-2,1,3855601247
215656D=2,6,6905509906897936D-3,6,499844332939018D-4,1,249998999411
37923D=1,7,031242646647361D-2,2,8818058565403952D-2,3,70683893415
45422D=2,2,718986111678625D-2,1,4105380776158048D+2,2,141318927862
5886D=3,1,504918176360183D-4,4,4341718112155860E+1,5,680565784795
6352D=2,2,1876220647186198D-2,1,2510592410844644D+2,1,3034146073731
7437D=2,1,5377102528552019D-2,7,3356164974290365D-3,7,980964089987
8229D=4,2,4999999993617622D-1,9,37499202498680113D=2,5,8582839536559
9024D=2,4,23828074569179D=2,2,3,03274772412848D+2 /
C DATA CL /1,5525129948040721D-2,3,4838679435896492D=3,1,642721079
17048025D=4 /
C LN4 = 1,3829436111989D
C IF (A,EQ.0,0)
C GO TO 1,2,1
C 1 K = LN4 + (((CON(8)*A+CON(7))*A+CON(6))*A+CON(5))*A+CON(4)*A
1+CON(3))*A+CON(2))*A+CON(1))*A = DLOG(A)+0.5+(((CON(16)*A+2CON(15))*A+CON(14))*A+CON(13))*A+CON(12))*A+CON(11))*A+CON(10))*A
3+CON(9))*A)
C GO TO 3
C 2 E = 1,0000+(((CON(24)*A+CON(23))*A+CON(22))*A+CON(21))*A+CON(20
1))*A+CON(19))*A+CON(18))*A+CON(17))*A = DLOG(A)+(((CON(32)*A
2+CON(31))*A+CON(30))*A+CON(29))*A+CON(28))*A+CON(27))*A+CON(26))*A
3A+CON(25))*A)
3 RETURN
C 4 K = 0.999999999D30
C 4 K = 0.999999999F30
C  F = 1.000
C  F = 1.0E0
C  RETURN
C END
SUBROUTINE ELINT3(XKSQ, XN, PHI, PIE)
C THIS SUBROUTINE CALCULATES THE INCOMPLETE ELLIPTIC INTEGRAL OF THE
C THIRD KIND. THE ARGUMENTS ARE:
C XKSQ VALUE OF K SQUARED
C XN VALUE OF MINUS ALPHA SQUARED
C PHI VALUE OF PHI
C PIE VALUE OF INCOMPLETE ELLIPTIC INTEGRAL OF THIRD KIND
101 FORMAT (7E16.8)
DATA HP/1.570796/
DATA ROUND/.0000050/
XKSQ=XKSQ
XN=XN
PHI=PHI
IF (XN.EQ.1.0.AND.XKSQ.EQ.1.0) GO TO 50
IF (XKSQ.GT.1.0) GO TO 48
IF (XN.LT.1.0) GO TO 48
IF (PHI.GT.1.0) GO TO 48
C NORMALIZE PHI
A=1.
P=P
GOTO 3
A=1.
B=1.
BB=1.
IF (ABS(P/1.570796).LE.10.0) GO TO 10
IF (P.GT.10.0) GO TO 10
J=P/(2.*P)
XX=2.*J
P1=P-XX*P
P=HP
B=1.
GOTO 10
D=SUM
B=0.
IF (P1.GT.1.0) GO TO 7,8
6  P=P1
   XXX=1,
   GOTO11
7  PIF=(XX+1,)*A*D
   GOTO47
8  XXX=1,
   XX=XX+2,
   P=P1*HP=P1
   GOTO11
9  PIF=A*(XX*D+XXX*SUM)
   GOTO47
10 IF(SK.EQ.,1.) GOTO48
    IF(FN.EQ.,1.) GO TO 48
11 IF(P.GT.,10,E=4)GOTO13
    IF(FN.GT.,0.)GOTO12
    SUM=P
   GOTO45
12 RRT=SQRT(FN)
   SUM=ATAN(P*RRT)/RRT
   GOTO45
13 S=SIN(P)
   S2=S*S
   C=COS(P)
   IF(SK.GT.,64)GOTO20
    IF(ABS(FN).GE.,6)GOTO15
C
POWER SERIES IN N AND K SQUARED
14 SA=1,
   SB=SK/2,
   CB=S*C
   CA=P
   FM=0,
   SUM=P
   X=SUM*1,E=8
   SA=SR=SA*FN
   CA=(-CB/(2.*FM+1,))+(1.,-5./(FM+1,))*CA
ELNT
Y = SA * CA
SUM = SUM + Y
IF((SB * CA) .GE. X) GO TO 141
IF(ABS(Y) .LT. X) GO TO 45
141 FM = FM + 1,
CB = CB * 92
SB = (1.0 / (FM + 1.0)) * SK * SB
GOTO 14
C POWER SERIES IN K SQUARED
15 PK = SK
RT = SQRT(1.0 + FN)
IF(RT .NE. 0.0) GO TO 16
GM = B / C
GOTO 18
16 IF(C.GT.4.0.E=3) GOTO 17
GM = (HP - (C / (RT * 3))) / RT
GOTO 18
17 GM = ATAN(RT * 8 / C) / RT
18 G1 = GM * 1.0 * E = 0
E = P
FM = S * C
H = 1.0
SUM = G
FM = 0.0
19 GM = (E - G) / FN
H = H * (1.0 - 0.5 / (FM + 1))
G2 = H * G * PK
SUM = SUM + G2
IF(G2 .LE. G1) GOTO 45
FM = FM + 1,
E = F / (2.0 * FM) + (1.0 - 0.5 / FM) * E
F = F + 32
PK = PK + SK
GOTO 19
20 SK = 1.0 * SK
IF(S LT C) GO TO 32
   ADDITION FORMULA
21  ZP=SQR(1.-SK*SK)
   RT=SQR(ABS(FN*(FN+1.)*((FN+SK)))
   SST=(1.-ZP)/(SK*(C+1.))
   XP=(SST*S*R1)/(1.+FN*SK=FN*SST*C*ZP)
   IF(FN)22,29,25
22  IF(RT NE 0.) GO TO 24
   R=S/(C+1.)
   IF(FN NE (-1.)) GO TO 23
   CF=(Z*R=SK*ST*S=(S/C)*ZP)/SKP
   GOTO30
23  CF=(SST*(S=(2.*R))/S*C/ZP)*(SK/SKP)
   GOTO30
24  IF(FN*(FN+SK) LT C) GO TO 26
25  CF=(FN/RT1) ATAN(XP)
   GOTO30
26  IF(ABS(XP) GE 1.) GOTO27
   YX=XP**2
   YX=2.*XP*(1.++YX*(1./3.++YX*(2.*YX/7.)))
   GOTO28
27  YX=ALOG((1.+XP)/(1.-XP))
28  CF=(FN*YX)/(2.*RT1)
   GOTO30
29  CF=0.
30  BB=-1.,
   S=SQR(SST)
   C=SQR(1.-SST)
   GOTO32
31  SUM=2.*SUM=CF
   GOTO45
32  U=S/C
   V=1. /C
   T=U*V
   W=U*2
IF(SGT0,1)GOT033
R=S*(1.0+S2*(1.0+S2*(1.0+S2/7.0)))
GOT034
33 P=ALPH(U+V)
34 D=1.0+FN
IF(DGT1,1)GOT037
C POWER SERIES IN 1/N AND 1 = (K SQUARED)
35 CA=1.0
CB=0.5*SKP
AL=(T+R)/2.0
REMUI*V**3
FM=0.0
SUM=AL
T1=SUM*1.0E6
36 CA=D*CA+CB
AL=(BE=(2.0*FM+1.0)*AL)/(2.0*(FM+2.0))
X=CA*AL
SUM=SUM+X
IF(ABS(X),LT,T1)GOT044
FM=FM+1.0
CB=((2.0*FM+1.0)/(2.0*(FM+1.0)))*CB*SKP
IF(ABS(BE),LT,10.0E-30) BE=0.0
BE=BE*W
GOT036
C POWER SERIES IN 1 = (K SQUARED)
37 RT=SQRT(ABS(FN))
38 IF(FN)38,39,40
38 Q=ALOG((1.0+RT*S)/(1.0-RT*S))/(2.0*RT)
GOT041
39 Q=S
GOT041
40 Q=ATAN(RT*S)/RT
41 SUM=FN*Q*RT
PKP=SKP
AP=0.5
FM = 1,
T1 = SUM + 1.0 * E = 8
42  Q = (R - Q) / 0
R = T / (2.0 * FM) - (1.0 * 5 / FM) * R
X = AP * (FN + Q + R) * PKP
SUM = SUM + X
IF (ABS(X) .LT. T1) GOTO 43
T = T + W
PKP = PKP + SKP
FM = FM + 1,
AP = AP * (1.0 + 5 / FM)
GOTO 42
44  IF (AB, LT, 0.) GOTO 31
45  IF (B) 5, 9, 46
46  PI = A * SUM
47  PI = PIE + PIE * ROUND
RETURN
C  ERROR RETURN
48  PI = 0.
GOTO 47
C  CASE OF PI(1,1,PHI)
50  PI = 0.5 * (TAN(P) / COS(P) + ALOG(TAN((HP + P) / 2.0)))
GO TO 47
END
SUBROUTINE INEL (F,E,PI,A,PHI,SKI,K3,K2,K1)
C THIS SUBROUTINE CALCULATES THE INCOMPLETE ELLIPTIC INTEGRALS OF THE
C FIRST, SECOND AND THIRD KINDS. THE ARGUMENTS ARE:
C F VALUE OF INCOMPLETE ELLIPTIC INTEGRAL OF THE FIRST KIND
C F VALUE OF INCOMPLETE ELLIPTIC INTEGRAL OF THE SECOND KIND
C PI VALUE OF INCOMPLETE ELLIPTIC INTEGRAL OF THE THIRD KIND
C A VALUE OF ALPHA SQUARED
C PHI VALUE OF PHI
C SKI VALUE OF K SQUARED
C K3 IF EQ 0, DO NOT COMPUTE PI ; IF NE 0, COMPUTE PI
C K2 IF EQ 0, DO NOT COMPUTE E ; IF NE 0, COMPUTE E
C K1 IF EQ 0, DO NOT COMPUTE F ; IF NE 0, COMPUTE F
C
DQUBLE PRECISION ARG,FD,ED
DATA PI/1.57079633/;
F=0.0
F=0.0
IF (K3,EQ,0) GO TO 220
CALL ELINT3 (SKI,=A,PHI,PI)
220 IF (K1,EQ,0) GO TO 240
IF (ABS(PHI-PI),GT,10.0**(-7)) GO TO 230
ARG=1.0-SKI
CALL ELLC (ARG,FD,ED,1)
F=FD
GO TO 240
230 CALL ELINT3 (SKI,0.0,PHI,F)
240 IF (K2,EQ,0) GO TO 260
IF (ABS(PHI-PI),GT,10.0**(-7)) GO TO 250
ARG=1.0-SKI
CALL ELLC (ARG,FD,ED,2)
E=ED
GO TO 260
250 CALL ELINT3 (SKI,=SKI,PHI,E)
E=(1.0-SKI)*F+0.5*SKI*SIN(2.0*PHI)/SQRT(1.0-SKI*SIN(PHI)**2)
260 RETURN
END
OVERLAY(AXSY, 3, 0)
PROGRAM PREP
SURROUTINE PREP
C **
* PREPARE TAPES 3 AND 11 FOR USE BY LINK 5 (MATSO)
COMMON/SPACER/WKAREA(5000)
DIMENSION TEMP(105), Y2(100)
COMMON HEDR(10), CASE, NB, NNU
1, FLG03, FLG04, FLG05, FLG06, FLG07
2, FLG08, FLG09, FLG10, FLG11, FLG12
3, FLG13, FLG14, FLG15, FLG16, FLG17
4, FLG18, FLG19, FLG20, FLG21, FLG22
5, FLG23, FLG24, FLG25, FLG26, FLG27
COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNA(5), TYPEE(5), NL(11), IEC, NSIGEC,
3 TYPEC(5), NUNFC(5)
C DOUBLE PRECISION HEDR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1, FLG08, FLG09, FLG10, FLG11, FLG12
2, FLG13, FLG14, FLG15, FLG16, FLG17
3, FLG18, FLG19, FLG20, FLG21, FLG22
4, FLG23, FLG24, FLG25, FLG26, FLG27
REAL MN
DIMENSION COSSOR(100), RHS(100)
DIMENSION A(105), R(100, 5), FF(100), T(100)
DATA FOURPI/12.5663706/
C*** ***AXISYMMETRIC FLOW ONLY MS = 0
C*** ***CROSS FLOW ONLY MS = 1
C*** ***EXTRA CROSS FLOW ONLY MS = 2
C*** ***AXISYMMETRIC AND CFS FLOW MS = 3
C*** ***AXISYMMETRIC AND EXTRA CFS FLOW MS = 4
C*** ***CROSS AND EXTRA CFS FLOW MS = 5
C*** ***AXISYMMETRIC, CROSS, AND EXTRA CFS FLOW MS = 6
NCK1=0
NCK2=0
NCK3=0
NCK4=0
NCK5=0
NCK6=0
IF (FLG12 .EQ. 0 .OR. (FLG04 .EQ. 0 .AND. FLG21 .EQ. 0)) GO TO 3
IF (FLG05 .EQ. 0) GO TO 4
C***  ***SKIP OFF BODY COORDINATES
READ (12)
4  NI=NT+NB
  READ (12) (TEMP(I),I = 1,NI),(TEMP(I),I = 1,NI)
  (TEMP(I),I = 1,NT),(Y2(I),I = 1,NT)
  REWIND 12
3  REWIND 3
2  IF (FLG03) 5,000,5
C ***  * PREPARE AXIASSYMMETRIC MATRIX TAPE (3)
5  IF (FLG19 .GT. 0) GO TO 2000
  IF (FLG22 .GT. 0) GO TO 255
  K = 0
  L = NT+NSIGA
  READ (4) (A(I),I=1,NT),(FF(I),I=1,NT)
  IF (FLG16 .NE. 0) GO TO 20
  K = K+1
  DO 10 I = 1, NT
10  R(I,K) = A(I)
20  IF (NNU) 60,60,30
30  DO 50 J = 1, NNU
  READ (4) MS,(A(I),I=1,NT)
  IF (MS .EQ. 1 .OR. MS .EQ. 2) GO TO 50
  K = K+1
  DO 40 I = 1, NT
40  R(I,K) = A(I)
50  CONTINUE
60  IF (FLG14 .LE. 0) GO TO 290
  NR = NMA+1
  READ (4) (R(I,1),I=NR,NT)
PREP 036
PREP 037
PREP 038
PREP 039
PREP 040
PREP 041
PREP 042
PREP 043
PREP 044
PREP 045
PREP 046
PREP 047
PREP 048
PREP 049
PREP 050
PREP 051
PREP 052
PREP 053
PREP 054
PREP 055
PREP 056
PREP 057
PREP 058
PREP 059
PREP 060
PREP 061
PREP 062
PREP 063
PREP 064
PREP 065
PREP 066
PREP 067
PREP 068
PREP 069
PREP 070
REWIND 4
DO 220 I = NR, NT
220 R(I,1) = R(I,1)=FF(I)
IF (FLG14,EQ, NR) GO TO 245
DO 240 I = 1, NT
READ (9) (A(J),J=1,NT)
A(N1+1) = R(I,1)
240 WRITE (3) (A(J),J=1,L)
245 DO 250 I = NR, NT
READ (9) (A(J),J=1,NT), (A(J),J=1,NT)
A(N1+1) = R(I,1)
250 WRITE (3) (A(J),J=1,L)
C PRESCRIBED TANGENTIAL VELOCITY
INPUT TO SOLVIT ON TAPE 3
C OUTPUT FROM SOLVIT ON TAPE 3
C TAPES 1 AND 2 ARE SCRATCH TAPES
CALL SOLVIT(WKAREA,NT,NSIGA,5000,3,1,2,3,NCK1)
IF(NCK1, EQ, 1) GO TO 9010
251 REWIND 9
GO TO 800
C*** ***AXISYMMETRIC FLOW * GENERATED (RESEP) BOUNDARY CONDITIONS
C*** ***NPR1 = THE NUMBER OF ELEMENTS ON BODY 1
C*** ***NPR2 = THE NUMBER OF ELEMENTS ON BODY 2
255 NPR1 = ND(1) = 1
NPR2 = ND(2) = 1
NSIGA = 3
NSIGC = 1
NSIGFC = 1
L = NT + NSIGA
C*** *** IS THE TOTAL WIDTH OF THE MATRIX FOR AXISYMMETRIC FLOW INCL.
C*** *** RIGHT HAND SIDES
READ (4)
READ(4) ( COSSQR(I),I = 1,NPR1), (RHS(I),I = 1,NPR1 )
REWIND 4
DO 260 I = 1, NPB1
R(I,1) = 0,0
260 END
R(T, 2) = 1.0

260 R(T, 3) = COSQ(I)
NREGIN = NPA1 + 1
NEND = NPB1 + NPA2
DO 265 I = NREGIN, NEND
R(T, 1) = 1.0
R(T, 2) = 0.0
265 R(T, 3) = 0.0

290 REWIND 4
ASSIGN 400 TO M
IF (FLG12 .NE. 0) ASSIGN 300 TO M
DO 700 I = 1, NT
GO TO M, (300, 400)

300 READ (9) (A(J), J=1, NT), (A(J), J=1, NT), (A(J), J=1, NT)
GO TO 500

400 READ (9) (A(J), J=1, NT)
500 DO 600 J = 1, NSIGA
K = NT + J
600 A(K) = R(I, J)
700 WRITE (3) (A(J), J=1, L)
C AXISYMMETRIC FLOW INPUT TO SOLVIT ON TAPE 3
C OUTPUT FROM SOLVIT ON TAPE 3
C TAPES 1 AND 2 ARE SCRATCH TAPE
CALL SOLVIT(WKAREA, NT, NSIGA, 5000, 3, 1, 2, 3, NCK2)
IF (NCK2 .NE. 1) GO TO 9020

701 REWIND 9
C ** * PREPARE CROSSFLOW MATRIX TAPE (11)
C ** * SKIP SINA * READ COSA
800 IF (FLG04, EQ, 0) GO TO 1610
K = 0
L = NT + NSIGA
IF (FLG22, EQ, 0) GO TO 910
READ (4) (A(I), I=1, NT), (A(I), I=1, NT)
IF (FLG17, NE, 0) GO TO 820
K = K + 1
DO 810 I = 1, NT
810 R(I,K) = A(I)
820 IF (NNU) 900, 900, 830
830 DO 850 J = 1, NNU
     READ (4) MS,(A(I),I=1,NT)
     IF (MSEQ.0,0,0,0) GO TO 850
     K = K+1
     DO 840 I = 1, NT
840 R(I,K) = -A(I)
850 CONTINUE
900REWIND 4
     GO TO 1000
C*** ***CROSS FLOW  * GENERATED (RESLP) BOUNDARY CONDITIONS
910 DO 920 I = 1, NPB1
920 R(I,1) = -RHS(I)
     DO 930 I = NBEGIN, NEND
930 R(I,1) = 0,0
1000 ASSIGN 1300 TO M
     IF (FLG12,NE,0) ASSIGN 1200 TO M
     DO 1600 I = 1, NT
     GO TO M, (1200,1300)
1200 READ (10) (A(J),J=1,NT),(A(J),J=1,NT),(A(J),J=1,NT)
C*** ***FORM PHI MATRIX FROM THEMA (CROSS FLOW) MATRIX
     DO 1250 J = 1, NT
1250 A(J) = Y2(I)*A(J)
     GO TO 1400
1300 READ (10) (A(J),J=1,NT)
1400 DO 1500 J = 1, NSIGC
     K = NT+J
1500 A(K) = R(I,J)
1600 WRITE (11) (A(J),J=1,L)
C CROSS FLOW  INPUT TO SOLVIT ON TAPE 11
C OUTPUT FROM SOLVIT ON TAPE 3
C TAPES 1 AND 2 ARE SCRATCH TAPES
     CALL SOLVIT(NKAREA,NT,NSIGC,5000,11,1,2,3,NCK3)
IF(NCK3,EQ.,1) GO TO 9030

1605 REWIND 10
1610 CONTINUE

C**  ***EXTRA CROSS FLOW
    REWIND 11
    IF (FLG21,EQ.,0.AND.,FLG22,EQ.,0) RETURN
    K = 0
    L = NT + NSIGEC.
    IF (FLG22,GT.,0) GO TO 1800

C**  ***EXTRA CROSS FLOW  *  NON=UNIFORM FLOW ONLY
C**  ***SKIP RECORD WITH SINES AND COSINES
    READ (4)
    DO 1650 J=1,NNU
    READ(4) MS, (A(I),I=1,NT)
    IF (MS,LT.2,OR,MS,EQ.3) GO TO 1650
    K = K+1
    DO 1640 I = 1,NT
    1640 R(I,K) = A(I)
    1650 CONTINUE
    GO TO 1900

C**  ***EXTRA CROSS FLOW  *  GENERATED (RESEP) BOUNDARY CONDITIONS
1800 DO 1820 I = 1,NPB1
1820 R(T,1) = COSSQR(I)
    DO 1840 I = NBEGIN,NEND
    1840 R(I,1) = 0,0
1900 REWIND 4

C**  ***IS 1920  *  SOLVE A MATRIX
    ASSIGN 1920 TO M
C**  ***IS 1940  *  SOLVE POTENTIAL MATRIX
    IF (FLG12,NE.,0) ASSIGN 1940 TO M
    DO 1980 I = 1,NT
    GO TO M, (1920,1940)

C**  ***SOLVE A MATRIX
1920 READ (8) (A(J),J =1,NT)
    GO TO 1960
1940 READ (B) (A(J),J=1,NT), (A(J),J=1,NT), (A(J),J=1,NT)
C*** **FORM PHI MATRIX FROM THETA (EXTRA CROSS FLOW) MATRIX
DO 1950 J = 1,NT
1950 A(J) = Y2(I) * A(J) / 2.0
1960 DO 1970 J = 1,NSIGEC
K = NT + J
1970 A(K) = R(I,J)
1980 WRITE (11) (A(J),J=1,L)
C*** **EXTRA CROSS FLOW INPUT TO SOLVIT ON TAPE 11
C*** **OUTPUT FROM SOLVIT ON TAPE 3
C*** **TAPE 1 AND 2 ARE SCRATCH TAPES
CALL SOLVIT (*KAREA,NT,NSIGEC,5000,11,1,2,3,NCK4)
IF(NCK4 .EQ. 1) GO TO 9040
1985 REWIND 8
REWIND 11
C RETURN
GO TO 9070
2000 IF(FLG23 .GT. 0) GO TO 3000
NR = NT = NMA
L = NMA+1
READ (4) (R(I,1),I=1,NMA)
READ (4) (FF(I),I=1,NR)
DO 2100 I = 1, NR
2100 FF(I) = FF(I)/FOURPI
BACKSPACE 4
WRITE (4) (FF(I),I=1,NR)
REWIND 4
DO 2300 I = 1, NMA
READ (9) (A(J),J=1,NMA), (T(J),J=1,NR)
DO 2200 J = 1, NR
2200 R(I,1) = R(I,1) - T(J)*FF(J)
A(L) = R(I,1)
2300 WRITE (3) (A(J),J=1,L)
C PRESCRIBED VORTICITY INPUT FOR SOLVIT ON TAPE 3
C OUTPUT FROM SOLVIT ON TAPE 3
C TAPFS 1 AND 2 ARE SCRATCH TAPES
   CALL SOLVIT(WKAREA,NMA,L = NMA,5000,3,1,7,3,NCK51)
   IF(NCK5 ,EQ, 1) GO TO 9000
2500 REWIND 9
   GO TO 800
3000 NR = NT = NMA
   NMAP1 = NMA + 1
C*** * CALCULATE THE NUMBER OF RHS
C
   LL = 0
   DO 3100 I=1,NB
       IF( NLF(I),GT, 0 )GO TO 3100
   LL = LL + 2
3100 CONTINUE
   L = NMAP1 + LL
C
C*** * READ SINS FOR STREAMFLOW RHS
C
   READ(4) ( R(I,1),I=1,NMA )
C
C*** * READ INPUT PRESCRIBED VORTICITIES
C
   READ(4) ( FF(I),I=NMAP1,NT )
   WRITE(6,8001) (FF(I),I=NMAP1,NT)
   8001 FORMAT(1H1,* THE INPUT PV ARE */(6E20,7))
   DO 3125 I = NMAP1,NT
   3125 FF(I) = FF(I) / (=FOURIPI)
C
C*** * READ STRIP VORTEX RHS
C
   LLD2P1 = LL/2 + 1
   DO 3150 J=2,LLD2P1
      3150 READ(4)MS,( R(I,J),I=1,NMA )
C
C*** * IPV IS BODY NUMBER OF 1ST PRESCRIBED VORTICITY BODY
C

IPV = NB + FLG14 + 1
JBOD = LLD2P1
NN = NMA

C

DO 3300 KCNT = IPV, NB
JBOD = JBOD + 1
NN = NN + ND(KCNT) = 1

C*** * READ COLUMN OF RHS CALCULATED BY NOTS FORMULA
C
READ(4) (R(ICNT,JBOD),ICNT = 1,NMA)

C*** * MULTIPLY NOTS COLUMN BY LAST PRESCRIBED VORTICITY ON THAT BODY
C
DO 3300 ICNT = 1, NMA
3300 R(ICNT,JBOD) = R(ICNT,JBOD) * FF(NN) * (-FOURPI)

C
REWIND 4
DO 3400 I=1, NMA
NEND = NMA
JBOD = LLD2P1
READ(9) ( A(J), J=1, NMA), ( T(J), J=NMAP1:NT )

C

DO 3350 KCNT=IPV, NB
JBOD = JBOD + 1
NBEG = NEND + 1
NEND = NEND + ND(KCNT) = 1

C*** * SUM PV VORTEX ELEMENTS * INPUT PV AND ADD TO NOTS RHS
C

DO 3350 NCNT=NREG, NEND

C*** WHEN COMING OFF UNIT 9, THE VORTEX ELEMENTS( T(J) ) ARE STILL
C*** WHILE NOTS COLUMNS COMING OFF UNIT 4 ARE RHS, THE TWO SHOULD
C *** ADDED TO FORM A COMPLETE RHS, BUT THEY ARE SUBTRACTED SINCE TH
C *** OF T(J) MUST BE CHANGED
C
3350 R(I,JBD)= R(I,JBD) - T(NCNT) * FF(NCNT)
C
C *** * ATTACH ALL RHS FOR ROW NUMBER I
C
3375 LRHS = 0
DO 3375 ICNT=NMAP1,L
  LRHS = LRHS + 1
3375 A(ICNT) = R(I,LRHS)
C
3400 WRITE(3)(A(J),J=1,L)
C
C*** * RING WING OPTION INPUT FOR SOLVIT ON TAPE 3
C*** * OUTPUT FOR SOLVIT ON TAPE 3
C
CALL SOLVIT(WKAREA,NMA,L=NMA,5000,3,1,7,3,NCK6)
  IF(NCK6 .eq. 1) GO TO 9050
3500 REWIND 9
  GO TO 800
9000 WRITE(6,9001)
9001 FORMAT(61H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR PRESCRIBED VORT
  ICITY)
  GO TO 9080
9010 WRITE(6,9011)
9011 FORMAT(71H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR PRESCRIBED TANG
  ENTIAL VELOCITY)
  GO TO 9080
9020 WRITE(6,9021)
9021 FORMAT(51H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR AXISYMMETRIC FL
  LOW)
  GO TO 9080
9030 WRITE(6,9031)
9031 FORMAT(51H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR CROSS FLOW)
  GO TO 9080
9040 WRITE (6,9041)
9041 FORMAT (51H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR EXTRA CROSS FL
10w)
   GO TO 9080
9050 WRITE(6,9051)
9051 FORMAT(51H NOT ENOUGH SPACE RESERVED IN SOLVIT FOR RING WING )
9070 CONTINUE
9080 STOP
   END
SUBROUTINE SOLVIT (A, ND, MD, KD, NI, MM, NO, NW, NCK)

DIRECT MATRIX SOLUTION

WRITTEN BY J. L. HESS * PROGRAMMED BY T. M. RIDDELL

DIMENSION A (KD)

LOGICAL LAST

CALL TIMEV(AA1)
NCK=0
N = ND
M = MD
KORE = KD
NPM = N + M
IF(MAX0(M, NPM, M * N) .GT. KORE) NCK = 1
MT = MM
REWIND MT
NIN = NI
REWIND NIN
NOUT = NO
REWIND NOUT
MPI = M + 1
NN = N
NEL = NPM

C = = CALCULATE THE MAXIMUM NO. OF ROWS, #K#
C 10 K = (KORE - NEL) / NEL
C
C TEST TO SEE IF THE REST OF THE MATRIX WILL FIT IN CORE
C
C LAST = K .GE. NN
IF (LAST) K = NN
C
C READ #K# ROWS OF THE AUGMENTED #A# MATRIX
C
30 NT = 0
   DO 40 IB = 1, K
      NS = NT + 1
      NT = NT + NEL
   40 READ (NIN) (A(IO), IO = NS, NT)
C
C CHECK TO SEE IF WE WERE UNLucky ENOUGH TO END UP WITH ONLY ONE ROW
C
IF (K .EQ. 1) GO TO 90
C
C #K# IS GREATER THAN #1# SO WE CAN START THE TRIANGULARIZATION
C
NELP1 = NEL + 1
NS = = NEL
NELP2 = NELP1 + 1
C
C FORM THE #TRAPZOIDAL# ARRAY (8)
C
DO 50 IB = 2, K
NP = NELP2 - IB
NS = NS + NELP1
NT = NS
   DO 50 IO = IR, K
      NT = NT + NEL
   50 MN = NT
NB = NS
A(NT) = (-A(NT)) / A(NS)
DO 50 NF = 2, NP
MN = MN + 1
NB = NB + 1
50 A(MN) = A(MN) + A(NT) * A(NB)
TF (LAST) GO TO 90
C
C --- WRITE THE TRAPEZOIDAL MATRIX ON TAPE
C
NT = 0
NP = NEL
NS = -NEL
DO 60 IO = 1, K
NS = NS + NEL + 1
NT = NT + NEL
WRITE (MT) NP, (A(IB), IB = NS, NT)
60 NP = NP + 1
NP = NP = M
NS = KORE = NEL + 1
C
C --- READ ANOTHER ROW
C
DO 80 IO = 1, NP
READ (NIN) (A(IB), IB = NS, KORE)
C
C --- MODIFY THIS ROW BY THE TRAPEZOIDAL ARRAY
C
NT = 1
MN = NS
DO 70 IB = 1, K
NB = NT
NF = MN + 1
A(MN) = (-A(MN)) / A(NT)
DO 65 NN = NF, KORE
NR = NR + 1
65 A(NN) = A(NN) + A(MN) * A(NR)
MN = NF
70 NT = NT + NELP1
C
C = = WRITE THE MODIFIED ROW ON TAPE
C
80 WRITF (NOUT) (A(NT), NT = MN, KURE)
REWIND NOUT
REWIND NIN
C
C = = SWITCH THE TAPES
C
NT = NIN
NIN = NOUT
NOUT = NT
C
C = = RECALCULATE ROW LENGTH AND LOOP BACK
C
NEL = NEL - K
NN = NEL - M
GO TO 10
C
C = = REWIND ALL TAPES
C
90 REWIND MT
REWIND NIN
REWIND NOUT
C
C = = CONDENSE THE MATRIX
C
NN = NEL
NL = NFL + 1
IF (K .EQ. 1) GO TO 105
NS = 1
NT = NFL
DO 100 IR = 2, K
NS = NS + NELP1
NT = NT + NEL
DO 100 IO = NS, NT
A(NL) = A(IO)
100 NL = NL + 1
105 N1 = KORF = K * M + 1
C
C    =    THERE, NOW WE CAN START THE BACK= SOLUTION
C    *=* NOTE..THE FIRST AVAILABLE LOCATION FOR THE SOLUTIONS IS A(N1)
C
NREM = N
NEL = NPM
LAST = K * EQ. N
NPASS = 0
C
C    =    SOLVE FOR THE ANSWERS CORRESPONDING TO #K# ROWS
C
110 KM1 = K + 1
KP1 = K + 1
NS = NL = MP1
NPASS = NPASS + 1
DO 130 MN = 1, M
NF = NS + MN
A(NF) = A(NF) / A(NS)
NT = NS
IF (KM1 *EQ. 0) GO TO 130
DO 125 IR = 1, KM1
NF = NF = IB = M
NT = NT = MP1 = IB
SUM = 0.0
NP = NF
N2 = MP1 + IR
DO 120 IO = 1, IR
SOLV 141
SOLV 142
SOLV 143
SOLV 144
SOLV 145
SOLV 146
SOLV 147
SOLV 148
SOLV 149
SOLV 150
SOLV 151
SOLV 152
SOLV 153
SOLV 154
SOLV 155
SOLV 156
SOLV 157
SOLV 158
SOLV 159
SOLV 160
SOLV 161
SOLV 162
SOLV 163
SOLV 164
SOLV 165
SOLV 166
SOLV 167
SOLV 168
SOLV 169
SOLV 170
SOLV 171
SOLV 172
SOLV 173
SOLV 174
SOLV 175
NN = NT + IO
NP = NP + N2 = IO
120 SUM = SUM + A(NN) * A(NP)
125 A(NF) = (A(NF) = SUM) / A(NT)
130 CONTINUE

C = = MOVE THE SOLUTIONS TO CONTIGUOUS LOCATIONS STARTING AT A(N1)
C
N1 = KORE + 1
DO 140 NN = 1, K
DO 135 MN = 1, M
NL = NL - 1
N1 = N1 - 1
135 A(N1) = A(NL)
140 NL = NL + NN

C = = WRITE THE SOLUTIONS ON TAPE
C
WRITE (NIN) K
NS = N1 - 1
DO 145 MN = 1, M
NT = NS + MN
145 WRITE (NIN) (A(IO), IO = NT, KORE, M)

C = = TEST IF THIS IS THE LAST PASS
C
IF (LAST) GO TO 200

C = = WE MUST NOW MODIFY THE TRIANGULAR MATRIX TO REFLECT THE EFFECT OF
C THE SOLUTIONS OBTAINED SO FAR (EQ 21)
C = = NOTE...LOCATIONS A(1) TO A(N1-1) ARE NOW FREE TO USE
C
C = = CALCULATE THE NEXT VALUES OF *NEL# AND *NREM#
C
NELOLD = NEL

SOLV
KOLD = K
NFL = NEL - K
NREM = NREM = K

C C = = NOW APPLY THE INCREDIBLE FORMULA FOR THE NEW #K#

C K = (-4 * M + 1) / 2 + IFIX(SQRT(0.25 + FLOAT((4 * M + 2) * M +
12 * (KORE - NELOLD))))
NROW = NREM = K + 1
IF (K < LT, NREM) GO TO 150
LAST = "TRUE."
NROW = 1
K = NREM

150 NS = 1
NT = NFLOLD + 1

C C = = READ IN THE ROWS TO BE MODIFIED

C DO 190 IB = 1, NREM
NT = NT + 1
IF (IB <= NROW) GO TO 160
NS = NS + NN
NT = NT + NN

160 READ (MT) NN, (A(IO), IO = NS, NT)
NP = N1 = 1
NF = NT = M = KM1
NN = NN = KOLD
DO 170 MN = 1, M
N2 = NF
NA = NP + MN
NB = NA
SUM = 0.
DO 165 IO = 1, KOLD
SUM = SUM + A(N2) * A(NA)
N2 = N2 + 1

SOLV 211
SOLV 212
SOLV 213
SOLV 214
SOLV 215
SOLV 216
SOLV 217
SOLV 218
SOLV 219
SOLV 220
SOLV 221
SOLV 222
SOLV 223
SOLV 224
SOLV 225
SOLV 226
SOLV 227
SOLV 228
SOLV 229
SOLV 230
SOLV 231
SOLV 232
SOLV 233
SOLV 234
SOLV 235
SOLV 236
SOLV 237
SOLV 238
SOLV 239
SOLV 240
SOLV 241
SOLV 242
SOLV 243
SOLV 244
SOLV 245
165 \( NA = NA + M \)
166 \( N2 = N2 + MN - 1 \)
170 \( A(N2) = A(N2) \rightarrow SUM \)

C
C = - WRITTF THE MODIFIED ROW ON TAPE OR CUMPENSE THE ROW
C

NL = NT = M + 1
IF (IB \geq NROW) GO TO 175
NF = NL = KPI
WRITTF (NOUT) NN, (A(IO), IO = NS, NF), (A(IO), IO = NL, NT)
GO TO 190

175 NF = NL = KOLD
DO 180 MN = NL, NT
A(NF) = A(MN)

180 NF = NF + 1
190 CONTINUE
REWIND MT
REWIND NOUT

C
C = - SWITCH THE TAPES
C

NT = MT
MT = NOUT
NOUT = NT

C
C = - LOOP BACK THRU THE SOLUTION
C

NL = NF
GO TO 110

C
C = - START TO WRAP IT UP
C

200 REWIND NN
N2 = N
**NOTE:** At this point all locations A(1) thru A(KORE) are free

```fortran
C DO 220 IH = 1, NPASS
READ (NIN) K
N1 = NP = K + 1
NS = N1
NT = N2
C
C = = READ IN THE SOLUTIONS
C
DO 210 IO = 1, M
READ (NIN) (A(NN), NN = NS, NT)
NT = NT + N
210 NS = NS + N
220 N2 = N1 = 1
C
C = = REWIND ALL INPUT TAPES
C
REWIND NIN
REWIND MT
REWIND NOUT
C
C = = WRITE THE SOLUTIONS ON TAPE
C
NT = 0
DO 230 IO = 1, M
NS = NT + 1
NT = NT + N
230 WRITE (NW) (A(NN), NN = NS, NT)
C
CALL //MEV(AA2)
C
BB = (AA2 - AA1) / 60,
C
WRITE (6, 300) N, N, M, BB
C
300 FORMAT (AHTHE I5, 2H X I5, 12H MATKIX WITH I4, 35H RIGHT SIDES WA
C 1S SOLVDF DIRECTLY IN F8.3, 9H MINUTES.)
RETURN
END
```
OVERLAY(AXY,4,0)
PROGRAM PART4
SUBROUTINE PART4

* COMPUTE VELOCITY COMPONENTS AND PRINT

COMMON /IPSF/ PSF
COMMON MHDR(10), CASE, NNU
1 ,FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
2 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
3 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
4 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
5 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27
COMMON NT, NO(11), MN, NUNA(5), TYPEA(5)
1 NER1, NER2, NMA, NSIGA, NSIGC
2 NUNC(5), TYPEC(5), NLF(11), IFC, NSIGEC
3 TYPFEC(5), NUNEC(5)
COMMON /COMB1N/CAY(2)
INTEGER FLG03 ,FLG04 ,FLG05 ,FLG06 ,FLG07
1 ,FLG08 ,FLG09 ,FLG10 ,FLG11 ,FLG12
2 ,FLG13 ,FLG14 ,FLG15 ,FLG16 ,FLG17
3 ,FLG18 ,FLG19 ,FLG20 ,FLG21 ,FLG22
4 ,FLG23 ,FLG24 ,FLG25 ,FLG26 ,FLG27
REAL MN

COMMON /C4/ X1(100), Y1(100), X2(100), Y2(100), DELS(100)
1 SINA(100), COSA(100), XP(100), YP(100)
COMMON /TC/ Z(100), PHI(100,5), K(100,5), T(100,5),
1 SUMV, SUMM(5)
2

* START

DO 20 J = 1,10
DO 20 I = 1, 500
20 RB(I,J) = 0.0
REWRITE 3
* READ OFF-BODY XP,YP
NP = NO(NR+1)
READ (12) (XP(I), I = 1, NP), (YP(I), I = 1, NP)
* READ XI, X1, Y2, DELS WITH MATCH NO. ADJUSTMENT IF ANY
30 NI = NT + NB
READ (12) (X1(I), I = 1, NI), (Y1(I), I = 1, NT), (X2(I), I = 1, NT)
100 SUMV = SUMV + B(I)*DFLS(I)*Y2(I)*#2
   SUMV = SUMV + 141593
   IF (FLG03, LE, 0) GO TO 1000
50 L = 1
55 LS = 0
   IF (FLG16, NE, 0) GO TO 200
   DO 150 I = 1, NT
   RB(I,L) = A(I)
150 RB(I,L+1) = B(I)
200 IF (NNU) 600, 600, 300
300 DO 500 J = 1, NNU
   READ (4) MS, (A(I), I = 1, NT), (B(I), I = 1, NT)
   TF (MS, EQ, 1, OR, MS, EQ, 2, OR, MS, EQ, 5) GO TO 500
   L = L + 2
500 LS = LS + 1
   IF (LS, EQ, 1, AND, FLG16, GT, 0) L = L + 2
DO 400 I = 1, NT
RB(I,L) = A(I)
400 RB(I,L+1) = RB(I,L+1)
500 CONTINUE
IF (FLG23 .LE. 0) GO TO 600
IPV = NB = FLG14 + 1
NN = NMA
DO 550 KCNT = IPV, NB
L = L + 2
NN = NN + ND(KCNT) = 1
C *** READ NOTS COLUMNS OF RMS
READ(4) (RB(I,L), I=1,NT), (RB(I,L+1), I=1,NT)
C *** MULITPLY NOTS COLUMN BY LAST INPUT PV ON THAT BODY
DO 550 I=1,NT
RB(I,L) = RB(I,L) * Z(NN)
550 RB(I,L+1) = RB(I,L+1) * Z(NN)
600 REWIND 4
NSTG = NSIGA
IF (FLG23 .LE. 0) NSIG = 2.0 * NSIG = 1
C CALL AXIS
CALL OVERLAY (4MAXXY, 4, 1, 6HRECALL)
1000 IF (FLG04 .LE. 0) GO TO 2000
IF (FLG03 .LE. 0) GO TO 1050
READ (4) (A(I), I=1,NT), (B(I), I=1,NT)
1050 L = 1
LS=0
IF (FLG17 .LE. 0) GO TO 1200
DO 1100 I = 1, NT
RB(I,L) = A(I)
1100 RB(I,L+1) = B(I)
1200 IF (NNU) 1600, 1600, 1300
1300 DO 1500 J = 1, NNU
READ (4) MS, (A(I), I=1,NT), (B(I), I=1,NT)
IF (MS.EQ.0 .OR. MS.EQ.2 .OR. MB.EQ.4) GO TO 1500
L = L + 2
C

PAR4  106
IF (LS.EQ.1.AND.,FLG17,GT,0) L=L+2
PAR4  107
DO 1400 I = 1, NT
PAR4  108
RB(I,L) = A(I)
PAR4  109
1400 RB(I,L+1) = R(I)
PAR4  110
CONTINUE
PAR4  111
1600 REWIND 4
PAR4  112
NSTG = NSIGC
PAR4  113
CALL CROSS
PAR4  114
CALL OVERLAY (4HAXSY,4,2,6HRECALL)
PAR4  115
C2000 IF (FLG21,LE,0) RETURN
PAR4  116
2000 IF (FLG21,LE,0) GO TO 2500
PAR4  117
2050 REWIND 4
PAR4  118
IF(FLG22,GT,0) GO TO 2400
PAR4  119
L = 0
PAR4  120
C*** ***IF CONTROL REACHES THIS POINT, THERE IS AT LEAST 1 NNU
PAR4  121
C*** ***SKIP RECORD WITH SIN AND COS
PAR4  122
READ (4)
PAR4  123
DO 2200 J = 1, NNU
PAR4  124
READ(4) MS, ( A(I),I=1,NT ), ( B(I),I=1,NT )
PAR4  125
L = L + 1
PAR4  126
DO 2200 I = 1, NT
PAR4  127
RB(I,L) = A(I)
PAR4  128
2200 RB(I,L+1) = R(I)
PAR4  129
2400 REWIND 4
PAR4  130
NSIG = NSIGC
PAR4  131
C*** ***CALL TO EXCROS FOR GENERATFDU (RESEP) BOUNDARY CONDITIONS
PAR4  132
CALL EXCROS
PAR4  133
CALL OVERLAY (4HAXSY,4,3,6HRECALL)
PAR4  134
RETURN
PAR4  135
C
PAR4  136
2500 CONTINUE
PAR4  137
END
OVERLAY(AXSY,4,1)
PROGRAM AXIS
SUBROUTINE AXIS

* COMPUTE AXISYMMETRIC VELOCITY COMPONENTS AND PRINT

COMMON /COMB1N/CHAY(2)
COMMON /IPSF/ PSF
COMMON /MEDR(10), CASEF, N8, NNU
1 FLG03, FLG04, FLG05, FLG06, FLG07
2 FLG08, FLG09, FLG10, FLG11, FLG12
3 FLG13, FLG14, FLG15, FLG16, FLG17
4 FLG18, FLG19, FLG20, FLG21, FLG22
5 FLG23, FLG24, FLG25, FLG26, FLG27
COMMON /MT(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNC(5), TYPEC(5), NL(11), IFC, NSIGEC,
3 TYPSEC(5), NSEI(5)

DOUBLE PRECISION MDR, CASE
INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1 FLG08, FLG09, FLG10, FLG11, FLG12
2 FLG13, FLG14, FLG15, FLG16, FLG17
3 FLG18, FLG19, FLG20, FLG21, FLG22
4 FLG23, FLG24, FLG25, FLG26, FLG27
REAL MN

COMMON /C4/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1 SNA(100), COSA(100), X(100), YP(100)
COMMON /TC/ RB(100, 10), SIG(100, 5), A(100), B(100),
1 Z(100), PHI(100, 5), XN(100, 5), T(100, 5),
2 NSIG, NP, NI,
3 SUMV, SUMM(5)
COMMON /ITERF/ IER

DIMENSION UB(100), YB(100), XB(100)
DIMENSION VX(100,5), VY(100,5), V(100,5), TH(100,5), CP(100,5), SUMTDS(5)

DATA FOURPI /12,5683706/
EQUIVALENCE ( VX(1,1) , XN(1,1) ) , ( VY(1,1), T(1,1) ) , ( V(1,1), T3(1,1) ) , ( TH(1,1), SIG(1,1) ), ( CP(1,1), T3(1,1) )

C
* START
NC=NT
IF (FLG19,GT,0) NC=NMA
IF (FLG08,EQ,0) GO TO 10
C
* TITLE FOR MATRIX PRINT
WRITE(6,150)HDR,CASE,PSF
WRITE(6,8)
A FORMAT (1H43H MATRICES A,R,Z BY ROWS * AXISYMMETRIC FLOW //)
C
* READ AXIS SIGMA
10 DO 20 N=1,NSTG
SUMM(N)=0.0
SUMTDS(N)=0.0
20 READ (3) (SIG(I,N),I=1,NC)
IF (FLG19,LE,0) GO TO 25
READ (4)
NR = NMA+1
IF (FLG23,GT,0) GO TO 21
READ (4) (SIG(I,1),I=NR,NT)
REWIND 4
GO TO 25
C
C*** * RING WING
C
21 LIFBOD = 0
DO 22 K = 1,NB
IF( NLK(K),GT,0) GO TO 22
LIFBOD = LIFBOD + 1
22 CONTINUE
   LBP1 = LFBOD + 1
   DO 23 N=1,LBP1
   DO 23 I=NR,NT
23  SIG(I,N) = 0.0
   LBP2 = LBP1 + 1
   DO 24 N = LBP2,NSIG
      DO 24 I=NR,NT
24  SIG(I,N) = SIG(I,N) / (= FOURPI)
   REWIND 4
C *** SIGMAS HERE HAVE BECOME THE INPUT PV
   DO 26 N = LBP2,NSIG
   DO 26 I=NR,NT
26  SIG(I,N) = SIG(I,N) */ (=FOURPI)
C * NO. OF MIDPOINTS LOOP
   25 DO 100 I=1,NT
      * READ MATRICES A, B, Z
      READ(9) (A(J),J=1,NT),(B(J),J=1,NT),(Z(J),J=1,NT)
C * NO. OF FLOWS LOOP
      N1=0
      DO 70 N=1,NSIG
      N1=N1+2
      SN=0.0
      ST=0.0
      SP=0.0
      DO 30 J=1,NT
      SN=SN+A(J)*SIG(J,N)
      ST=ST+B(J)*SIG(J,N)
      IF(FLG23 .GT. 0) Z(J) = 0.0
30   SP=SP+Z(J)*SIG(J,N)
      IF(FLG22 .GT. 0) GO TO 68
      IF(FLG12 .EQ. 0) GO TO 40
      XN(I,N) = SN
      PHT(I,N) = SP - RB(I,N1-1)
      GO TO 50
40 XN(I,N)=SN=RB(I,N1=1)
PHI(I,N)=SP
50 IF (FLG1,=EQ,0) GO TO 60
   T(I,N)=ST
   GO TO 65
60 T(I,N)=ST+RB(I,N1)
65 SUMM(N)=SUMM(N)+PHI(I,N)  * Y(P(I}*RB(I,N1=1)*DELS(I)
   CP(I,N)=1.0-T(I,N)**2
   GO TO 70
68 XN(I,N) = SN
   PHI(I,N) = SP.
   T(I,N) = ST
   CP(I,N) = 1.0-T(I,N)**2
70 CONTINUE
   IF (FLG0B,=EQ,0) GO TO 100
   WRITE (6,80) I,(A(J),J=1,NT)
80 FORMAT (1HO 13H MATRIX A ROW I6/ (1H 10F10.5))
   WRITE (6,85) I,(B(J),J=1,NT)
85 FORMAT (1HO 13H MATRIX B ROW I6/ (1H 10F10.5))
   WRITE (6,90) I,(Z(J),J=1,NT)
90 FORMAT (1HO 13H MATRIX Z ROW I6/ (1H 10F10.5))
100 CONTINUE
   IF (MN,=EQ,0,0) GO TO 130
   C
   * MACH NO. ADJUSTMENT
   D1=MN*MN
   D2=1.0-D1
   D3=SQR(D2)
   D4=.7*D1
   D5=.2*D1
   DO 120 N=1,NSIG
   DU 120 I=1,NT
   TX=(T(I,N)*COSA(I)=1.) / D2+1.0
   TY=(T(I,N) * SINA(I)) / D3
   T(I,N)=SQR(TX*T+TY*TY)
120 CP(I,N)=(1.0+D5*(1.0-T(I,N)**2)) * 3.51*10**2)
   D4
C * ELIMINATE MACH NO EFFECT FOR PRINTOUT

DO 122 I=1,N1

122 X1(I)=X1(I)*M3

N=0
J1=0
DO 126 K=1,NR
M=N+1
N=N+ND(K)=1
DO 124 J=M,N
J1=J1+1
T1=X1(J1+1)-X1(J1)
T2=Y1(J1+1)-Y1(J1)
X2(J)=(X1(J1+1)+X1(J1))/2.
DELs(J)=SQR((T1+T1+T2+T2))
COSA(J)=T1/DELS(J)
124 SINA(J)=T2/DELS(J)
126 J1=J1+1

C * PRINT AXIS FLOW (ON-BODY) OUTPUT

130 DO 250 L=1,NSIG
KA=L
IF (FLG16,LE.0) KA=L-1
IF (FLG22,GT.0,UR,FLG23,GT.0)KA=L
IF (FLG22,GT.0)GO TO 136
SUMM(L)=6.2831853*SUMM(L)
DO 135 J=1,N1
135 SUMTDS(L)=SUMTDS(L)+T(J,L)*DELS(J)
136 I=1
J=1
REWIND 1
REWIND 3
M=1
N=ND(M)
NSM=N+1
X8(1)=X1(1)
UB(1)=.9999999
YB(1) = Y1(1)
DN 138 KK = 1, NSM
XB(KK+1) = X2(KK)
YB(KK+1) = Y2(KK)
138 1!B(KK+1) = CP(KK, L)
WRITE(3) N
WRITE(3) (XB(K), K = 1, N)
WRITE(3) (YB(K), K = 1, N)
WRITE(3) (UB(K), K = 1, N)
IF (ITER, GE, 1) GO TO 139
WRITE(15) N
WRITE(15) (X1(K), K = 1, N)
WRITE(15) (Y1(K), K = 1, N)
139 CONTINUE
LCTR = 22
140 WRITE(6, 150) HEDR, CASE, PSF
150 FORMAT (1H1 25X, 26HDougLas AIRCRAFT COMPANY / 
1 28X, 21HlNостью EUROPE DIVISION /// 
2 6X, 10A6, 4X, 10HCASE NO. A6, 10H PSF = .A4 ///)
IF (FLG22, GT, 0) GO TO 178
IF (L, GT, 1, OR, FLG16, NE, 0) GO TO 170
WRITE (6, 160)
160 FORMAT (1H34H ON-BODY UNIFORM AXISYMMETRIC FLOW )
GO TO 190
170 IF (TYPEA(KA), GE, 0, 0) GO TO 175
WRITE (6, 172)
172 FORMAT (1H44H FLOW GENERATOR * ROTATING BODY * TYPE ERROR )
175 IF (NUNA(KA), EQ, 123456) WRITE (6, 177)
177 FORMAT (27H ON-BODY STRIP VORTEX FLOW )
178 IF (NUNA(KA), NE, 123456) WRITE (6, 180) NUNA(KA)
180 FORMAT (1H42H ON-BODY NON-UNIFORM AXISYMMETRIC FLOW NO., IB )
190 WRITE (6, 200)
200 FORMAT (1H5X 24H TRANSFORMED COORDINATES // 
1 12X 1HX 13X 1HY 13X 2HT1 12X 2HCP 9X 5HSlN A 
2 6X 5HCOS A 7X 5HSlGMA 11X 1HN 13X 3HPHI // )
210 WRITE (6,220) X1(I),Y1(I),X2(J),Y2(J), 
1 T(J,L),CP(J,L),STNA(J),COSA(J),SIG(J,L),XN(J,L),PHI(J,L)
220 FORMAT (1H I3,2F14.7/ 4X 4F14.7,2F11.5,3F14.7)
220 I=I+1
220 J=J+1
220 IF (I.EQ.N) GO TO 230
220 IF (I.LT.N) GO TO 210
220 LCTR=LCTR+22
220 GO TO 140
230 M=M+1
230 N=N+1(M)
230 WRITE (6,240) X1(I),Y1(I)
240 FORMAT (1H I3, 2F14.7//)
240 I=I+1
250 IF (J = NT ) 210, 242, 242
242 IF (FLG2.2,GT,0) GO TO 250
242 WRITE (6,244) SUMM(L), SUMV, SUMDS(L)
244 FORMAT (1HO 10X 13H ADDED MASS = F12.7, 4X 9H VOLUME = F12.7, 5X 18H SUM (T)(DELTA S) = F12.7)
250 CONTINUE
252 LL = .1
252 IF (FLG23 .GT, 0) CALL COMBU(LL)
252 IF (FLG05 .EQ, 0) RETURN
252 IF (FLG05 .EQ, 0) GO TO 700
252 * OFF=BODY POINT
253 IF (FLG15 .LE, 0) GO TO 258
253 M = 0
254 DO 254 I = 1, N8
254 IF (NLF(I) .LE, 0) M = M + 1
254 IF (M .EQ, 0) GO TO 258
255 READ (4)
255 IF (FLG22.GT,0) READ(4)
DO 256 J = 1, M
256 READ(4) (RB(I,J), I = 1, N), (T3(I,J), I = 1, N)
REWIND 4.
258 DO 300 I = 1, N
      L = 0
      * READ MATRICES X, Y, Z
      READ (9) (A(J), J = 1, NT), (B(J), J = 1, NT), (Y(J), J = 1, NT)
      * NO. OF FLOW
      DO 300 N = 1, NSIG
          KA = N
          IF (FLG16 .LE. 0) KA = N + 1
          SX = 0.0
          SY = 0.0
          SP = 0.0
          * NO. OF ELEMENTS LOOP
      DO 260 J = 1, NT
          SX = SX + A(J) * SIG(J, N)
          SY = SY + B(J) * SIG(J, N)
          IF (FLG23 .GT. 0) Z(J) = 0.0
          260 SP = SP + Z(J) * SIG(J, N)
          PHI(I, N) = SP
          IF (FLG22 .GT. 0) GO TO 270
          IF (FLG11 .GT. 0) GO TO 270
          IF (N, NE, 1, OR, FLG16 .GT. 0) GO TO 262
          VX(I, N) = SX + 1.
          GO TO 280
      262 IF (NUNA(KA), NE, 123456) GO TO 270
      L = L + 1
      VX(I, N) = SX + RR(I, L)
      VY(I, N) = SY + T3(I, L)
      GO TO 300
      270 VX(I, N) = SX
      280 VY(I, N) = SY
      300 CONTINUE
      IF (MN, EQ, 0, 0) GO TO 330
C

* MACH NO. ADJUSTMENT
DO 320 N=1,NSIG
DO 320 I=1,NP
VY(I,N)=VY(I,N)/D3
320 VX(I,N)=(VX(I,N)-1.)/D2+1.
DO 322 I=1,NP
322 XP(I)=XP(I)*D3

C

* COMPUTE VT AND THETA
330 DO 335 I=1,NP
VERT(I,N)=SQR2*(VX(I,N)**2+VY(I,N)**2)
335 TH(I,N)=ATAN2(VY(I,N),VX(I,N)) * 57.29578

C

* PRINT AXIS FLOW (OFF-BODY) OUTPUT
DO 450 L=1,NSIG
KA = L
IF(FLG16 .LE. 0)KA = L - 1
IF(FLG22 .GT. 0 .OR. FLG23 .GT. 0) KA = L
I=1
LCTR=45
340 WRITE(6,150)HEDR,CASE,PSF
IF (L .GT. 1 .OR. FLG16 .NE. 0) GO TO 370
IF(FLG22 .GT. 0) GO TO 378
WRITE (6,360)
360 FORMAT (1H 35H OFF-BODY UNIFORM AXISYMMETRIC FLOW )
GO TO 390
370 IF (TYPEA(KA),GE.,0.) GO TO 375
WRITE (6,172)
375 IF (NUNA(KA),EQ.,123456) WRITE (6,377)
377 FORMAT (28H OFF-BODY STRIP VORTEX FLOW)
378 IF (NUNA(KA),NE.,123456) WRITE (6,380) NUNA(KA)
380 FORMAT (1H 43H OFF-BODY NON-UNIFORM AXISYMMETRIC FLOW NO. I8)
390 WRITE (6,400)
400 FORMAT (1H 5X, 24H TRANSFORMED COORDINATES //
1 12X 1HX 13X 1HY 13X 2HVX 12X 2HVV 12X 2HVT 10X
2 5THFTA 11X 3HPHT //)

410 WRITE (6,420) I,XP(I),YP(I),
     1 TH(I,L), PHI(I,L)
420 FORMAT (1I3, 1F14.7)
     I=I+1
     IF (I.GT.NP) GO TO 450
     IF (I.LE.LCTR) GO TO 410
     LCTR=LCTR+45
     GO TO 340
450 CONTINUE
500 LL = 0
     IF(FLG23 .GT. 0)CALL COMMU(LL)
     RETURN
700 CONTINUE
END
SURROUTINE COMBO(LL)

COMMON /IPSF/ PSF
COMMON /COMBIN/CHAY(2)

COMMON MDR(10), CASE, WB, NNU
1 1, FLG03, FLG04, FLG05, FLG06, FLG07
2 2, FLG08, FLG09, FLG10, FLG11, FLG12
3 3, FLG13, FLG14, FLG15, FLG16, FLG17
4 4, FLG18, FLG19, FLG20, FLG21, FLG22
5 5, FLG23, FLG24, FLG25, FLG26, FLG27

COMMON NT, ND(11), MN, NUNA(5), TYPEA(5),
1 NER1, NER2, NMA, NSIGA, NSIGC,
2 NUNEC(5), TYPEC(5), NLF(11), IEC, NSIGEC,
3 DOUBLE PRECISION MDR, CASE

INTEGER FLG03, FLG04, FLG05, FLG06, FLG07
1 1, FLG08, FLG09, FLG10, FLG11, FLG12
2 2, FLG13, FLG14, FLG15, FLG16, FLG17
3 3, FLG18, FLG19, FLG20, FLG21, FLG22
4 4, FLG23, FLG24, FLG25, FLG26, FLG27

REAL MN

COMMON /C4/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),
1 SINA(100), COSA(100), XP(100), YP(100)

COMMON /TC/ RB(100,10), SIG(100,5), A(100), B(100),
1 Z(100), PHI(100,5), XN(100,5), T(100,5),
2 T3(100,5), NSIG, NP, NI,
3 SUMV, SUMM(5)

DIMENSION C(2,2), DV(2), TFIRST(2,5), TLAST(2,5), TSUM(2,5)
DIMENSION CP(100)
EQUIVALENC (CP(1), A(1))
EQUIVALENC (DV(1), CHAY(1))
DIMENSION VX(100,5), VX(100,5), VT(100,5)
EQUIVALENC (VX(1,1), VX(1,1), VX(1,1), TT(1,1))
1 (VT(1,1), T3(1,1))
C ** ICNT WILL BE THE NUMBER OF LIFTING BODIES
C * NFLOW WILL BE THE NUMBER OF FLOWS
C
ICNT = 0
DO 10 K=1,NB
10 IF( NLF(K) .LE. 0) ICNT = ICNT + 1
NFLOW = 1 + 2*ICNT
IF(LI .EQ. 0) GO TO 1000
READ(5,4)(DV(I),I=1,ICNT)
4 FORMAT (6F10.0)
WRITE(6,6) (DV(I),I=1,ICNT)
6 FORMAT (1H1,42H THE INPUT DV FOR COMBINATION SOLUTION ARE / 
   1 (2X,6F10.4) )
C
C *** IPTCNT WILL BE THE LAST MIDPOINT ON BODY K
C
IPTCNT = 0
C
ILIFT = 0
DO 100 K=1,NB
IPTCNT = IPTCNT + ND(K) = 1
IFIRST = IPTCNT + ND(K) + 2
IF ( NLF(K) .GT. 0 ) GO TO 100
ILIFT = ILIFT + 1
C
DO 50 J=1,NFLOW
TFIRST(ILIFT,J) = T(IFIRST,J)
TLAST(ILIFT,J) = T(IPTCNT,J)
50 TSUM(ILIFT,J) = TSUM(ILIFT,J) + TLAST(ILIFT,J)
100 CONTINUE
C
C ** IPVBOD WILL BE 1ST PRESCRIBED VORTEX FLOW
C * LASTSV WILL BE LAST STRIP VORTEX FLOW
IPVBOD = 2*ICNT
LASTSV = IPVBOD - 1
DO 300 I=1,ICNT
   JCNT = 0
   DV(I) = DV(I) - TSUM(I,1)
C
   DO 200 J=IPVBOD,NFLOW
      DV(I) = DV(I) - TSUM(I,J)
C
   DO 250 J=2,LASTSV
      JCNT = JCNT + 1
      250 C(I,JCNT) = TSUM(I,J)
   300 CONTINUE
C
   CALL SOLCOM( DV, C, ICNT )
   DO 500 I = 1,NT
   C
     *** ADD PV FLOWS TO AXIS FLOW FOR COMBINATION SOLUTION ON BODY
     JCNT = 0
     DO 350 J=IPVBOD,NFLOW
        XN(I,1) = XN(I,1) + XN(I,J)
        350 T(I,1) = T(I,1) + T(I,J)
     C
     *** ADD K * STRIP VORTEX BODY VELOCITY FOR COMBINATION SOLUTION
     DO 400 J = 2,LASTSV
        JCNT = JCNT + 1
        XN(I,1) = XN(I,1) + CHAY(JCNT)*XN(I,J)
        400 T(I,1) = T(I,1) + CHAY(JCNT)*T(I,J)
        CP(I) = 1.0 - T(I,1)**2
   500 CONTINUE
I=1
J=1
M=1
N=ND(M)
LCTR = 22
540 WRITF(6,550)HEDR,CASF,PSF
550 FORMAT(1H1, 25X, 26H DOUGLAS AIRCRAFT COMPANY /)
   1  28X, 21HLONG BEACH DIVISION //
   2  6X, 10AB, 4X, 10HCASE NO. A6, 9H PSF = A4 //
   WRITE(6, 560)
560 FORMAT(1H, 21H COMBINATION SOLUTION )
   WRITE(6, 700)
700 FORMAT(1H 5X 20H TRANSFORMED COORDINATES //
   1 12X, 1XH 13X 1HY 13X 2HT1 12X 2HCP 9X 5HSIN A 6X 5HCOS A 11X 1HN
   2 //)
710 WRITE(6, 720) I, X(I), Y(I), X2(J), Y2(J), T(J, 1), CP(J), SINA(J), COSA(J)
   2 , XN(J, 1)
720 FORMAT(1H I3, 2F14, 7 / 4X 4F14, 7, 2F11, 5,F14, 7 )
   I=I+1
   J=J+1
   IF( I .EQ. N ) GO TO 730
   IF( I .LE. LCTR ) GO TO 710
   LCTR = LCTR + 22
   GO TO 540
730 M=M+1
   N=N+ND(M)
   WRITE(6, 740) I, X(I), Y(I)
740 FORMAT(1H I3, 2F14, 7 // )
   I=I+1
   IF( J .LT. NT ) GO TO 710
   M1 = 0
   N1 = 0
   DO 800 K = 1, NB
   M1 = M1 + 1
   N1 = N1 + ND(K) - 1
   CIRC = 0.0
   THRUST = 0.0
   DO 790 I = M1, N1
   CIRC = CIRC + ( T(I, 1) * DELS(I) )
   790 THRUST = THRUST + ( Y2(I) * CP(I) * SINA(I) * DELS(I) )
   THRUST = -6.283186 * THRUST
WRITF(6,795)K,CIRC,THRUST
795 FORMAT(///13H BODY NO. , I4,5X,14H CIRCULATION = ,F14.7,5X,
1 9H THRUST = ,F14.7)
800 CONTINUE
RETURN
C *** OFF BODY COMBINATION SOLUTION
1000 IPVANO = 2 + ICNT
     LASTSV = IPVANO - 1
     DO 1500 I=1,NP
         JCNT = 0
     C *** ADD PV FLOWS TO AXES FLOW FOR COMBINATION SOLUTION OFF BODY
         DO 1350 J = IPVANO,NFLOW
         VX(I,1) = VX(I,1) + VX(I,J)
         1350 VX(I,1) = VX(I,1) + VX(I,J)
     C *** ADD K * STRIP VORTEX OFF BODY VELOCITY
         DO 1400 J=2,LASTSV
             JCNT = JCNT + 1
             VX(I,1) = VX(I,1) + CHAY(JCNT) * VX(I,J)
         1400 VX(I,1) = VX(I,1) + CHAY(JCNT) * VX(I,J)
         VY(I,1) = SQRT( VX(I,1)**2 + VY(I,1)**2 )
     1500 CONTINUE
     I = 1
     LCTR = 45
     1540 WRITE(6,550)HEDR,CASE,PSF
     WRITF(6,1560)
     1560 FORMAT(1H ,31H COMBINATION SOLUTION OFF BODY )
     WRITF(6,1700)
     1700 FORMAT(1H ,5X,24H TRANSFORMED COORDINATES //
         1 12X,1HX,13X,1HY,13X,2HVX,12X,2HYY,12X,2HVT //)
     1710 WRITF(6,1720)I,XP(I),YP(I),VX(T=1),VY(T=1),VT(T,1)
     1720 FORMAT(1H ,I3,5F14.7)
         I = I + 1
         IF(I ,GT. NP)GO TO 1750
         IF(I ,LE. LCTR )GO TO 1710
         LCTR = LCTR + 45
SUBROUTINE SOLCOM( DV, A , ICNT)

C ** NOTE THAT THIS SUBROUTINE SOLVES ONLY A 1X1 OR 2X2 MATRIX
C ** IT IS SEPARATED SO THAT IF PROGRAM IS EVER INLARGED, THIS
C ** IS WHERE THE MATRIX SOLUTION FOR THE COMBINATION PART OF
C ** THE PROGRAM WILL GO, THE MATRICES HAVE BEEN FORMED GENERALLY
C ** IN SUBROUTINE COMBO.
C
D     DIMENSION DV(2), A(2,2)
IF(ICNT .EQ. 2) GO TO 20
DV(1) = DV(1) / A(1,1)
RETURN
20  DV(1) = (DV(1) = (A(1,2)/A(2,2) ) * DV(2) ) / 
     1     ( A(1,1) = A(1,2)*A(2,1) / A(2,2) )
DV(2) = ( DV(2) = A(2,1)*DV(1) ) / A(2,2)
RETURN
END
1 (VZ(1,1), T3(1,1)), (T2(1,1), T(1,1))

C  * START
C
IF (FLGO0, EQ, 0) GO TO 10
C
* TITLE FOR MATRIX PRINT
WRITE(6,150) HEDR, CASE, PSF
WRITE (6, 8)
8 FORMAT (' 1H 36H MATRICES A, B, Z BY ROWS A CROSS FLOW //')
C
* READ CROSS SIGMAS
10 DO 20 I=1, NSTG
   SUMM(I)=0.0
20 READ (5) (SIG(I,N), I=1, NT)
   N D0 100 I=1,NT
   * NO. OF MIDPOINTS LOOP
   DO 100 I=1, NT
   * READ MATRICES A, B, Z
   READ (10) (A(J), J=1, NT), (B(J), J=1, NT), (Z(J), J=1, NT)
   * NO. OF FLOWS LOOP
   M=N
   DO 70 N=1, NSIG
      M=M+2
      SA=SA+A(J)*SIG(J,N)
      SB=SB+B(J)*SIG(J,N)
      S2=S2+Z(J)*SIG(J,N)
   70 CONTINUE
   * NO. OF ELEMENTS LOOP
   DO 30 J=1, NT
      SA=SA+A(J)*SIG(J,N)
      SB=SB+B(J)*SIG(J,N)
      S2=S2+Z(J)*SIG(J,N)
   30 CONTINUE
C
* INITIALIZE UNIFORM OR NON-UNIFORM PARAMETERS
IF (FLGZ1, GT, 0) GO TO 38
IF (N, EQ, 1, AND, FLG17, LF, 0) GO TO 35
C1=RR(I, M)
C2=-RR(I, M-1)
C3=0.0
GO TO 40
K = L
I F (FLG17 .LE. 0) K = 1
I F (FLG21 .GE. 0) G O T O 138
S U M M (L) = 3.141593 * S U M M (L)
138  I = 1
J = 1
M = 1
N = N D ( M )
L C T R = 22
140  W R I T E (6,150) H E D R , C A S E , P S F
150  F O R M A T (1H1 25X, 26H DOUGLAS A I R C R F T C O M P A N Y /
  1  28X, 21H L O N G B E A C H D I V I S I O N //
  2  6X,10A6,4X,10HC A S E N O . A6,10H PSF = ,A4 //)
T F (FLG22 .GT. 0) G O T O 175
I F (L . LT . 1 . O R . FLG17 . LT . 0) G O T O 170
W R I T E (6,160)
160  F O R M A T (1H 27H O N - B O D Y U N I F O R M C R O S S F L O W )
G O T O 190
170  I F (T Y P E C (K C ) . GE . 0 ) G O T O 175
W R I T E (6,172)
172  F O R M A T (1H 31H F L O W G E N E R A T O R * R O T A T I N G B O D Y )
175  W R I T E (6,180) N U N C (K C )
180  F O R M A T (1H 35H O N - B O D Y N O N - U N I F O R M C R O S S F L O W N O . I 8 )
190  W R I T E (6,200)
  1  12X 1 H X 13X 1 H Y 13X 2 H T 12X 2 H T 3 9 X 5 H 3 I N A
  2  6X 5 H C O S A 7 X 5 H S I G M A 1 X 1 H N 13X 3 H P H I //)
210  W R I T E (6,220) I , X 1 ( I ) , Y 1 ( I ) , X 2 ( J ) , Y 2 ( J ) ,
T P ( J , L ) , T ( J , L ) ,
  1  S I N A ( J ) , C O S A ( J ) , S I G ( J , L ) , S I G ( J , L ) , X I N ( J , L ) , P H I ( J , L )
220  F O R M A T (1H 13,2F14.7/ 4X 4F14.7,2F11.5,3F14.7)
  I = I + 1
  J = J + 1
I F ( I , . LE . N ) G O T O 230
I F ( I , . LE . L C T R ) G O T O 210
L C T R = L C T R + 22
GO TO 140
230 M=M+1
3 = N+ND(M)
WRITE (6,240) I,X1(I),Y1(I)
240 FORMAT (1H13, 2F14.7 //)
I=I+1
IF(J,GT,NT)GO TO 242
GO TO 210
242 IF(FLG22,GT,0)GO TO 250
WRITE(6,244) SUMM(L), SUMV
244 FORMAT (1HO 10X,14H ADDED MASS = F12.7; 4X,10H VOLUME = F12.7)
250 CONTINUE
252 IF (FLG05,EQ,0) RETURN
C  * OFF=BODY POINT
   DO 300 I=1,NP
C  * READ MATRICES X,Y,Z
   READ (10) (A(J),J=1,NT), (B(J),J=1,NT), (Z(J),J=1,NT)
C  * NO. OF FLOW
   DO 300 N=1,NSIG
      SX=0.0
      SY=0.0
      SP=0.0
   C  * NO. OF ELEMENTS LOOP
      DO 260 J=1,NT
         SX=SX+A(J)*SIG(J,N)
         SY=SY+B(J)*SIG(J,N)
      260 SP=SP+Z(J)*SIG(J,N)
      VX(I,N)=SX
      PHY(I,N)=VP(I)*SP
   IF (FLG22,GT,0) GO TO 270
   IF (FLG11,GT,0) OR, N, NE, 1, OR, FLG17, GT, 0) GO TO 270
   VY(I,N)=SY+1.
   VZ(I,N)=SP+1.
   GO TO 300
C  * PERTURBATION OR NON=UNIFORM VX,VZ
270  VY(I,N)=SY
    VZ(I,N)=SP
300  CONTINUE

C * PRINT CROSS FLOW (OFF-BODY) OUTPUT

330  DO 450  L=1,NSIG
    KC = L
    IF (FLG17.LE.0) KC=L-1
    I=1
    LCTR=45
340  WRITE(6,150) NEDR,CASF,PSF
    IF (FLG22.GT.0) GO TO 375
    IF (I.GT.1.OR.FLG17.NE.0) GO TO 370
    WRITE (6,360)
360  FORMAT (1H 2AH OFF-BODY UNIFORM CROSS FLOW )
    GO TO 390
370  IF (TYPEC(KC).GE.0.) GO TO 375
    WRITE (6,172)
375  WRITE (6,380) NUNC(KC)
380  FORMAT (1H 36H OFF-BODY NON-UNIFORM CROSS FLOW NO. I8 )
390  WRITE (6,400)
400  FORMAT (1H 5X, 24H TRANSFORMED COORDINATES // )
    1 12X 1HX 13X 1HY 13X 2HX 12X 2HY 12X 2HVZ 12X 3HPHI //)
410  WRITE (6,420) I,XP(I),YP(I),VPX(I,L),VY(I,L),VZ(I,L),PHI(I,L)
420  FORMAT (1H I3, 6F14.7)
    I=I+1
    IF (I.GT.NP) GO TO 450
    IF (T.LE.LCTR) GO TO 410
    LCTR=LCTR+45
450  CONTINUE
450  CONTINUE
500  RETURN
C END
**PROGRAM EXCRS**

**COMMON** /IPSF/ PSF

**COMMON** IEDR(10), CASEF, N8, NNU

1. FLG03, FLG04, FLG05, FLG06, FLG07

2. FLG08, FLG09, FLG10, FLG11, FLG12

3. FLG13, FLG14, FLG15, FLG16, FLG17

4. FLG18, FLG19, FLG20, FLG21, FLG22

5. FLG23, FLG24, FLG25, FLG26, FLG27

**COMMON** NT, ND(11), MN, NUNA(5), TYPEA(5),

1. NER1, NER2, NMA, NSIGA, NSIGC,

2. NUNC(5), TYPEC(5), NLF(11), IFC, NSIGEC,

3. TYPEEC(5), NUNEC(5)

**INTEGER** FLG03, FLG04, FLG05, FLG06, FLG07

1. FLG08, FLG09, FLG10, FLG11, FLG12

2. FLG13, FLG14, FLG15, FLG16, FLG17

3. FLG18, FLG19, FLG20, FLG21, FLG22

4. FLG23, FLG24, FLG25, FLG26, FLG27

**REAL** MN

**COMMON** /C4/ X1(100), Y1(100), X2(100), Y2(100), DELS(100),

1. SINA(100), COSA(100), XP(100), YP(100)

**COMMON** /TC/ RB(100,10), SIG(100,5), A(100), B(100),

1. Z(100), PHI(100,5), XN(100,5), T(100,5),

2. T3(100,5), NSIG, NP, NI,

3. SUMV, SUMM(5)

**DIMENSION** VX(100,5), VY(100,5), VZ(100,5), T2(100,5)

**EQUIVALENCE** ( VX(1,1), XN(1,1) ), ( VY(1,1), T(1,1) ),

1. ( VZ(1,1), T3(1,1) ), ( T2(1,1), T(1,1) )
REPEAT 8
IF (FLG0A,EQ,0) GO TO 10
C*** ***TITLE FOR MATRIX PRINT
WRITE (6,150) HDR, CAS, PSF
WRITE (6,8)
8 FORMAT (1H 42H MATRICES A, B, Z BY ROWS * EXTRA CROSS FLOW //)
C*** ***READ EXTRA CROSS SIGMAS
10 DO 20 N = 1, NSIG
20 READ (3) ( SIG(I,N), I = 1, NT )
C*** ***NO. OF MIDPOINTS LOOP
DO 100 I = 1, NT
100 C*** ***READ MATRICES A, B, Z
C*** ***YOU MUST SOLVE POTENTIAL MATRIX FOR EXCROS
READ (8) ( A(J), J = 1, NT ), ( B(J), J = 1, NT ), ( Z(J), J = 1, NT )
C*** ***NO. OF FLOWS LOOP
M = 0
DO 70 N = 1, NSIG
M = M + 2
SA = 0.0
SB = 0.0
SZ = 0.0
C*** ***NO. OF ELEMENTS LOOP
DO 30 J = 1, NT
30 T2(I,N) = SB
T3(I,N) =SZ
40 CONTINUE
IF (FLG0A,EQ,0) GO TO 100
WRITE (6,80) I, A(J), J = 1, NT
80 FORMAT (1H 13H MATRIX A ROW I6// (1H 10F10.5) )
WRITE (6,85) I, B(J), J = 1, NT
85 FORMAT (1H K13H MATRIX B ROW I6// (1H 10F10.5) )
WRITE (6,85) I, Z(J), J = 1, NT
85 FORMAT (1H K13H MATRIX Z ROW I6// (1H 10F10.5) )
WRITE (6,85) I, A(J), J = 1, NT
85 FORMAT (1H K13H MATRIX A ROW I6// (1H 10F10.5) )
WRITE (6,85) I, B(J), J = 1, NT
85 FORMAT (1H K13H MATRIX B ROW I6// (1H 10F10.5) )
WRITE (6,85) I, Z(J), J = 1, NT
85 FORMAT (1H K13H MATRIX Z ROW I6// (1H 10F10.5) )
IF (I.GT.LCTR) GO TO 210
   LCTR = LCTR + 22
   GO TO 140
230  M = M + 1
   N = N + ND(M)
   WRITE (6,240) I, XI(I), YI(I)
240  FORMAT (1H13,2F14,7//)
   I = I + 1
   IF (J.GE.NT) GO TO 250
   GO TO 210
250  CONTINUE
C 252 IF (FLG05.EQ.0) RETURN
   257 IF (FLG05.EQ.0) CONTINUE
C***  ***OFF BODY POINTS
   DD 300 I = 1, NP
C***  ***READ MATRICES X,Y,Z
   READ (A) (/ A(J), J=1, NT, ) ) (B(J), J = 1, NT), (Z(J), J = 1, NT )
   DD 300 N = 1, NSIG
   SX = 0.0
   SY = 0.0
   SP = 0.0
C***  ***NUMBER OF ELEMENTS LOUP
   DD 260 J = 1, NT
   SX = SX + A(J) * SIG(J,N)
   SY = SY + B(J) * SIG(J,N)
   260  SP = SP + Z(J) * SIG(J,N)
   VX(I,N) = SX
   VY(I,N) = SY
   VZ(I,N) = SP
   PHY(T,N) = YP(I) * SP / 2.0
300  CONTINUE
C***  ***PRINT EXTRA CROSS FLOW (OFF-BODY) OUTPUT
   DD 450 L = 1, NSIG
   KEC = L
   I = 1
   KEC = L
   I = 1
LCTR = 45
340 WRITE (6,150) HEDR,CASE,PSF
   IF (FLG22,GT,0) GO TO 355
   WRITE (6,350) NUNEC(KEC)
350 FORMAT (43H OFF BODY NON-UNIFORM EXTRA CROSS FLOW NO. I8)
   GO TO 390
355 WRITE (6,357)
357 FORMAT (68H OFF BODY GENERATED (RESEP) BOUNDARY CONDITIONS)
1A CROSS FLOW)
390 WRITE (6,400)
400 FORMAT (1H 5X, 24H TRANSFORMED COORDINATES //)
   1 12X 1HX 13X 1HY 13X 2HVX 12X 2MVY 12X 2HVZ 12X 3PHI //)
410 WRITE (6,420) I,XP(I),YP(I),VX(I,L),VY(I,L),VZ(I,L),PHI(I,L)
420 FORMAT (1H I8, 6F14.7)
   I = I + 1
   IF (I,GTE,NP) GO TO 450
   IF (I,LTE,LCTR) GO TO 410
   LCTR = LCTR + 45
   GO TO 340
450 CONTINUE
C  RETURN
501 CONTINUE
C   FND
OVERLAY(A,XY,5,0)
SURROGATE BOUNDL
PROGRAM BOUNDL

*************** PROGRAM K90A ***************

SOLUTION OF THE 2-D, YAWED WING, AND AXISYMMETRIC, COMPRESSIBLE
BOUNDARY LAYER EQUATIONS USING KELLER'S BOX METHOD

*******************************************************************************
*******************************************************************************
*******************************************************************************

****************** FORMULATED BY I. CERECI ****************************
****************** AERODYNAMICS RESEARCH GROUP ****************************
******* DOUGLAS AIRCRAFT DIVISION, MCDONNELL-DOUGLAS CORP. *******
****************** LONG BEACH, CALIFORNIA ********************

-----------------------------------------------------------------------

COMMON NX,NP,NNPR,JI,IT,NRVP,LS,A,NP1,J11,J1M1,NNT,NXT,NXW,NXM
1 ,TITLE(15)
COMMON LG16,LG17,LG18,LG32,LG40
1 ,IG10,IG10,IGOW,IGN,IGCV,IGEG,IGNP,IGRC,IGTR
COMMON /HEADR/ CASE, IPAGE
COMMON/RCLS/F(100,2),U(100,2),V(100,2),ETA(100),DELETA(100)
COMMON/RCLS/XI(100),XS(100),ETAINF(100),BETA(100)
COMMON/BL11/VWPR,VWPR,DELV1
COMMON/BL13/FPRLN
COMMON /BL16/ NTYPES, IOUT
COMMON/RCLS5/EM(100,2),EDV(100,2),E(100,2),EB(100,2),VPRT(100)
COMMON/RCLS7/VGP ,DETA1
COMMON/RCLS8/A(100),CFL(100)
COMMON/BL10/SWP,TANL,WE,W(100,2),WP(100,2)
COMMON/BL12/TI,RMI,UI,RI,PR,PRT,FK,RL,REUI,RIHDI,PSI,ME
1 ,UE(100),RO(100),TW(100),QW(100),RP(100),FW(100)
2 ,RR(100),TE(100),RHODE(100),RMUE(100),GW(100),GPW(100)
3 ,RF1(100),RF2(100),YS(100),IGX1(100),FPW(100),RUL(100)
IF(NX LT, NXT) IGOL = 1
IF(NX GF, NXT) IGOT = 1

C ----
IGCV = 0
CALL FINF
IF(LG3P NE, 1 OR, NX EQ, 1) CALL HEAD
IF(LG32, EQ, 1) GO TO 130
WRITE(6, A015) NX, BETA(NX), XI(NX), XS(NX), ETAINF(NX)
GO TO 138

130 IF(NX EQ, 1) WRITE(6, 7000)
LC = LC+3
IF(LC LT, LCMAX) GO TO 135
CALL HEAD
LC = 0

135 WRITE(6, A015) NX, BETA(NX), XI(NX), XS(NX), ETAINF(NX)
LC = LC+2

138 IF(LSP EQ, 1) GO TO 700

C ----
IF(NX EQ, 1) CALL IVPF
IF(LSP EQ, 1) GO TO 700

C ----
IF(NX EQ, 1) CALL FLPR
IF(LSP EQ, 1) GO TO 700

C ----
IF(NX NE, 1 OR, XS(NX) EQ, 0) GO TO 140
CALL EDVS
IF(LSP EQ, 1) GO TO 700

C ----
140 IF(NX EQ, 1) GO TO 145
CALL SHFT
IF(LSP EQ, 1) GO TO 700
CALL FLPR
IF(LSP EQ, 1) GO TO 700

C ----
145 IT = 0
LC = LC+2
IF(NX=NXT)150,142,150
142 ITC=1
150 IT=IT+1
LC = LC+1
IF(IT.LT.9) GO TO 220
IF(ITC.EQ.0) GO TO 200
WRITE(6,6000)
ITC=0
IT=1
GO TO 220
200 WRITE(6,6010)
LSP=1
GO TO 700
220 CALL EDVS
   IF(LSP.EQ.1) GO TO 700
   CALL MDMX
300 IF(IGOL,EQ.1) GO TO 540
   IF(IGOT,EQ.1) GO TO 550
540 IF(ABS(DELV1).LT.EPSLN) GO TO 600
   IF(V(1,2).LT.0.0 .AND. LG16.NF.0) GO TO 670
   GO TO 150
550 EAG= DELV1/((V(1,2)+VPRI)*.05)
   IF(ABS(EAG).LT.0.02) GO TO 600
   IF(IGTR.EQ.1 .OR. V(1,2) .LE. 0.) GO TO 150
IGCV=1
   CALL EINF
   IF(NX.EQ.1) GO TO 150
   IF(LSP.EQ.1) GO TO 700
   IF(IGRC.EQ.0) GO TO 150
   CALL SHFT
   CALL FLPR
   GO TO 145
C----
600 WRITE(6,6030) V(1,2)
LC = LC+1
IF(IGTR,GT,1) IGTR=0
IGCV=1
CALL EINF
LC=LC+3
IF(LSP,EQ,1) GO TO 700
IF(IGRC,GT,0) GO TO 670
CALL SHFT
CALL FLPR
IF(LSP, EQ, 1) GO TO 700
LC = LC+2
GO TO 145

C ----
670 CALL OTPT
IF(NX, EQ, NXM) GO TO 700

C ----
IF(IGOT, EQ, 1) GO TO 100
IF(NX, GT, 1 AND LG16, NE, 0) CALL TRANS
IF(IGTR, GT, 1) IGRC=1
IF(LSP, EQ, 1) GO TO 700
IF(IGTR, GT, 1) GO TO 120

C ----
GO TO 100
700 IF(NX, EQ, 0) GO TO 800
IOUT=1
CALL OTPT
800 IF(LSP, EQ, 1) WRITE(6,9999)

C----
5010 FORMAT(I3,39X,11)
6000 FORMAT(1H,20X,43HCONVERGENCE IS SLOW - ITERATIONS CONTINUING,/) BOUN 141
6010 FORMAT(1H,15X,45H*** ITERATIONS EXCEED THE ALLOWABLE LIMIT ***/) BOUN 142
6015 FORMAT(1H0,///5X,8HSTATION=,I3,3X,SHBETA=E16.9,2X,3HXI=E16.9,2X, BOUN 143
   12HS=E16.9,2X,8HETAINF=E16.9,///)
6030 FORMAT(1H,32X,E20.9)
7000 FORMAT(1H,45X,20HOUTPUT IN SHORT FORM,/) BOUN 144
BOUN 145
BOUN 146
BOUN 147
BOUN 148
BOUN 149
BOUN 150
BOUN 151
BOUN 152
BOUN 153
BOUN 154
BOUN 155
BOUN 156
BOUN 157
BOUN 158
BOUN 159
BOUN 160
BOUN 161
BOUN 162
BOUN 163
BOUN 164
BOUN 165
BOUN 166
BOUN 167
BOUN 168
BOUN 169
BOUN 170
BOUN 171
BOUN 172
BOUN 173
BOUN 174
BOUN 175
7015 FORMAT(1H0, 5X, 8HSTATION=, I3, 3X, 5HRAFTA=, E16.9, 2X, 3HXI=, E16.9, 2X, 2HS
1=, E16.9, 2X, 8HETATNF =, E16.9 )
9999 FORMAT(1H0/45X, 37H********** CASE TERMINATED ********** // )
C RETURN
1000 CONTINUE
END
SUBROUTINE INPT

** SUBROUTINE INPT **
THIS SUBROUTINE PROCESSES ALL THE INPUT DATA TO THE PROGRAM

COMMON NX, NP, NPPR, JI, IT, NRVP, LSP, NPM1, JT1, JIM1, NTC, NXT, NXW, NXM
1 TITLE(15)
COMMON LG16, LG17, LG18, LG32, LG40
1 IGOL, IGOT, IGON, IGCV, IGFQ, IGNP, IGRC, IGTR
COMMON /HEDR/, CASE, IPAGE
COMMON /BLC7/VGP, DETA1
COMMON /BLC3/XI(100), XS(100), ETAiNF(100), BETA(100)
COMMON /BL12/TI, RMI, UI, RI, PR, PRT, FK, RL, RMUI, RH01, PSI, HE
1 UE(100), RO(100), TW(100), GW(100), RP(100), FW(100)
2 PS(100), TE(100), RHU(100), RMUE(100), GW(100), GPW(100)
3 RF1(100), RF2(100), VS(100), IGX1(100), FPW(100), ROL(100)

COMMON /BL13/ EPSLN
COMMON /BL15/ NXY
COMMON /RADIUS/ ROMAX
DIMENSION SC(301), XR(301), DUEDX(100); PE(100)
DATA DATA1/1,4/; DATA2/6035,0/

EPSLN = .005
PR = .72

C—— READ CASE DATA
READ(5,5005) TITLE, CASE
READ(5,5010) NXT, LG16, LG17, LG18, LG32, LG26, LG40, LG41
READ(5,5020) TI, RMI, UI, FK, RL, RI
READ(5,5025) ROMAX, DFTA1, VGP
IF(LG40 .EQ. 0) GO TO 7000
READ(5,1013) NXT
READ(5,1014) (XS(I), I=1,NXW)
READ(5,1014) (YS(I), I=1,NXW)
READ(S,1014) (UE(I),I=1,NXM)
GO TO 7001
7000 CONTINUE
READ(3) NXM
READ(3) (XS(I),I=1,NXM)
READ(3) (YS(I),I=1,NXM)
READ(3) (UE(I),I=1,NXM)
7001 CONTINUE
NXY=NXM
DO 50 I=1,NXM
RO(I)=YS(I)
GW(I)=0,0
RF(I)=0,0
FW(I)=0,0
GW(I)=0,0
GPW(I)=0,0
RF(I)=0,0
TW(I)=0,0
RP(I)=0,0
FPW(I)=0,0
BR(I)=0,0
50 IGK(I)=0,0
ETAINF(1)=6,0
ETAINF(2)=10,0
NXNS=NXM
85 CALL HEAD
WRITE(6,2050) TITLE,CASE
WRITE(6,2500) LG16, LG17, LG18, LG32, NXT
150 XS1=0,0
SD1=0,0
160 DO 180 I=2,NXM
SDA1=(XS(I)-XS(I-1))**2+(YS(I)-YS(I-1))**2
SQDA1=SQRT(SDA1)
SD2=SD1+ABS(SQDA1)
S(2*I+1)=SD2
180
SD1=SD2
IF(I,EQ,2) S(3)=XS1
180 CONTINUE
WRITE(2, NXM)
WRITE(2, (S(2*I+1), I=1, NXM))
LC = 1
LCMAX = 36
WRITE(6, 2550)
DO 185 I=1,NXM
IF(LC .LT. LCMAX) GO TO 182
CALL HEAD
WRITE(6, 2550)
LC = 1
LCMAX = 49
182 XBG = XS(I) * RL
SBG = S(2*I+1) * RL
WRITE(6, 3100) I, XS(I), YS(I), XBG, SBG, S(2*I+1)
LC = LC+1
IF(FK .EQ. 1) R0(I) = YS(I) * RL
YS(I) = XS(I)
XS(I) = SBG
185 CONTINUE
IF(LCMAX .EQ. 36 .AND. LC .GE. 18) CALL HEAD
IF(LCMAX .EQ. 49 .AND. LC .GE. 45) CALL HEAD
IF(FK .EQ. 0 .OR. LG18 .NE. 1) GO TO 203
CALL SLOPE(NXM, XS, YS, RF1, 1)
IF(RL .EQ. 0) GO TO 203
DO 202 I=1,NXM
RF1(I) = RF1(I) * RL
202 CONTINUE
IF(UI, NE, 0 .OR. RMI, NE, 0 .OR. ) GO TO 204
WRITE(6, 9030)
LSP = 1
GO TO 1800
204 IF(TT, NE, 0 .OR. ) GO TO 205
IF(LG41 .EQ. 0) GO TO 201
WRITE(6,9045)
TI=519.0*(5/9.)
GO TO 205
201 CONTINUE
WRITE(6,9040)
TI = 519.0
205 IF(RMI .NE. 0.) UI = 0,
   IF(LG41 .EQ. 1) GO TO 206
   IF(UI .NE. 0.) RMI=UI/( SQRT(TI)*49.1)
   IF(RMI .NE. 0.) UI=RMI* SQRT(TI)*49.1
   RMI= 1.0E-06*(9.0311276E-03*TIR*1.238522= (.56843634E-06*TIR*TIR
   1.+.38312556E-03*TIR*1.436156)**.5)
   RMI=RMI*UI/UIT
   PSI=RHIOI*TIA1718.0
   RMI2=RMI*RMI
   DK1=RMI2*(DATA1=1.0)*.5
   UEA4=DATA1*RMI2*.5
   SHI=DATA2*TIR
207 HE = SHI + .5*(UI**2)
   GO TO 209
206 TIR=TI*9.0/5.
   UII=UI/3.048
   IF(UI .NE. 0.) RMI=UI/( SQRT(TIR)*49.1)
   IF(RMI .NE. 0.) UI=(RMI* SQRT(TIR)*49.1)*.3048
   RMI= 1.0E-06*(-9.0311276E-03*TIR*1.238522= (.56843634E-06*TIR*TIR
   1.+.38312556E-03*TIR*1.436156)**.5)
   RMI=RMI*UI/UIT
   PSI=RHIOI*TIR1718.0
   SHI=DATA2*TIR
   HF=SHI+.5*(UI**2)
   RMI=RHMI*47.88025
   RMI=RHMI*515.379
   PSI=PSI*47.88025
   HE=HF*.3048*.5048
209 CONTINUE
   IF(LG26 .EQ. 2) GO TO 270
   IF(LG26 .EQ. 3) GO TO 210
   WRITE(6,9050)
   LSP=1
   GO TO 1800
210 CONTINUE
208 DO 220 I=1,NXM
   UE(I)=UI*SQRT(1.0+UE(I))
220 CONTINUE
   GO TO 300
270 DO 280 I=1,NXM
   UE(I)=UE(I)*UI
280 CONTINUE
C----
300 CONTINUE
   DO 320 I=1,NXM
   RHOE(I)=RHOI
   RMUE(I)=RMUI
   PF(I) = 0.0
   TE(I) = 0.0
320 CONTINUE
500 IST = 1
   MULT = 0
   CALL SLOPE(NXM,XS,UE,DUEDX,MULT)
C----
510 DO 1500 I=1,NXM
   IF(I .GT. 1) GO TO 520
   VT1=RHOI*RMU1
520 VT2=VT1*UE(I)
   VT3=VT2*UE(I)
   IF(FK,NE,0.) GO TO 550
   VT4=1.0
   VT5=1.0
   VT42=1.0
   GO TO 1500
INPT 141
INPT 142
INPT 143
INPT 144
INPT 145
INPT 146
INPT 147
INPT 148
INPT 149
INPT 150
INPT 151
INPT 152
INPT 153
INPT 154
INPT 155
INPT 156
INPT 157
INPT 158
INPT 159
INPT 160
INPT 161
INPT 162
INPT 163
INPT 164
INPT 165
INPT 166
INPT 167
INPT 168
INPT 169
INPT 170
INPT 171
INPT 172
INPT 173
INPT 174
INPT 175
GO TO 600
550 IF (FK, NE, 1, 0) GO TO 560
   IF (RO(I), NE, 0) GO TO 555
   IF (I, GT, 1) WRITE(6, 9095) I
   R0(I) = 0.0001 * RL
555 VT4 = R0(I) / RL
   VT4 = VT4 * VT4
   VT5 = RL / R0(I)
   GO TO 600
560 WRITE(6, 9060)
   LSP = 1
   GO TO 1800
600 ROL(I) = VT4
   FX2 = VT2 * VT42
   IF (I, EQ, 1) XI(I) = FX2 * XS(I)
   IF (I, EQ, 1) GO TO 680

650 XI(I) = XI(I-1) + (FX1 + FX2) * (XS(I) - XS(I-1)) * 0.5
680 FX1 = FX2
   IF (I, NE, 1) GO TO 930
   IF (IST, EQ, 0) GO TO 1500
   BETA(1) = 1.0
   IF (FK, EQ, 1) BETA(I) = BETA(1) / 2.
   GO TO 1500
930 IF (I, EQ, 2) GO TO 950
   BETA(I) = 2.0 * XI(I) / (VT3 * VT42) * DUEDX(I)
   GO TO 1500
950 BETA(I) = 2.0 * XI(I) * (UE(I) - UE(I-1)) / (VT3 * VT42 * (XS(I) - XS(I-1)))
1500 CONTINUE
C=====
   WRITE(6, 2600) DETAI, RL, PR, VGP, RHOI, SNP, FK, RMUI, KE, FPSLN
1   ,UI, TI, RI, RMI
CALL HEAD
WRITF(6,2900)
WRITF(6,3000)
LC=1
LCMAX=45
DO 1550 I=1,NXM
IF(LC.LT.LCMAX) GO TO 1520
CALL HEAD
WRITF(6,3000)
LC=1
LCMAX=49
1520 IF(FK .NE. 0.) GO TO 1530
WRITE(6,3200) I,YS(I),RO(I),TW(I),UE(I),PE(I),BR(I)
GO TO 1540
1530 WRITF(6,3200) I,YS(I),ROL(I),TW(I),UE(I),PF(I),BR(I)
1540 CP=1.-(UE(I)*UE(I)/(UI*UI))
WRITF(6,3250) XS(I),RF1(I),GW(I),CP, RMUE(I), FPW(I)
RMACH = 0.
WRITE(6,3250) BETA(I),RF2(I),RP(I),RMACH,TE(I),XI(I)
LC = LC+4
1550 CONTINUE
1590 IGXI=0
IF(XI(I).GE.0.) GO TO 1600
WRITF(6,9070)
IGXI=1
1600 IF(NXM .NE. 1) RETURN
DO 1700 I=2,NXM
IF(XI(I).GT.0.) GO TO 1620
WRITE(6,9080) I
IGXI=1
1620 IF(XI(I).GT.XI(I-1)) GO TO 1700
WRITF(6,9090) I
IGXI=1
1700 CONTINUE
IF(10GXI.EQ.0) RETURN
INPT 246
ISP=1
INPT 247
1A00 RETURN
INPT 248
C = - - -
INPT 249
1013 FORMAT(14)
INPT 250
1014 FORMAT(6F10,0)
INPT 251
250 FORMAT(1H1,25X,15A4,10X,6HCASE,4A4///1H1,54X,10HCASE DATA ///)
INPT 252
2500 FORMAT(1H0,10X,8HTRFLAG =,I1,10X,7HTRINT =,I1,10X,5HTVC =,I1,10X,
INPT 253
110X,ALSHORTP =,I1///1H1,30X,31HTRANSITION SPECIFIED AT STATION,I4
INPT 254
11)
2550 FORMAT(1H1,50X,19HBODY GEOMETRY DATA /
INPT 255
11H0,21X,1HK,9X,3H/C,15X,3MY/C,16X,1HX,17X,1HS,16X,3HS/C /)
INPT 256
2600 FORMAT(1H0/1H0,40X,43HREFERENCE QUANTITIES AND CONTROL PARAMETERS,
INPT 257
A /1H0,16X, 6HH1 = ,F9.5, 16X,8HC = ,F15.7, 10X,
INPT 258
1 AHPRO = ,F9.5, /1H0,16X, 6HK = ,F9.5, 16X,8HRMOFB = ,
INPT 259
2 E15.7, 10X,8HSWEEP = ,F9.4 /1H0,16X, 6HHK = ,F9.5, 16X ,
INPT 260
3 AHMREF = ,E15.7, 10X,8HHE = ,E15.7 /1H0,16X, 6HHEP1 = ,
INPT 261
4 F15.7, 10X,8HVRFF = ,E15.7, 10X, 8HTRFF = ,F10.3 /1H0,16X, 262
5 31X, 8HREY = ,E15.7, 10X,8HMRFF = ,E15.7/)
INPT 263
2900 FORMAT(1H1,50X,12HSTATION DATA ///)
INPT 264
3000 FORMAT(1H0/1H0,7X,1HXN,12X,3H/C,13X,4HRO/C,14X,2HTW,16X,2HUE,15X,2HPE,
INPT 265
1 14X,2HFW///1H1,20X,4H'S,12X,6HALPHA1,13X,2HOU,16X,2MCP,14X,
INPT 266
2 3MHUF,13X,3HFPM///1H1,20X,4HBF1,12X,6HALPHA2,13X,2MRR,16X,
INPT 267
3 2HME,15X,2HTE,12X,5HSQUIG,/) 268
INPT 269
3100 FORMAT(1H1,19X,13,53X,15X,7))
INPT 270
3200 FORMAT(1H1,1H1,1H1,3X,6E17.6)
INPT 271
3250 FORMAT(1H1,11X,6E17.6)
INPT 272
5005 FORMAT(15A4,A4)
INPT 273
5010 FORMAT(14,111)
INPT 274
5020 FORMAT(5F10,0,E12.6)
INPT 275
5025 FORMAT(3F10,0)
INPT 276
5030 FORMAT(2F8.0,6E0.5,F6.0,F6.0,F7.0,2F6.0,F7.0,8X,F4.0,11)
INPT 277
5040 FORMAT(3F14.9)
INPT 278
9004 FORMAT(1H1,37H*ERROR = INPUT REFERENCE LENGTH = 0 ,/)
INPT 279
9005 FORMAT(1H1,51H*ERROR = INPUT SURFACE DISTANCE AT STATION 1 LT 0,)
INPT 280
9006 FORMAT(1H,66H**ERROR - INPUT SURFACE DISTANCE NOT IN ASCENDING ORDER AT STATION, I3 //)
9010 FORMAT(1H, 51H**ERROR - INPUT X NOT IN ASCENDING ORDER AT STATION, I3 //)
9020 FORMAT(1H,51H**ERROR - INPUT NTR OR S AT STATION 1 ARE INCORRECT)
9030 FORMAT(1H,42H**ERROR - N INPUT FOR EITHER VREF OR MREF)
9040 FORMAT(1H,52H**WARNING - TREF INPUT = 0., VALUE RESET TO 519., 1)
9045 FORMAT(1H,67H**WARNING - TREF INPUT = 0., VALUE RESET TO 288.3
133 DEGREES KFLVIN )
9050 FORMAT(1H, 27H**ERROR - CHECK PCOM INPUT )
9060 FORMAT(1H,48H**ERROR - INPUT FLOW INDEX NOT EQUAL TO 1. OR 0. )
9070 FORMAT(1H,38H**ERROR - XI AT STATION 1 NE OR GT 0. )
9080 FORMAT(1H,25H**ERROR - XI AT STATION 1.12H IS NEGATIVE)
9090 FORMAT(1H,25H**ERROR - XI AT STATION 1.26H IS NOT IN ASCENDING ORDER)
9095 FORMAT(1H,41H**WARNING - ROIC INPUT=0. AT STATION 1, I3, 1
23H - VALUE RESET TO .0001 )
9100 FORMAT(1H, 40H. 25HINGS NOT SUCCESSFUL 1ER = ,I2,2X,10HAT STATION, 1
1X,13/H ,12X,1HI,15X,3HX/C,22X,3HY/C, //)
9150 FORMAT(1H,47H**ERROR - INPUT CP EXCEEDS ALLOWABLE LIMITS OF, 1
2E17 6,1X, 10HAT STATION, I3 //)
9200 FORMAT(1H,11X,13,2E20.9)
9300 FORMAT(1H,40H. 25HINSR NOT SUCCESSFUL 1ER = ,I2,2X,10HAT STATION, 1
1X,13/1H ,12X,1HI,15X,3HXIC,22X,3HYC, //)
FND
SUBROUTINE EINF
C **
C THIS SUBROUTINE CALCULATES THE TRANSFORMED Y = GRID POINTS
C
COMMON NX,NP,NPPR,JIT,IT,NRVP,LSP,NPM1,J1,JHM1,NTC,NX1,NXM
1 ,TITLE(15)
COMMON LG16,LG17,LG18,LG32,LG40
1 ,T,G0L,T,G0T,IG0W,IG0N,IGCV,TGEG,IGNP,IGHC,IGTR
COMMON/BLC1/F(100,2),V(100,2),V(100,2),ETA(100),DELETA(100)
COMMON/BLC3/XI(100),XS(100),ETAINF(100),BETA(100)
COMMON/BLC7/VGP,DETA1

IF(IGCV,EQ,1) GO TO 30
30 IF(NX,EQ,1) GO TO 50
IF(NX,EQ,2) GO TO 250
IF(IGNP,EQ,1) GO TO 300
IF(IGNP,EQ,0) GO TO 800
WRITE(6,9910)
LSP=1
GO TO 1800

50 IF(IGNP,EQ,0) GO TO 1000
IF(IGNP,EQ,1) GO TO 100
WRITE(6,9920)
LSP=1
GO TO 1800

100 DO 120 J=2,NP
IF(ABS(V(J,2)),LT,1.E-5) GO TO 500
120 CONTINUE
DO 140 J=2,NP
IF(ABS(V(J,2)),LT,1.E-6) GO TO 500
IF(U(J,2),GE.,999999) GO TO 500
IF(U(J,2),LT.0.) GO TO 400
140 CONTINUE
WRITE(6,9930)
J = NP
GO TO 530

C -----
250 IF(IIGNP, EQ, 0) GO TO 1000
300 DO 320 J=JI,NP
   IF( ABS(V(J,2)), LT, 1., E=5) GO TO 500
   IF( ABS(V(J,2)), LT, 1., E=6) GO TO 500
320 CONTINUE
   DO 340 J=JI,NP
   IF( ABS(V(J,2)), LT, 1., E=4) GO TO 500
   IF(J, EQ, NP) GO TO 330
   IF(U(J,2),GE.,999999) GO TO 500
330 IF(U(J,2),LT.0.) GO TO 400
340 CONTINUE

C -----
   WRITE(6,9980)
   IGRC=1
   ETAINF(NX)=ETAINF(NX)+10.
   GO TO 1010

C -----
400 WRITE(6,9940) J
   LSP=1
   GO TO 1800

C -----
500 IF(IGTR, GT, 1) GO TO 600
530 ETAINF(NX)=ETA(J)
   JI=J
   IIGNP=0
   WRITE(6,6010) ETAINF(NX)
   GO TO 1800
600 IGTR=0
RETURN

C ----

1000 K = NX + 1
   IF (K .EQ. NST) GO TO 620
   IF (ETAINF(NX+1) .GT. ETAINF(NX+2)) .AND. (IGTR .LE. 1) GO TO 850
   ETAINF(NX) = ETAINF(NX+1) + 2,

200 IF (IGTR .EQ. 1) ETAINF(NX) = ETAINF(NX+1) + 10,
   GO TO 1000

850 ETAINF(NX) = ETAINF(NX-2) + (XI(NX) - XI(NX-2)) / (XI(NX+1) - XI(NX+2)) * 
   1(ETAINF(NX+1) - ETAINF(NX+2))

C ----

1000 IF (NX .GT. 1) NPPR = NP
1010 IF (VGP .EQ. 1) GO TO 1020
   ARGLOG = 1, + ETAINF(NX) / DETA1 * (VGP = 1,
   DLOG1 = ALOG(ARGLOG)
   ARGINT = 1, + DLOG1 / ALOG(VGP)
   NP = INT(ARGINT) + 1
   GO TO 1050

1020 ARGINT = ETAINF(NX) / DETA1 + 1,
   NP = INT(ARGINT)

1050 NP1 = NP - 1
   IF (NP .LE. 100) GO TO 1060
   WRITE (6, 9970) NX, NP
   LSP = 1
   GO TO 1800

1060 ETA(1) = 0,
   DELTA(1) = DETA1
   M = 1
   M1 = M + 1
   NP = M * (NP = 1) + 1
   NP1 = NP - 1
   DO 1080 J = M1, NP, M
   N = J - M + 1
   ETA(J) = DELTA1 + VGP * ETA(N-1)
   DELTA(J-1) = ETA(J) - ETA(N-1)
IF(M .EQ. 1) GO TO 1080
DELETA(J-1) = DELETA(J-1)/M
FFA(J-1) = FTA(J) - DELETA(J-1)
DELETA(J-2) = DELETA(J-1)
IF(M .EQ. 2) GO TO 1080
FFA(J-2) = FTA(J-1) - DELETA(J-2)
DELETA(J-3) = DELETA(J-1)
1080 CONTINUE
IF(VGP .NE. 1.) ETAINF(NX) = ETA(NP)
TGNP = 1
1800 RETURN
C ----
6010 FORMAT(1H /16X,10H* ETAE = ,F10.6)
9910 FORMAT(1H ,22H** ERROR AT FTAE(1)**)
9920 FORMAT(1H ,22H** ERROR AT ETAE(2)**)
9930 FORMAT(1H0,49H** WARNING - INPUT ETAE AT STATION IS TOO SMALL /
1 1H ,4RH** CALCULATIONS CONTINUING WITH THE INPUT ETAINF )
9940 FORMAT(1H ,1X,39H**ERROR - FP PROFILE IS NEGATIVE AT I =, I3)
9970 FORMAT(1H ,1X,16H**ERROR - IMAX (, I3, 20H) EXCEEDS 100 =IMAX =, I3)
9980 FORMAT(1H ,3RH** WARNING - ETAE IS BEING REESTIMATED )
C2000 CONTINUE
FND
EINF
SUBROUTINE IVPF

** SUBROUTINE IVPF

THIS SUBROUTINE GENERATES THE INITIAL VELOCITY PROFILE

COMMON NX,NP,NPPR,JI,JI,NRVP,LSP,NPM1,J1,J1,NTC,NXT,NXW,NXM

1 TITLE(15)

COMMON/BLC1/F(100,2),U(100,2),V(100,2),ETA(100),DELET(100)

COMMON/BLC3/XI(100),XS(100),ETAINF(100),BETA(100)

COMMON/BLS/EM(100,2),EDV(100,2),E(100,2),EB(100,2),VPRT(100)

--- POLHAUSEN INITIAL VELOCITY PROFILE

IF(IT.LE.1.AND.NX.EQ.1) E(1,1) = 1.

IF(1,2)=0.

U(1,2)=0.

V(1,2)=0.

DO 50 J=2,NP

EDVE=ETA(J)/ETAINF(1)

EDVE2=EDVE*EDVE

EDVE3=EDVE2*EDVE

EDVE4=EDVE3*EDVE

FPGT = (.5*EDVE+.75*EDVE2-.2*EDVE3)*ETA(J)**2*ETAINF(1)*BETA(1)/6.

F(J,2)=EDVE2*ETAINF(1)**(1.**0.5*EDVE2+0.2*EDVE3) + FPGT

UPGT = EDVE*(1.-3.*EDVE2-4.*EDVE3)*ETAINF(1)**2*BETA(1)/6.

U(J,2)=2.*EDVE2*EDVE4 + UPGT

VPRT = (1.-6.*EDVE+9.*EDVE2-4.*EDVE3)*ETAINF(1)*BETA(1)/6.

V(J,2)=2.*ETAINF(1)**(1.-3.*EDVE2+7.*EDVE3) + VPRT

50 CONTINUE

1000 RETURN

END
SURROUTINE FLPR

**C**

**C** **SURROUTINE FLPR**

**C** CALCULATE THE FLUID PROPERTIES AND THE TVC TERM

**C**

**COMMON** NX,NP,NPP,JI,IT,NRVP,LSP,NP1,JI1,JI11,NTC,NX1,NX,NXM

**COMMON** LG16,LG17,LG18,LG32,LG40

**COMMON**/BL1/F(100,2),U(100,2),V(100,2),ETA(100),DELTA(100)

**COMMON**/BL2/XI(100),XS(100),ETA(100),BETA(100)

**COMMON**/BL12/IMI/UI/RI/PR/PRT/FK/RL/RMUI/RMUI/PSI/HE

**COMMON**/BL12/IMI/UI/RI/PR/PRT/FK/RL/RMUI/RMUI/PSI/HE

A1 = 0,

IF(XI(NX) .GT. 0.) A1 = SQRT(2.*XI(NX))/( RHOI *UE(NX))

G(NP,2) = 1.0

SUM = 0.

F1 = RF1(NX)

TVCT(1) = 1.0

DO 500 I=2,NP

IF(LG18.EQ.0) GO TO 450

F2 = RF1(NX)

SUM = SUM + (F1+F2)/2.*DELTA(I=1)

TVCT(I) = SQRT(1. + 2.*A1*SUM/(KL*RUL(NX)*RDL(NX)))

F1 = F2

GO TO 500

C---- IF NO TVC THEN TVC = TERM = 1.0
**SUBROUTINE EDVS**

**THIS SUBROUTINE COMPUTES THE EDHY VISCOSITY**

```plaintext
COMMON NX, NP, NPPR, JI, IT, NRVP, LSP, NPM1, JII, JIM1, NXT, NXW, NXM
1 TITLE(15)
COMMON LG16, LG17, LG18, LG32, LG40
1 IGO1, IGOL, IGOT, IGOW, IGON, IGCV, IGEC, IGEC, IGRC, IGTR
COMMON/BLC1/F(100,2), U(100,2), V(100,2), ETA(100), DELTA(100)
COMMON/BLC3/XI(100), XS(100), ETAINF(100), BETA(100)
COMMON/BLC5/FM(100,2), EDV(100,2), E(100,2), FB(100,2), VPRT(100)
COMMON/BL10/SHP, TANL, WEW(100,2), WP(100,2)
COMMON/BL12/YI, RMI1, UI, RI, PR, PRT, FK, KL, RMUI1, RMOT1, PSI1, HE
1 , UE(100), RO(100), TW(100), GW(100), RP(100), FW(100)
2 , RR(100), TE(100), RMUE(100), RMUE(100), GW(100), GPW(100)
3 , RF1(100), RF2(100), VS(100), IGX1(100), FPW(100), ROL(100)
COMMON/BL19/C(100,2), G(100,2), GP(100,2),
1 RHO(100), RMU(100), TVCT(100)
COMMON/BL20/ RTHTR, UEIN, ROIN, GAMAT
DIMENSION EDVI(100), EDVU(100), EDVW(100)
DATA DATA1/0168/, DATA2/40/, DATA3/080/, DATA4/44/
```

```plaintext
IF(IGOL, EQ, 1) GO TO 500
SQXT = SQRT(2. * XI(NX))
TC = RHO1 * UE(NX) * ROL(NX) / SQXT
IF(IGOT, EQ, 1) GO TO 600
```

--- FPS FOR LAMINAR FLOWS

500 DO 520 J = 1, NP
EM(J, 2) = 1.0
F0V(J, 2) = 0.
VPRT(J) = PRT
520 CONTINUE
C---- EPS FOR TURBULENT FLOWS

600  SUM = 0.
     SUMT = 0.
     F1 = 1.0
     F3 = 0.
     DO 620 J=1, NP
         EM(J,2) = 1.0
         EDV(J,2) = 0.
         IF(J .EQ. 1) GO TO 620
         F2 = (1.0 - U(J,2))/TVCT(J)
         F4 = F2 * U(J,2)
         SUM = SUM + (F1+F2)/2.0 * DEleta(J=1)
         SUMT = SUMT + (F3+F4)/2.0 * DEleta(J=1)
         F1 = F2
         F3 = F4
     620 CONTINUE

     RTHI = UEF(NX) *(RHOMU/RMU1) * SUMT/TC
     IF(RTHI .LT. 425.0) GO TO 720
     IF(RTHI .GT. 6000.0) GO TO 750
     XPHI = RTHI/425.0 - 1.0
     CPHI = 55.0 * EXP(-0.243 * SQRT(XPHI) - 0.298 * XPHI))
     GO TO 770
     720 CPHI = 0.0
     GO TO 770
     750 CPHI = 55
     770 DATA1 = DATA1 *(1.55/(1.0 + CPHI))

E----

IFLGF0 = 0
        IF(IT .LE. 1) E(1,2) = E(1,1)
        J = 1
        SUM1 = 0.
        F1 = 1.0
        VWPSQT = V(1,2)
        DU1WY = TC*UEF(NX) * ABS(VWPSQT)
TAUIW = RMUI *DUDYW *E(1,2)
USTAR = SQRT(TAUIW/RMUI)
1025 DPDX = -TC*TP*UE(NX)*RMUI *BETA(NX)
PPLUS = -RMUI *DPDZ/(RHOI*RHOI)
*USTAR*USTAR*USTAR
A = 26
ARMT = 1.1*11.8*PPLUS
1030 CONTINUE
1045 APLUS = A/SQRT(ARMT)
EDV0(J) = DATAN*UE(NX)*(RHOI/RMUI) * ABS(SUM/TC)
VPRT(J) = PRT
IF (IFLGED .EQ. 1) GO TO 1098
F2 = 1./TVCT(J)
IF(J .EQ. 1) GO TO 1060
SUM1 = SUM1 + (F1+F2)/2.*DELETA(J=1)
F1 = F2
1060 Y = SUM1 /TC
IF(TVCT(J) .GT. 1.005) Y = R0(NX)*ALOG(TVCT(J))
VPLUS = Y*USTAR*RHOI/RMUI
YA = YPLUS/APLUS
EL = DATA2*Y
IF(YA .LT. 20.) EL = EL *(1.0-EXP(-YAYA))
BPLUS = 34.
IF(J .NE. 1) GO TO 1065
VPRT(J) = DATA2/DATA4 * BPLUS/APLUS
GO TO 1070
1065 VPRT(J) = EL/(DATA4*Y)
YB = YA*APLUS/BPLUS
IF(YB .LT. 20.) VPRT(J) = VPRT(J)/(1.0-EXP(-YB))
1070 VPSQTV=V(J,2)
DUDY = TC*UE(NX)*ABS(VPSQTV) / F2
EDV1(J) = EL*EL*DUDY*RHOI/RMUI * TVCT(J)
IF(EDV1(J) .LT. EDV(J)) GO TO 1100
IFLGED=1
IF (J .LE. 2) WRITE(6,9030)
1098 EDV(J,2)=EDV0(J)
GO TO 1200
1100 EDV(J,2) = EDVI(J)
1200 J = J + 1
   IF (J .LE. NP) GO TO 1030
C----
   IF(IFLGED,EQ,1) GO TO 2050
   WRITE(6,9020)
   EDV(J,2) = EDVO(J)
2050 IF(LG17,EQ,0 .OR. IGTR,EQ,2) GO TO 2500
   UR = UE(NX)**RHOI/RHUI
   IF(NX .GT. NXT) GO TO 2150
   F1 = 0.
   SUMT = 0.
   DO 2100 J=2,NP
   F2 = U(J,2)*(1.0-U(J,2))
   SUMT = SUMT + (F1+F2)*DELETA(J=1)*.5
   F1 = F2
2100 CONTINUE
   RTHTR = UR*SUMT/TX
   IF(LG16,NE,0) GO TO 2300
   UEIN = 0.
   ROTN = 0.
   GO TO 2300
2150 IF(IT .GT. 1) GO TO 2500
   UEIN = UFIN + .5*(1.0/UE(NX))+(1.0/UE(NX-1)))*XS(NX)=XS(NX-1)
   IF(FK .EQ. 0.) GO TO 2200
   RAIN = RAIN + .5*(1.0/R0(NX))+(1.0/R0(NX-1)))*XS(NX)=XS(NX-1)
   GO TO 2300
2200 ROTN = XS(NX)=XS(NX)
2300 ATR = 60.
   GTR = ((UR/ATR)**2)*UE(NX)/(RTHTR**2.487)
   ARFXP = GTR*UEIN*RAIN
   IF(FK .NE. 0.) ARFXP = ARFXP*R0(NXT)
   IF(ARFXP .GT. 10.) GO TO 2500
   GAMAT = 1.0*EXP(-ARFXP)
WRITE(6,9500) GAMAT
2500 DO 2550 J=1,NP
   IF(J.GT.1) GO TO 2510
   EDVM(1) = EDV(1,2)
   GO TO 2550
2510 IF(J.EQ.NP) GO TO 2520
   FDVM(J) = (EDV(J-1,2)+EDV(J,2)+EDV(J+1,2))/3.0
   GO TO 2550
2520 EDVM(NP) = (FDV(NP-2,2)+EDV(NP-1,2)+EDV(NP,2))/3.0
2550 CONTINUE
   DO 2560 J=1,NP
      EDV(J,2)=EDVM(J)
      IF(LG17.EQ.0,OR, IGTR.EQ.2) GO TO 2560
      EDV(J,2) = EDV(J,2)*GAMAT
   END
2560 CONTINUE
3000 E(1,2) = (EM(1,2)+EDV(1,2))
   DO 3010 J=2,NP
      E(J,2) = (EM(J,2)+EDV(J,2))
      TVCT(J)*TVCT(J)
      EB(J-1,2) = 0.5* (E(J,2) + E(J,1,2))
3010 CONTINUE
1800 RETURN
C----
9010 FORMAT(1H,30X,43H**PLUS EXCEPTION THE LAMINARIZATION LIMIT **)
9020 FORMAT(1H,30X,45H**NOTE = EPS DISTRIBUTION = EPS(INNER) ONLY**)
9030 FORMAT(1H,30X,45H**NOTE = EPS DISTRIBUTION = EPS(OUTER) ONLY**)
9500 FORMAT(1H,41X,11H GAMMA(TR) =, E17.6)
END
E(J,1)=E(JI,2)
F8(J-1,1)=EB(JIM1,2)
90 CONTINUE
IF (IGRC,EQ,0) GO TO 100
IGRC=0
GO TO 1800

C -----
100 DO 120 J=JII,JP
   F(J,2)=PHI+ETA(J)
   U(J,2)=1.
   V(J,2)=0.
120 CONTINUE
IF (IGRC,EQ,1) GO TO 80
GO TO 1800

200 DO 220 J=1,JI
210 F(J,?)=F(J,1)
   U(J,2)=U(J,1)
   V(J,2)=V(J,1)
215 EDV(J,2)=EDV(J,1)
   F(J,2)=F(J,1)
   IF (J.EQ.JI) GO TO 220
   EB(J,2)=EB(J,1)
220 CONTINUE
   PHI=F(JI,2)=ETA INF(NX=1)
   IF (IGTR, GT, 1) PHI = F(JI,2)=ETA INF(NX)
GO TO 100

1800 RETURN
END
SUBROUTINE M0MX

** SUBROUTINE M0MX **

FIND THE SOLUTION OF THE X-MOMENTUM EQUATION

COMMON NX,NP,NPR,JI,IT,NRVPP,LSP,NPM1,JI1,JI1,NTC,NXT,NWX,NXM
1 , TITLE (15)
COMMON/BLC1/F(100,2),U(100,2),V(100,2),ETA(100),DELETA(100)
COMMON LG16,LG17,LG32,LG40
1 , IGOL, IGOT, IGON, IGON, IGEV, IGNP, IGRC, IGTH
COMMON/BLC3/XI(100),XS (100),ETAINF(100),ETA(100)
COMMON/BLC5/FM(100,2),EDV(100,2),E(100,2),EB(100,2),VPRT(100)
COMMON/BLC8/A(100),CEL(100)
COMMON/BLC11/VPWR,VSVP,DELV1
COMMON/BLC12/TI,REI,UI,RI,PR,PR,TK,HL,REUI,ROH1,PSI,HE
1 , UF(100),RO(100),TW(100),GW(100),RP(100),FW(100)
2 , IB(100),TE(100),RMOE(100),RMUE(100),GW(100),GPW(100)
3 , RF1(100),RF2(100),YS (100),IGX1(100),FPW(100),RDL(100)
COMMON/BLC19/C(100,2),G(100,2),GP(100,2)
1 , ROH1(100),RHO(100),TVCT(100)
DIMENSION A1(100),BL1(100),G1(100),D1(100),SF(100),S1(100)
1 , X(100),Y(100),Z(100),DELF(100),DELU(100),DELY(100)

-----

IF(IT,GT,1) GO TO 100
IF(NX,EQ,1) CEL(1)=0,
IF(NX,GT,1) CEL(NX)=2.*XI(NX-1)/(XI(NX)-XI(NX-1))+1,

VWPRV=V(1,1)
UWPRU=U(1,1)

100 VWPRV=V(1,2)
UWPRU=U(1,2)

A(1)=0,
DO 320 J = 2, NP
320 A(J) = DELETE(J=11) / 2,

C -----
DO 900 J = 2, NP
FB = (F(J,2) + F(J=1,2)) * 0.5
VB = (V(J,2) + V(J=1,2)) * 0.5
IF (NX = GT. 1) GO TO 600
CFR = 0,
CUB = 0,
CVB = 0,
GO TO 700
600 CFR = (F(J=1) + F(J=1,1)) * 0.5
CUB = (U(J,1) + U(J=1,1)) * 0.5
CVR = (V(J,1) + V(J=1,1)) * 0.5

C -----
700 TM1 = A(J) /FB(J=1,2)
A(J) = TM1 * ((1. + CEL(NX)) * VB + CEL(NX) * CVB)
R1(J) = 1. + TM1 * ((E (J,2)=E (J=1,2))/DELETE(J=11) + (1.
1 + CEL(NX)) * FB + CEL(NX) * CVB)

C -----
IF (NX ,EQ, 1) RR = 0.,
IF (NX ,GT, 1) RR = (ER(J=1,1) * ((V(J,1) = V(J=1,1)) / DELETE
1 (J=1)) + (((E (J,1) = E (J=1,1)) / DELETE(J=11)) + CFB
2 )+CVB +BETA(NX=1) * 5
3 =CEIL(NX)=CFCB*CVB=CUB*CUB)
S(J)=V(J=1,2)=V(J,2)=DELETE(J=11)/EB(J=1,2)
1 *FB*VB = (BETA(NX)=CEL(NX)) * ((U(J,2) + U(J=1,2)) * 5)*2
2 =CEIL(NX)=CFB*VB = BETA(NX)*5
3 *VH*(E(J,2)=E(J=1,2))/DELETE(J=1) + BETA(NX)*5)

900 CONTINUE
C -----
905 D(2) = = 0.5*(A(2)/EB(1,2)*(BETA(NX)*CEL(NX)*(U(2,2)+U(1,2)))
1 +A(2)*A(2) + B1(2)/A(21)
1 +A(2)*A(2)
G1(2) = = D(2) = 1. / A(2)


```
Y(J) = F(1,2)+F(2,2) + DELTA*(U(2,2)+U(1,2))
X(J) = 0.5* (Z(J) + A(J) * Y(J) + B1(J) * (U(1,2)-U(2,2)))
1 DELTA = ( (V(2,2)+V(1,2))*5 / A(J) )
Z(J) = -X(J)*(V(2,2)+V(1,2)) * (U(1,2)-U(2,2)) / A(J)
DO 950 J = 3, NP
TMRI1 = A(J) * SE(J=1)
TMR21 = A(J) * SE(J=1) + G1(J=1) * (Z(J) - B1(J)) * (U(J,2)+U(J=1,2))
1 )A(J)/ER(J=1,2)*BETA(NX)+CE(NX))
TMR31 = 1 + A(J) * G1(J=1)
D(J) = (A(J) * A(J) * A1(J) = (A(J)**2)*(BETA(NX)+CE(NX)) *
1 (U(J,2)+U(J=1,2))/ER(J=1,2) + B1(J)+
SE(J)= - A(J) = TMR11 + D(J)
G1(J) = (TMR31 + D(J) = 1_s) / A(J)
TMM1 = A(J)*(U(J=1,2)+U(J=2)) + F(J=1,2) + Y(J=1)
TMM2 = S(J) * A1(J) * Y(J=1) = (B1(J) = 2_s) * Z(J=1)
TMM3 = A(J)*(V(J=2)+Z(J=1) + Y(J=1,2)) * (U(J,2)+U(J=1,2))
Y(J) = TMM1 = TMR11 * X(J)
Z(J) = (TMR31 * X(J) = TMM3) / A(J)
950 CONTINUE
```

C ===
DEL(NP) = Y(NP)
DELU(NP=1) = X(NP)
DELV(NP) = Z(NP)
C ===
J = NP
1000 J = J + 1
DELF(J) = Y(J) = SE(J) * DELU(J)
DELU(J=1) = X(J) = D(J) * DELU(J)
DELV(J) = Z(J) = G1(J) * DELU(J)
IF (J = GT, 3) GO TO 1000
DELF(2) = Y(2) = SE(2) * DELU(2)
DELV(2) = Z(2) = G1(2) * DELU(2)
```
DELF(1) = 0,
DELV(1) = X(2) - D(2) * DELU(2)
DELU(1) = 0.

C ----
IF (IT .EQ. 1) WRITE (6, 9510)
WRITE (6, 9521) IT, V(1,2), DELV(1)

C ----
DO 1020 J=1,NP
 IF (J .EQ. NP) GO TO 1010
 U(J,2) = U(J,2) + DELU(J)
1010  F(J,2) = F(J,2) + DELF(J)
 V(J,2) = V(J,2) + DELV(J)
1020 CONTINUE
 DELV1 = DELV(1)
1800 RETURN

C ----
9510 FORMAT (1H0, 21X,1HI,20X,4HFPPW, 26X,5HDELVW )
9521 FORMAT (1H ,20X,I2,10X,E20.9,10X,E20.9)
END
SUBROUTINE TRNS

**SUBROUTINE TRNS**

**THIS SUBROUTINE COMPUTES THE LOCATION OF B, L TRANSITION**

COMMON NX, NP, NPPR, JI, IT, NRV, LSP, NPM1, JI1, JIM1, NTC, NXT, NXW, NXM
   , TITLE(15)
COMMON LG16, LG17, LG18, LG32, LG40
   , IGO, IG0, IGOW, IGON, IGCV, TEGF, IGBP, IGRC, IGTR
COMMON/HEL1/F(100, 2)/U(100, 2)/V(100), ETA(100), DELETA(100)
COMMON/BLC3/XI(100), XS(100), ETAINF(100), BETA(100)
COMMON/BL12/TI, RMI, UT, RI, PR, PK, RL, RMUI, RHODI, PSI, HE
   , UE(100), RO(100), TW(100), GW(100), RP(100), FW(100)
   , RR(100), TE(100), RHOD(100), RMUE(100), G(00), GPW(100)
   , RF(100), RF2(100), VE(100), IGX1(100), FPW(100), RDL(100)
COMMON /BL14/ RX1, RTH1, CF0, CF1, CF2, CFSUM0, CFSUM,
   , THETA(100), DELS(100), FPPN(100)
COMMON/BL20/ RTHTR, UEIN, ROIN, GAMAT
DIMENSION C(3)
DATA C1, C2, C3, C4, C5, C6 /66.7663, 3.357287, 12.31885, 48447.19,
   , -19886.08, 1.0,

K = NX - 1
IIGR = 0
IF (LG16, NE, 0) GO TO 50
WRITE (6, 9010)
LSP = 1
GO TO 1000
50 IF (V(1, 2), LE, 0.,) GO TO 600
AG1 = UE(NX) * RHODI/RMUI
IF (FK .EQ. 1.) GO TO 55
RX12 = AG1 * XS(NX)
RTH12 = AG1 * THETA(NX)
TRNS 001
TRNS 002
TRNS 003
TRNS 004
TRNS 005
TRNS 006
TRNS 007
TRNS 008
TRNS 009
TRNS 010
TRNS 011
TRNS 012
TRNS 013
TRNS 014
TRNS 015
TRNS 016
TRNS 017
TRNS 018
TRNS 019
TRNS 020
TRNS 021
TRNS 022
TRNS 023
TRNS 024
TRNS 025
TRNS 026
TRNS 027
TRNS 028
TRNS 029
TRNS 030
TRNS 031
TRNS 032
TRNS 033
TRNS 034
TRNS 035
GO TO 57
55 RTH1P=AG1*THETA(NX)*ROL(NX)
   S2=0.
   DO 56 I=2,NX
      DELS3= XS(I)=XS(I-1)
      S2=S2+(ROL(I)**2)*(DEL33)
   RX12=AG1*S2
   CONTINUE
   IF(IGTR,NE,0) GO TO 60
      RX1=1.E-05 * RX12
      RTH1=RTH12
      IGTR=1
   GO TO 1800
56 RX2=1.E-05 * RX12
   RTH2=RTH12
   IF(RX2,LT,1.E) GO TO 550
   RTH9 = C1 + C2*RX2 + C6*SQR(T(C3*RX2*RX2+C4*RX2+C5))
   RTD = RTH9=RTH2
   IF(RTD,GT,0.,AND. RTD,GE,10.) GO TO 550
   IF(ARS(RTD),GF,10.) GO TO 65
      IGTR=3
   RQT1=0.
   RQT2=RX2
   GO TO 95
C -----
65 IGTR=3
   AG1=RTMP=((RTH2=RTH1)/(RX2-RX1))*RX2
   AG2=(RTH2=RTH1)/(RX2-RX1)
   AG3=AG2**3.357287
   AG4=AG1**66.4663
   C(1)=12.31085=AG3*AG3
   C(2)=48047.19=2.*AG3*AG4
   C(3)=-1988.08=AG4*AG4
   RSM4AC = C(2)*C(2) - 4.*C(1)*C(3)
   IF(RSM4AC,GF,0.) GO TO 70

TRNS 036
TRNS 037
TRNS 038
TRNS 039
TRNS 040
TRNS 041
TRNS 042
TRNS 043
TRNS 044
TRNS 045
TRNS 046
TRNS 047
TRNS 048
TRNS 049
TRNS 050
TRNS 051
TRNS 052
TRNS 053
TRNS 054
TRNS 055
TRNS 056
TRNS 057
TRNS 058
TRNS 059
TRNS 060
TRNS 061
TRNS 062
TRNS 063
TRNS 064
TRNS 065
TRNS 066
TRNS 067
TRNS 068
TRNS 069
TRNS 070
WRITE(6,7000) NX
GO TO 550
70  ROT1 = (-C(2) + SQRT(BSMXAC))/2.*C(1)
ROT2 = (-C(2) - SQRT(BSMXAC))/2.*C(1)
IF(ROTI .LE. 0.*AND.ROTI .LE. 0.0) GO TO 550
IF(ROTI .GT. 0.*AND.ROT2 .LT. 0.0) GO TO 80
IF(ROTI .GT. 0.*) RX=1.E05*ROT1
IF(ROT2 .LT. 0.*) RX=1.E05*ROT2
GO TO 100
80  IGR=1
RX=ROT1 * 1.E05
GO TO 100
90  IGR=2
RX=ROT2 * 1.E05
C ----
GO TO 100
95  RX=1.E05*ROT2
XTR=XS(NX)
GO TO 200
100  UETR = UE(NX) + (UE(NX)=UE(K))*(1.E05*RX=RX1)/(RX2=RX1)
XTR=RX*RMUI/(RHOI*UETR)
IF(XTR=XS(NX)) 300,200,500
200  WRITE(6,6010) NX
GO TO 700
300  IF(XTR .LE. XS(K)) GO TO 500
WRITE(6,6020) XTR
GO TO 1000
500  IF(IGR,EQ,1) GO TO 90
550  IF(V(1,2)*LE.0.) GO TO 600
IF(IGR,EQ,0) GO TO 1800
RX1=RX2
RTI1=RTI2
CFN = CF1
CFSUMO = CFSUM
RETURN
600 IGTR=2
   LG17 = 0
   IF(V(1,2),LT,0.) GO TO 800
   WRITE(6,6030) NX
700 NXT=NX
   WRITE(6,6050) NXT
   CF1 = CFO
   CFSUM = CFSUM0
   IF(LG17,LT,0.) GO TO 1800
   ROIN = 0.
   UEIN = 0.
   GO TO 1800
800 XTR=XS(NX)-(XS(NX)=XS(K))* V(1,2)/(V(1,2)=V(1,1))
   WRITE(6,6040) XTR

C ----
1000 NXT=NX
   WRITE(6,6050) NXT
   CF1 = CFO
   CFSUM = CFSUM0
   IF(LG17,LT,0.) GO TO 1800
   ROIN = XS(NX) = XTR
   UEIN = ROIN/UE(NX)
   IF(FK,NE,0.) ROIN = ROIN/RO(NX)
1800 RETURN

C ----
6010 FORMAT(1H1//////////30X,
   1 34HTRANSITION HAS OCCURRED AT STATION, I3/)
6020 FORMAT(1H1//////////30X,
   1 30HTRANSITION HAS OCCURRED AT S =,F12.6 /)
6030 FORMAT(1H1//////////45X,38HLAMINAR SEPARATION OCCURRED AT STATION, I3,
   1 /)
6040 FORMAT(1H1//////////45X,38HLAMINAR SEPARATION OCCURRED AT S =,F12.6)
6050 FORMAT(1H0,35X,33HTURBULENT FLOW STARTED WITH NTR = ,I3 /////)
7000 FORMAT(1H1//////////40X,39HATTEMPT TO FIND X(TR) FAILED AT STATION ,I3/)
9010 FORMAT(1H,24H** ERROR IN TRFLAG INPUT, /)
SUBROUTINE SLOPE(NPX, XC, YC, DYDX, NER)

C ** SURROUTINE SLOPE
C COMPUTE THE DERIVATIVE DYDX FROM X VS Y INPUT
C
DIMENSION XC(150), YC(150), DYDX(150)
DIMENSION X(300), Y(300), XY(301)

XY(1) = NPX
IF(MER .NE. 0) GO TO 20
NP2M1 = NPX
DO 10 I=1,NPX
   X(I) = XC(I)
   Y(I) = YC(I)
10 CONTINUE
   GO TO 60
20 NP2 = 2*NPX
   NP2M1 = NP2-1
   DO 40 I=1,NPX
      XY(I*2) = XC(I)
      XY(2*I+1) = YC(I)
40 CONTINUE
   NLA = 2
   DO 50 I=2,NP2M2
      X(I-1) = XY(I)
      Y(I-1) = XY(I+1)
      IF(I .EQ. NP2) GO TO 50
      XY(I) = (XY(I)+XY(I+2))/2
      CALL INS1(X(I), XY, Y(I), NLA, NER)
50 CONTINUE
   DO 80 I=1,NP2M1
      IF(I .GT. 1) GO TO 100
      DYDX(I) = (Y(I+1)-Y(I))/ (X(I+1)-X(I))
      GO TO 200
80 IF(I .LT. NP2M1) GO TO 150
   DYDX(I) = (Y(I)-Y(I-1))/ (X(I)-X(I-1))
100 GO TO 100
200 CONTINUE
GO TO 200

150 IF(Y(I-1).EQ.Y(I)).AND. Y(I).EQ.Y(I+1)) GO TO 180
   A1 = ((X(I)-X(I+1))/((X(I-1)-X(I))*(X(I-1)-X(I+1))))
   A2 = ((2.*X(I)-X(I+1)-X(I-1))/((X(I-1)-X(I-1))*(X(I)-X(I+1))))
   A3 = ((X(I)-X(I+1))/((X(I+1)-X(I+1))*(X(I+1)-X(I))))
   DYDX(I) = A1*Y(I-1) + A2*Y(I) + A3*Y(I+1)
   GO TO 200

180   DYDX(I) = 0.

200 CONTINUE
   IF(MER.EQ.0) RETURN
   DO 300 I=1,NPX

300   DYDX(I) = DYDX(2*I-1)
   RETURN
END
SUBROUTINE OTPT

** SUBROUTINE OTPT

  OUTPUT THE RESULTS OF THE B, L, CALCULATIONS

  COMMON NX,NP,NPPN,J1,IT,NVNP,LSP,NPM1,J11,J12,NTC,NXT,NXW,NXM
  COMMON LG16, LG17, LG18, LG32, LG40

  COMMON /HEADR/ CASE, IPAGE
  COMMON /BL1/F(100,2),U(100,2),V(100,2),ETA(100),DELETA(100)
  COMMON /BLC3/XI(100),XS(100),ETAINF(100),BETA(100)
  COMMON /BLC5/EM(100,2),EDV(100,2),E(100,2),FB(100,2),VPRT(100)
  COMMON /BL12/T1, RMI, UI, Ri, PR, PRT, FK, RL, RMUI, RHOI, PSI, HE
  COMMON /BL14/RX, RTHI, CFA, CF1, CF2, CF3, RHO(100)

  COMMON /BL16/ NTYP, IDOUT

  COMMON /BL17/ RX(100), CFA(100), CF(100), ETAE(100), CDI(100), ST(100)

  COMMON /BL19/C(100,2), G(100,2), GP(100,2)

  COMMON/RADIUS/ RMAX

  DIMENSION Y(100)

  IF(IDOUT .NE. 0) GO TO 900

  IF(IPRT = (NP+18)/30)  
  IF(NP .LT. 30) IPRT=1
  A1 = 0.

  UEUI = UE(NX)/UI

  THETA(NX) = 0.
DELS(NX) = 0
RX(NX) = XS(NX) * RMUI * UE(NX) / RMUI
THETA = 0
W = 0
CF(NX) = 0
USTUF = 0
YPLUS = 0
UPLUS = 0
IF(NX .EQ. 1) CF1 = 0
IF(NX .EQ. 1) CFSUM = 0
CFA(NX) = 0
ST(NX) = 0
TNP(NX) = NP
ETA(NX) = ETA(NP)
FPPW(NX) = V(1, 2)

!----
IF(XI(NX) .EQ. 0) GO TO 300
A1 = SORT(2, XI(NX)) / ( RMUI * UE(NX) * RL(NX) )
JNP = NP
SUM1 = 0.0
SUM2 = 0.0
F1 = 0.0
F3 = 1.0
DO 90 J = 2, JNP
F2 = U(J, 2) * (1.0 - U(J, 2))
SUM1 = SUM1 + (F1 + F2) / 2.0 * DFETA(J - 1)
F1 = F2
90 CONTINUE

THETA(NX) = SUM1 * A1
DELS(NX) = A1 * (ETA(NX) + F(1, 2) - F(JNP, 2))
RTHETA = RX(NX) * DTHETA(NX) / XS(NX)
W = DELS(NX) / DTHETA(NX)
CF(NX) = SORT(2, XI(NX)) * RMUI * V(1, 2) * RL(NX) * F(1, 2)
CF(NX) = ABS(CF(NX))
USTUF = SORT(CF(NX) / 2.)
IF(FK .GE. 1.0) GO TO 200
IF(NX .EQ. 1) CF1 = CF(NX) * UEUI * UEUI
IF(NX .EQ. 1) GO TO 300
CF2 = CF(NX) * UEUI * UEUI
CFSUM = CFSUM + (CF1 + CF2) * (YS(NX) = YS(NX=1)) * 0.5
IF(XI(NX=1) .EQ. 0.0) CFSUM = 2.0 * THETA(NX)
CFA(NX) = CFSUM / (YS(NX) = YS(1))
IF1 = CF2
GO TO 300
CONTINUE
IF(NX .EQ. 1) CF1 = CF(NX) * UEUI * UEUI * R0(NX)
IF(NX .EQ. 1) GO TO 300
CF2 = CF(NX) * UEUI * UEUI * R0(NX)
CFSUM = CFSUM + (CF1 + CF2) * (YS(NX) = YS(NX=1)) * 0.5 * RL
IF(XI(NX=1) .EQ. 0.0) CFSUM = 2.0 * THETA(NX)
CFA(NX) = CFSUM / (R0MAX * R0MAX)
IF1 = CF2
300 IF(LG32.EQ.1 .AND. IGX1(NX) .EQ. 0) GO TO 420
LCMAX = 42
LC = 60
SUMY = 0.
F1 = 1.0
DO 400 J = 1, NP
IF(LC .LT. LCMAX) GO TO 340
LC = 1
CALL HEAD
WRITE(6, 2000) NX, YS(NX)
IF((LG32.EQ.0 .OR. IGX1(NX) .NE. 0)) WRITE(6, 2100)
400 IF(J .EQ. 1) GO TO 350
F2 = 1.0 / TVCT(J)
SUMY = SUMY + (F1 + F2) / 2.0 * DELTA(J=1)
F1 = F2
350 IF(J .EQ. 1) GO TO 350
F2 = 1.0 / TVCT(J)
SUMY = SUMY + (F1 + F2) / 2.0 * DELTA(J=1)
F1 = F2
350 IF(J .EQ. 1) GO TO 350
F2 = 1.0 / TVCT(J)
SUMY = SUMY + (F1 + F2) / 2.0 * DELTA(J=1)
F1 = F2
350 I = 1 + ((J-1) / IPRT) * IPRT
IF(J .EQ. 1) .AND. J .NE. NP) GO TO 400
Y(J) = SUMY*A1
IF(USTUE.EQ.0) GO TO 370
YPLUS = Y(J)*UE(NX)*RHOE(NX)*USTUE/HRUF(NX)
UPPLUS = U(J,2)/USTUE
LC=LC+1
370 WRITE(6,6060)
1 J,ETA(J),F(J,2),U(J,2),V(J,2),Y(J),YPLUS,UPPLUS,FDV(J,2)
400 CONTINUE
420 CONTINUE

C----
R ----
700 IF(LG32.EQ.1 .AND. IGX1(NX),EQ,0) GO TO 800
WRITE(6,3000)
WRITE(6,3200) NX,XS(NX),IHTHETA(NX),DELS(NX),CF(NX),V(1,2),GW(NX)
WRITE(6,3250) YS(NX), RX(NX), RTHETA, H, CFA(NX), GPW(NX), ST(NX)
800 IF(INOUT.EQ.0) GO TO 1800

C----
900 CALL HEAD
WRITE(6,3500) TITLE
LCMAX=45
LC=1
WRITE(6,4000)
IF(LSP.EQ.1 .AND. NX.GT.1) NX=NX-1
DO 1000 I=1,NX
IF(LC.LT.LCMAX) GOTO 940
CALL HEAD
WRITE(6,4000)
LC=1
940 RTHETA=0.
H=0.
IF(FXI(I),EQ,0) GO TO 950
RTHETA = HX(I)*IHTHETA(I)/XS(I)
H = DELS(I)/THETA(I)
950 WRITE(6,4200) I,XS(I),IHTHETA(I),DELS(I),CF(I),FPPW(I),GW(I),INP(I)
WRITE(6,4250) YS(I), RX(I), RTHETA, H, CFA(I), GPW(I), ST(I), ETA(I)
LC=LC+3
1000 CONTINUE
   DO 1003 I=1, NX
1003 DELS(I)=DELS(I)/RL
   WRITE(?) NZ
   WRITE(?) (DELS(K), K=1, NX)
C THE CALCULATION OF THE BASE DRAG IS DONE AT THIS POINT USING
C A METHOD GIVEN IN HOERNER'S BOOK ON AERODYNAMIC DRAG
   IF(R0(NX) .LT. ROMAX) GO TO 1010
   CDBASE=.029/SQRT(CFA(NX))
   GO TO 1020
   CDBASE=.029*(R0(NX)/ROMAX)**3/SQRT(CFAH)
1020 WRITE(6,1030) CDBASE
1030 FORMAT(1H0, 77W THE BASE DRAG FOR THIS CONFIGURATION BASED ON THE
           1MAXIMUM FRONTAL AREA IS = , F15.8)
1900 RETURN
C = = = =
2000 FORMAT(1H ,3X, 11HSTATION NO.,13,30X, 5H8/C = ,F12.6 /)
2100 FORMAT(1H0,2X,1HI,7X,3HETA,11X,1HF,14X,2HFP,14X,3HFPP,14X,1HY,14X,
   1 5HPLUS,12X,5HPLUS,12X, 4HEPS+ /)
2150 FORMAT(1H0,2X,1HI,7X,3HETA,11X,1HF,14X,2HFP,14X,3HFPP,11X,
   1 7HY/THETA,9X,5HPLUS,12X,5HPLUS,12X, 4HEPS+ /)
2200 FORMAT(1H0,2X,1HI,7X,3HETA,11X,1HF,14X,2HFP,14X,3HFPP,14X,1HY,12X,
   1 4HM/ME,17X,5H WP, 12X,4HEPS+ /)
2300 FORMAT(1H0,2X,1HI,7X,3HETA,11X,1HG,14X,2HGP,14X,2M Y,13X,4H T ,
   1 14X,4H PRT,12X,2HMU,14X,2HM /)
2400 FORMAT(1H0,2X,1HI,7X,3HETA,11X,1HG,14X,2HGP,11X,7HY/THETA,10X,
   1 4MT/TE,11X,10H PRT ,14X, 6HM/U,10X, 4HM/ME, /)
3000 FORMAT(1H0/ 7X, 1HN, 12X, 4H 8 ,13X,5HET/ETA,12X,4HDELS,14X,2HCF,
   1 14X,4HFPPW,14X,2HGW, /1M, 5X, 3H/C,12X,2HRX,13X,6HET/ETA,14X
   2,1HM,15X,3HCF,14X,3HGPW, 14X, 2HST , /)
3200 FORMAT(1H /1H ,17,4X, 6E17.6 )
3250 FORMAT(1H ,F11.5,6E17.6 )
3500 FORMAT(1H,3X,14HOUTPUT SUMMARY,15A4/)

OTPT 141
OTPT 142
OTPT 143
OTPT 144
OTPT 145
OTPT 146
OTPT 147
OTPT 148
OTPT 149
OTPT 150
OTPT 151
OTPT 152
OTPT 153
OTPT 154
OTPT 155
OTPT 156
OTPT 157
OTPT 158
OTPT 159
OTPT 160
OTPT 161
OTPT 162
OTPT 163
OTPT 164
OTPT 165
OTPT 166
OTPT 167
OTPT 168
OTPT 169
OTPT 170
OTPT 171
OTPT 172
OTPT 173
OTPT 174
OTPT 175

OTPT
400 FORMAT(1H0/7X,1HN,12X,4H S,15X,5H HTHETA,12X,4H DELS,14X,2H C,14X,1H0)
1 4H PPM,14X,2HG,15X,4H MAX,1H,5X,3H X/C,12X,2HRX,13X,
2 6HR HTHETA,14X,1HN,15X,3H CFA,14X,3H GPW,14X,2H ST,14X,6H ETAINF,/
4200 FORMAT(1H,1H0,17,4X,6E17.6,8X,1I)
4250 FORMAT(1H,1H0,11,6,6E17.6,4X,1F11.6)
6060 FORMAT(1H,1I,3,2X,1F10.6,7E16.6)
9000 CONTINUE
END
SURROUTINE HEAD

C

C ** SUBROUTINE HEAD
C

COMMON /HEADR/ CASE, IPAGE
WRITE (6,100) CASE
IPAGE = IPAGE + 1
RETURN

100 FORMAT(1H1,'/1M ,2X,6H CASE ,A4,21X,51H***** CEBECI-KELLER BOUNDAR
1Y LAYER PROGRAM *****, 23X , 13HPROGRAM K9QA )

END
PROGRAM SMOOTH
SUBROUTINE SMOOTH

**THE SUBROUTINE CONTROLS THE SMOOTHING OF THE INPUT COORDINATES**

DIMENSION X(NPTS), Y(NPTS), Xu(NPTS), Yo(NPTS)
REIIND 1
READ(5,1) NPTS, ITAPE
IF(ITAPE .NE. 0) GO TO 5
READ(5,1) (X(I), I=1, NPTS)
READ(5,1) (Y(I), I=1, NPTS)
GO TO 7
5 READ(1) (X(I), I=1, NPTS)
READ(1) (Y(I), I=1, NPTS)
7 CONTINUE
CALL SMOPT(X, Y, Xu, Yo, NPTS)
WRITE(10,10) (XO(I), I=1, NPTS)
WRITE(10,10) (YO(I), I=1, NPTS)
REIIND 10
1 FORMAT(214)
2 FORMAT(6F10,0)
10 FORMAT(6F10,6)
RETURN
20 CONTINUE
END
SUBROUTINE SM5PT (XI,YI,X0,Y0,N)

C THIS ROUTINE USES THE OPTIMUM 5 POINT SMOOTHING METHOD. A 3 POINT
C METHOD IS USED AT THE END POINTS.
C
DIMENSION XI(1),YI(1),X0(1),Y0(1)
C
J = 2
I = -1
10 X0(J+I) = XI(J+I)
   Y0(J+I) = YI(J+I)
   X0(J) = 0.25*(XI(J-1) + 2.0*XI(J) + XI(J+1))
   Y0(J) = 0.25*(YI(J-1) + 2.0*YI(J) + YI(J+1))
   IF (I .EQ. 1) GO TO 20
   J = N-1
   I = 1
   GO TO 10
C
20 CONTINUE
   N2 = N + 2
   DO 30 J=3,N2
      X0(J) = (-XI(J-2)+4.0*XI(J-1)+10.0*XI(J)+4.0*XI(J+1)-XI(J+2))*
             1.0/0.0625
      Y0(J) = (-YI(J-2)+4.0*YI(J-1)+10.0*YI(J)+4.0*YI(J+1)-YI(J+2))*
             1.0/0.0625
   30 CONTINUE
C
RETURN
END
OVERLAY(AXSY,1,0)
PROGRAM ITFRAT
C
SUBROUTINE ITERAT
DIMENSION S(100),DELLS(100),X(100),Y(100),SURF(100),SINAL(100),
COSAL(100),SMD(100),DFLSR(201),TSINAL(201),TCOSAL(201),
2 XNEW(100),YNEW(100),DESU(100)
REWIN 15
REWIN 2
C
SURFACE DISTANCES FROM BOUNDARY LAYER INPUT
READ(2) N
READ(2) (S(I),I=1,N)
C
BOUNDARY LAYER DISPLACEMENT THICKNESS
READ(2) NN
READ(2) (DELLS(I),I=1,NN)
C
THIS DO LOOP IS USED IN CASE THE BOUNDARY LAYER DOES NOT
C
CALCULATE A COMPLETE BOUNDARY DISPLACEMENT THICKNESS ARRAY
C
DO 10 I=NN,N
10 DELLs(I) = DELLs(NN)
C
THE COORDINATES OF THE TRANSFORMED BODY ARE READ IN AT THIS POINT
READ(15) NTS
READ(15) (X(I),I=1,NTS)
READ(15) (Y(I),I=1,NTS)
C
THE SURFACE DISTANCE OF THE INPUT BODY COORDINATES ARE CALCULATED
SURF(1)=0,0
DO 30 I=2,NTS
DESURF=SQRT((X(I)-X(I-1))**2+(Y(I)-Y(I-1))**2)
30 SURF(I)=DESURF+SURF(I-1)
S(N)=SURF(NTS)
C
NOTE S(I) IS SURFACE DISTANCE FROM LEADING EDGE TO TRAILING EDGE
C
BASED ON COORDINATES ASSOCIATED WITH THE BOUNDARY LAYER SOLUTION
C
SURF(1) IS SURFACE DISTANCE FROM LEADING EDGE TO TRAILING EDGE
C
BASED ON THE X AND Y COORDINATES OF THE TRANSFORMED ORIGINAL
C
BOD AT WHICH THE VALUE OF DISPLACEMENT THICKNESS IS TO BE ADDED
C
NEXT THE COSINE AND SIN OF THE LOCAL SURFACE ANGLES
ARE FOUND AT THE MIDPOINTS OF THE X AND Y COORDINATES

NTS = NTS - 1
NTSMM = NTS + 1

DO 40 I = 1, NTS

DERSU(I) = SQRT((X(I+1) - X(I))**2 + (Y(I+1) - Y(I))**2)
SINAL(I+1) = (Y(I+1) - Y(I)) / DERSU(I)
COSAL(I+1) = (X(I+1) - X(I)) / DERSU(I)
SINAL(I) = SINAL(2)
COSAL(I) = COSAL(2)
SINAL(NTSMM) = SINAL(NTS)
COSAL(NTSMM) = COSAL(NTS)

THE SURFACE DISTANCE CORRESPONDING TO EACH OF THESE SIN AND COSINE PAIRS ARE CALCULATED HERE

SMO(1) = 0.0
SMO(2) = 0.5 * SURF(2)

DO 50 I = 2, NTS

SMO(I+1) = SMO(I) + 0.5 * (SURF(I+1) - SURF(I-1))
SMO(NTSMM) = SMO(NTS) + 0.5 * (SURF(NTS) - SURF(NTSMM))

DISPLACEMENT THICKNESS AT THE VALUES OF S AND Y ARE FOUND HERE

CALL TABLE1(N, S, DELS, DELSTB)
CALL TABLE1(NTSMM, SMO, SINAL, TSINAL)
CALL TABLE1(NTSMM, SMO, COSAL, TCOSAL)

NEW COORDINATES ARE FORMED HERE

DO 60 I = 1, NTS

SURFF = SURF(I)
CALL INS1(SURFF, DELSTB, DELST, 1, NER)
CALL INS1(SURFF, TSINAL, SINAL, 1, NER)
CALL INS1(SURFF, TCOSAL, COSAL, 1, NER)
XNFW(I) = X(I) - DELST * SINTU
YNFW(I) = Y(I) + DELST * COSU
XNFW(1) = X(1) - DELLS(2)

IF SEPARATION HAS OCCURRED, THE BODY IS MODIFIED TO ACCOUNT FOR THIS BY ADDING A CIRCULAR RADIUS TO CREATE A SEPARATION BUBBLE
IF (NN.N EQ. N) GO TO 60
XNFW1 = XNF (NN - 3)
XNEW2=XNEW(NN=2)
XNEW3=XNEW(NN=1)
YNFW1=YNFW(NN=3)
YNFW2=YNFW(NN=2)
YNFW3=YNFW(NN=1)
CALL CIRCLE(XNEW1,XNEW2,XNEW3,YNFW1,YNFW2,YNFW3,RADIUS,XCENT,
1 YCENT,DYDX)
NNN=NN+1
DO 70 I=NNN,NTS
DELTA=X(I)-XCENT
IF(DELTA,GE,0.,) KM=1
IF(DELTA,GE,0.,) GO TO 90
YCI=(RADIUS **2 - DELTA **2)
IF(DYDX,GE,0.,) GO TO 65
YNFW(I)=YCENT-SQRT(YCI)
GO TO 70
65 YNFW(I)=YCENT+SQRT(YCI)
70 CONTINUE
GO TO 80
90 CONTINUE
DO 100 I=KM,NTS
100 YNEW(I)=YNFW(KM=1)
CONTINUE
C IF THE BODY RADIUS BECOMES EQUAL TO THE DISPLACEMENT THICKNESS AT
C SOME POINT THEN THE NEW BODY IS MODIFIED TO KEEP THE SAME AREA
C DUE TO THE DISPLACEMENT THICKNESS
NXSL=NTS/2
DO 105 K=NXSL,NTS
DYNEW=YNEW(K)-Y(K)
IF(DYNEW,GE,0.,) KSL=K-2
IF(DYNEW,GE,0.,) GO TO 110
105 CONTINUE
GO TO 115
110 DA=3.14159*(YNEW(KSL)**2-Y(KSL)**2)
DO 112 I=KSL,NTS
CONTINUE
ITRT 071
ITRT 072
ITRT 073
ITRT 074
ITRT 075
ITRT 076
ITRT 077
ITRT 078
ITRT 079
ITRT 080
ITRT 081
ITRT 082
ITRT 083
ITRT 084
ITRT 085
ITRT 086
ITRT 087
ITRT 088
ITRT 089
ITRT 090
ITRT 091
ITRT 092
ITRT 093
ITRT 094
ITRT 095
ITRT 096
ITRT 097
ITRT 098
ITRT 099
ITRT 100
ITRT 101
ITRT 102
ITRT 103
ITRT 104
ITRT 105
ARFA=3.14159*(Y(I)+Y(I)+DARFA)
112 YNFW(I)=SORT(ARFA/3.14159)
115 WRITE(6,4)
   WRITE(6,5) (XNEW(I),YNEW(I),I=1,NTS)
C XNEW AND YNEW ARE THE VISCOUS COORDINATES WHICH SHOULD BE WRITTEN
C ON TAPE AND TRANSFERRED TO THE SMOOTHING ROUTINE
REWINI 1
WRITE(1) (XNFW(I),I=1,NTS)
WRITE(1) (YNEW(I),I=1,NTS)
4 FORMAT(1H,10X,4HXNEW,20X,4HYNEW)
5 FORMAT(2F20.8)
RETURN
END
SUBROUTINE TABLF1(N,X,Y,TARLF)
DIMENSION X(100),Y(100),TARLF(201)
C THIS ROUTINE SETS UP TABLES FOR INPUT TO SUBROUTINE INS1
C N IS THE NUMBER OF VALUES OF X AND Y TO BE PUT INTO ARRAYS
    J=1
DO 200 I=1,N
    TARLF(J)=X(I)
    TARLF(J+1)=Y(I)
    J=J+2
200 CONTINUE
    TARLF(1)=N
RETURN
END
SUBROUTINE CIRCLE (X1, X2, X3, Y1, Y2, Y3, R, XCENT, YCENT, DYDX)

XY1=(X1+X2+X3)*Y1
XY2=(X2+X3+Y2)*Y2
XY3=(X3+X1+Y3)*Y3
E1=(XY1-XY2)*(X2-X3)=(XY2-XY3)*(X1-X2)
F2=(Y1-Y2)*(X2-X3)=(Y2-Y3)*(X1-X2)
F=F1/E2
CENTK=F/2
D=((XY2-XY3)-E*(Y2-Y3))/(X2-X3)
H=D/2
F=XY1-D*X1-E*Y1
R=SQRT(H*H+CENTK**2-F)
DYDX=((X3-H)/(Y3-CENTK)
C=1/R
DYDXS=DYDX*DYDX
CC=C*(1-DYDXS)**1.5
CC=ABS(CC)
DDYDX=ABS(DYDX)
PHI=ATAN(DDYDX)
IF(DYDX ,GE, 0) GO TO 20
XCFNT=X3+R*SIN(PHI)
YCENT=Y3+R*COS(PHI)
GO TO 30
20 XCENT=X3+R*SIN(PHI)
YCENT=Y3+R*COS(PHI)
30 CONTINUE
RETURN
END
REFERENCES


FIGURE 1. COORDINATE SYSTEM FOR THE BOUNDARY LAYER ON A BODY OF REVOLUTION

FIGURE 2. EDDY-VISCOSITY DISTRIBUTION ACROSS A BOUNDARY LAYER
FIGURE 3. COORDINATES FOR AXIALLY SYMMETRIC BODY WITH THICK BOUNDARY LAYER

FIGURE 4. EFFECT OF TRANSITION REGION MODIFICATION ON THE SKIN FRICTION
FIGURE 5. TRANSITION CORRELATION CURVE FROM REFERENCE 32
FIGURE 6. COMPARISON OF EXPERIMENTAL AND CALCULATED TRANSITION LOCATIONS FROM VARIOUS METHODS FOR FAVORABLE GRADIENT FLOWS
FIGURE 7. FLOW DIAGRAM OF COMPUTER PROGRAM FOR AXISYMMETRIC ANALYSIS AND DESIGN METHOD (ADAM)
FIGURE 8. SCHEMATIC DIAGRAM OF CYLINDRICAL WAKE SHAPE USED TO MODEL SEPARATION

FIGURE 9. PRESSURE DISTRIBUTION FOR SPHERE IN SUPERCritical REGION USING CYLINDRICAL SEPARATION MODEL
FIGURE 10. SCHEMATIC DIAGRAMS OF CIRCULAR ARC FAIRING USED IN THE MODEL FOR SEPARATED FLOW.
FIGURE 11  AXISYMMETRIC BASE DRAG AS A FUNCTION OF FOREBODY SKIN FRICTION COEFFICIENT
FIGURE 12 COMPARISON OF BASE DRAG CALCULATED BY ADAM TO EXPERIMENTAL DATA

FIGURE 12  COMPARISON OF BASE DRAG CALCULATED BY ADAM TO EXPERIMENTAL DATA
FIGURE 13. SCHEMATIC OF HIGH FINENESS RATIO BODY FROM REFERENCE 35

FIGURE 14. HIGH FINENESS RATIO BODY AND SIMULATED TUNNEL USED IN POTENTIAL FLOW PROGRAM TO ACCOUNT FOR WALL EFFECTS ON PRESSURE DISTRIBUTION
FIGURE 13. EFFECT OF WIND TUNNEL WALLS ON INVISCID PRESSURE DISTRIBUTION FOR HIGH FINENESS RATIO BODY AS CALCULATED BY POTENTIAL FLOW PROGRAM
FIGURE 14. COMPARISON OF CALCULATED "VISCOS" PRESSURE DISTRIBUTION FOR HIGH FINENESS RATIO BODY TO EXPERIMENTAL DATA
FIGURE 15. EQUIVALENT BODY INCLUDING SEPARATED WAKE USED TO CALCULATE "VISCOUS" FLOW ABOUT SPHERE IN SUPERCritical REGIME
FIGURE 16. COMPARISON OF CALCULATED "VISCOUS" PRESSURE DISTRIBUTION FOR SPHERE IN SUPERCritical REGIME TO EXPERIMENTAL DATA
FIGURE 1. EFFECT OF "VISCOUS" MODELING ON CALCULATION OF LOCAL SKIN FRICTION COEFFICIENT FOR SPHERE IN SUPERCRITICAL REGIME
FIGURE 17. EQUIVALENT BODY INCLUDING SEPARATED WAKE USED TO CALCULATE "VISCOUS" FLOW ABOUT SPHERE IN SUBCRITICAL REGIME
FIGURE 18. COMPARISON OF CALCULATED "VISCOUS" PRESSURE DISTRIBUTION FOR SPHERE IN SUBCRITICAL REGIME TO EXPERIMENTAL DATA
FIGURE 1. EFFECT OF "VISCOUS" MODELING ON CALCULATION OF LOCAL SKIN FRICTION COEFFICIENT FOR SPHERE IN SUBCRITICAL REGIME
"The aeronautical and space activities of the United States shall be conducted so as to contribute ... to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons. Also includes conference proceedings with either limited or unlimited distribution.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include final reports of major projects, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546