NASA TECHNICAL MEMORANDUM

NASA TM X-72831

NASA TM X-72831

RAXBOD: A FORTRAN PROGRAM FOR INVISCID TRANSONIC FLOW OVER AXISYMMETRIC BODIES

By James D. Keller and Jerry C. South, Jr.

February 1976

(NASA-TM-X-72831) RAXBOD: A FORTRAN PROGRAM FOR INVISCID TRANSONIC FLOW OVER AXISYMMETRIC BODIES (NASA) 71 p HC $4.50 CSCL 01A

N76-21156

Unclas
G3/02 21583

This informal documentation medium is used to provide accelerated or special release of technical information to selected users. The contents may not meet NASA formal editing and publication standards, may be revised, or may be incorporated in another publication.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LANGLEY RESEARCH CENTER, HAMPTON, VIRGINIA 23665

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM X-72831		

4. Title and Subtitle	5. Report Date
RAXBOD: A FORTRAN PROGRAM FOR INVISCID TRANSONIC FLOW OVER AXISYMMETRIC BODIES	February 1976
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No
James D. Keller and Jerry C. South, Jr.	
9. Performing Organization Name and Address	10. Work Unit No.
NASA-Langley Research Center Hampton, Virginia	505-06-11-02
	11. Contract or Grant No.
12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
National Aeronautics and Space Administration Washington, D. C. 20546	Technical Memorandum
	14. Sponsoring Agency Code

15. Supplementary Notes

Special technical information release, not planned for formal NASA publication.

16. Abstract

A program called RAXBOD is presented for the analysis of steady, inviscid, irrotational, transonic flow over axisymmetric bodies in free air. The method solves the exact equation for the disturbance velocity potential function and applies the exact surface boundary condition. Instructions on program usage and listings of the program and sample cases are given.

17. Key Words (Suggested by Author(s)) (STAR category underlined)	18. Distribution Statement
<u>Aerodynamics</u> Transonic Flow Axisymmetric Flow	Unclassified - unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	69	$4.25

*Available from { The National Technical Information Service, Springfield, Virginia 22151
STIF/NASA Scientific and Technical Information Facility, P.O. Box 33, College Park, MD 20740

RAXBOD: A FORTRAN PROGRAM FOR INVISCID TRANSONIC FLOW OVER AXISYMMETRIC BODIES

By James D. Keller and Jerry C. South, Jr.
Langley Research Center

SUMMARY

A program called RAXBOD is presented for the analysis of steady, inviscid, irrotational, transonic flow over axisymmetric bodies in free air. Instructions on program usage and listings of the program and sample cases are given.

INTRODUCTION

The program described in this report is for the analysis of steady, inviscid, irrotational, transonic flow over axisymmetric bodies in free air. It solves the exact equation for the disturbance velocity potential and uses the exact surface boundary condition. Most of the background about the equations solved and the difference scheme used is given in reference 1. This report gives instructions on the use of the computer program and also some additional details which were not given in reference 1.

The next section gives a general description of the problem and the method of solution. Then the instructions for using the computer program and a description of the inputs and outputs are given. The appendices contain additional details about some specific parts of the program as well as listings of the program and the sample cases.

GENERAL DESCRIPTION

One of the important considerations when trying to solve the full potential equation is the choice of a coordinate system. For complex three-dimensional shapes cartesian coordinates may be best; however, for simpler two-dimensional or axisymmetric shapes the use of a coordinate transformation such that the body lies along a coordinate line can greatly simplify the application of the exact boundary condition at the body surface. The program described in this paper uses a body-normal coordinate system for closed bodies. For open bodies (i.e. bodies with a sting or simulated wake) it uses a body-normal system on the forebody up to the first horizontal tangent and a sheared cylindrical coordinate system aft of that point. This coordinate system is suitable for closed bodies which are blunt on both ends and convex and smooth over the entire body or for open bodies which are blunt-nosed and convex and smooth up to the first horizontal tangent. It is possible to treat pointed bodies and bodies with slope discontinuities but the coordinate system is not well-suited for them and their solution may not be as accurate as the blunt-body solutions.

A stretching is applied to both the normal and tangential coordinates such that the infinite physical space is mapped to a finite computational space. Thus, the boundary condition at infinity can be applied directly and there is no need for an asymptotic far-field solution. Details about the stretching functions are given in appendix A.

The general method of solution is to replace the governing second-order partial differential equation with a system of finite difference equations, including Jameson's "rotated" difference scheme (ref. 2) at supersonic points.

The difference equations are solved by a column relaxation method. In order to get both rapid convergence and sufficient resolution, the relaxation is generally done on three different grids. The difference equations are first solved on a crude grid (about 400 grid points) which yields rapid convergence. Interpolation of this solution is used as an initial condition for a refined grid. This procedure can be repeated to any desired refinement within computer time and storage limitations.

The boundary condition at the body surface is applied through the use of dummy points inside the body. Details of this computation are given in appendix B.

PROGRAM USAGE

The program was written in the FORTRAN programming language for use on a CDC 6600 computer operating under the NOS 1.0 operating system at Langley Research Center. The program is overlaid in order to reduce the computer storage required. One of the overlays uses several subroutines from the Langley Research Center graphics library to create a plot vector file which can then be post-processed in order to obtain plotted output. Some modifications to the program might be required in order to obtain plots on a different computer system.

The input cards for each case are summarized in the following table:

Read Order	Variables	Format
1	DESC	8A10
2	IXY	16I5
3	XO(I), I = 1, IXY	8E10.3
4	YO(I), I = 1, IXY	8E10.3
5	DYDXN, DYDXT, YMAX, XREF	8E10.3
6	IMAX, JMAX, MIT, MHALF, KLOSE, NPLOT	16I5
7	RF1, COVERG, QF3	8E10.3
8	DNDYO, ALF, DXIDXO, XM, CXM, DXIDXM	8E10.3
9	GAM, AMINF	8E10.3

The definitions of these input variables are as follows:

DESC - Description of case. Up to 80 alphanumeric characters. Appears on printed and plotted output.

IXY - The number of coordinate pairs used to describe the body. Presently limited to 100.

XO - Input coordinates in the axial direction - 8 per card.

YO - Input coordinates in the radial direction - 8 per card.

DYDXN - Body slope, $\frac{dy}{dx}$, at the nose. If it is infinite (as it is for blunt bodies) put in a value greater than 900.

DYDXT - Body slope at the tail (with proper sign). If it is infinite put in a value greater than 900.

YMAX - Maximum body radius. Used to calculate the reference area in computing the drag coefficient.

XREF - Body reference length. Used for scaling plots. XREF will scale to 5 inches on plots.

IMAX - Number of grid lines in the tangential direction. I = 1 is the forward stagnation line. I = IMAX is the rear stagnation line for closed bodies and downstream infinity for open bodies. For each grid refinement IMAX is increased such that $IMAX_{NEW} = 2\,(IMAX_{OLD}) - 1$. The present limit on IMAX is 81.

JMAX - Number of grid lines in the normal direction. J = 1 corresponds to an infinite distance from the body and J = JMAX is on the body. The same formula and limit that apply to IMAX also apply to JMAX.

MIT - Maximum number of iterations (complete relaxation cycles) allowed on the first grid. MIT is doubled for each grid refinement.

MHALF - Number of grid refinements to be done.

KLOSE - Body type.

= 0 for open body (i.e. one with a sting or wake).

= 1 for closed body.

NPLOT - Plot trigger. NPLOT = 1 causes write on disc for input to plot routines and calling of plot routines.

RF1 - Relaxation factor for subsonic points. Usual value is about 1.4. Should be in the range $0<RF1<2$. The program automatically reduces RF1 by 10 percent if: (1) The maximum correction, averaged over 10 cycles, is greater than that for the previous 10 cycles, and (2) the last maximum residual occurred at a subsonic point.

COVERG - Convergence criterion control parameter. Usual value is 1. Iterations stop when the maximum residual is less than $COVERG/(IMAX-1)^2$. This criterion is the order of the finite difference truncation error for subsonic points. If this degree of accuracy is not required, COVERG can be made larger.

QF3 - Supersonic damping factor for improving iterative stability (at the expense of a slower convergence rate). Usual value is 0.1, but many cases with subsonic free streams are successful with QF3 = 0. Definitely need some QF3 on fine meshes with supersonic free streams. Note that QF3 has no effect on the accuracy of the converged solution, only on the stability and convergence rate. QF3 is automatically increased if: (1) The maximum correction, averaged over 10 cycles, is greater than that for the previous 10 cycles, and (2) the last maximum residual is at a supersonic point.

DNDYO - Derivative of the normal coordinate stretching function at the body, $\left(\frac{d\eta}{dY}\right)_{Y=0}$. The value of DNDYO can be determined by choosing the desired step size for the first grid next to the body, $\Delta\eta_0$. Then $\left(\frac{d\eta}{dY}\right)_{Y=0} = \frac{\Delta\eta_0(1-\Delta Y)^{\alpha}}{\Delta Y}$ where $\Delta Y = 1/(\text{JMAX}-1)$. See Appendix A.

ALF - Exponent in the normal coordinate stretching function, α. Usual value is 1.3. Larger values of ALF move the last finite value of η farther away from the body and smaller values move it closer. See Appendix A.

DXIDXO - Derivative of the tangential coordinate stretching function at the nose, $\left(\frac{d\xi}{dX}\right)_{X=0}$. Since $\Delta X = 1/(\text{IMAX}-1)$ then $\Delta\xi_0 \approx \text{DXIDXO}/(\text{IMAX} - 1)$, which can be used to determine what value of DXIDXO to use. It is usually best to use $\Delta\xi_0 \approx \Delta\eta_0$. The above relation for $\Delta\xi_0$ is only approximate however, and it might be necessary to adjust DXIDXO to get the desired $\Delta\xi_0$. See Appendix A.

XM - Axial location, x_m, (in physical coordinates) of the junction (or matching point) between the two tangential stretching functions, for open bodies only. See Appendix A. Usual value about the same as the body length.

CXM - Value of the computational coordinate, X, at the matching point of the two stretching functions (for open bodies only). Since X varies from zero to one, CXM is the fraction of the total number of grid points which will be in the first stretching region (ahead of x_m). Usual value is about 0.75.

DXIDXM - Derivative of the tangential stretching function at the matching point, $\left(\frac{d\xi}{dx}\right)_{X=X_m}$. $(\Delta\xi)_{X=X_m} \approx$ DXIDXM/(IMAX-1). Used only for open bodies. See Appendix A.

GAM - Ratio of specific heats.

AMINF - Free stream Mach number.

The Program Output is Described Below:

1.) Listing of body geometry input.

2.) Other input values.

3.) Computed geometric parameters in tangential direction.

I - Tangential grid index.

S - Arc length along reference surface.

X - Axial coordinate.

Y - Radial coordinate.

THET - Angle of reference coordinate surface, θ. For closed bodies θ is the same as the body angle, θ_B. For open bodies $\theta = \theta_B$ on the forebody and $\theta = 0$ on the afterbody.

THETB - Body angle, θ_B.

AK - Surface curvature on closed bodies. For open bodies AK is the surface curvature on the forebody and $AK = -\frac{d^2y}{dx^2}$ on the afterbody.

F - Derivative of the tangential stretch function, $\frac{dX}{d\xi}$.

4.) Computed geometric parameters in normal direction.

J - Normal grid index.

AN - Normal coordinate, η.

G - Stretching function derivative, $G(J) = \left(\frac{dY}{d\eta}\right)_j$.

GH - Stretching function derivative at half intervals,

$$GH(J) = \left(\frac{dY}{d\eta}\right)_{j + 1/2}.$$

5.) Iteration history.

IT - Iteration number.

DPMAX - Maximum ϕ correction, $\max_{ij} \left| \phi_{ij}^{IT} - \phi_{ij}^{IT - 1} \right|$

ID, JD - I, J location of DPMAX.

RMAX - Maximum residual, $\max_{ij} \left| R_{ij} \right|$, where R_{ij} is the right hand side of the difference equation (with ΔX^2, ΔY^2, etc. in denominator).

IR, JR - I, J location of RMAX.

ISUB, ISUP - Indicates if maximum residual occurred at a subsonic or supersonic point.

RAVG - Average value of the residual.

RF1 - Relaxation factor for subsonic points.

QF3 - Damping factor for supersonic points.

NS - Number of supersonic points.

SEC/CY - Time for iteration cycle.

6.) Time for iterations.

7.) Tabulated values of C_p and Mach number on the body.

8.) Drag coefficient by trapezoidal and Simpson integration of the C_p's.

9.) Rough plot of C_p along the body. This plot is distorted in the axial direction because it is for equal spacing in the computational space. The asterisks show the level of sonic C_p.

10.) Mach number chart of the flow field in the computational plane.

Numbers printed are the Mach number multiplied by 100. I values from top to bottom. J values from left to right.

11.) x and y coordinates of the sonic line.

APPENDIX A

COORDINATE STRETCHING FUNCTIONS

The normal coordinate stretching function is:

$$\eta = \frac{AY}{(1-Y)^{\alpha}}$$

where η is the physical coordinate normal to the body and Y is the computational coordinate which varies from zero at the body to one at infinity. The constant A controls the physical step size at the body, $A = \left(\frac{d\eta}{dY}\right)_{Y=0}$, and for a given value of A, the exponent α controls the size of the last finite value of η. Larger values of α move points farther away from the body.

The tangential coordinate stretching is a transformation between the physical arc length along the reference surface, ξ, and the computational coordinate, X, which varies from zero to one. For closed bodies the transformation is

$$\xi = \frac{\xi_{max}}{2} + (X - \tfrac{1}{2})\left[A + B\,(X - \tfrac{1}{2})^2\right]$$

where A and B are determined by specifying $\left(\frac{d\xi}{dX}\right)_{X=0}$ and requiring that $\xi = \xi_{max}$ at X = 1. These conditions give $A = \frac{3\,\xi_{max} - \left(\frac{d\xi}{dX}\right)_{X=0}}{2}$ and $B = 4\,(\xi_{max} - A)$.

For open bodies the tangential coordinate stretching is divided into two regions with the physical location of the dividing point, x_m, being an input quantity. Also input is the value of the computational coordinate at the dividing point, X_m. Since the computational coordinate varies from zero to one, X_m is equivalent to the fraction of the coordinates which are upstream of x_m. The stretching function for the region from the nose up to x_m is

$$\xi = a_1 X + a_2 X^3 + a_3 X^5 + a_4 X^7 \qquad 0 \le X \le X_m$$

In the region from x_m to infinity the stretching function is

$$\xi = \xi_m + b \frac{(X-X_m)(1-X_m)}{1-X} \qquad X_m \le X < 1$$

The coefficients in these expressions are determined by specifying ξ_m, $\left(\frac{d\xi}{dX}\right)_{X=0}$, and $\left(\frac{d\xi}{dX}\right)_{X=X_m}$ and requiring that $\frac{d\xi}{dX}$ and $\frac{d^2\xi}{dX^2}$ be continuous at $X = X_m$. These conditions give

$$a_1 = \left(\frac{d\xi}{dX}\right)_{X=0} \qquad b = \left(\frac{d\xi}{dX}\right)_{X=X_m}$$

$$a_2 = \frac{70C_1 - 22C_2 + 2C_3}{16 X_m^2}$$

$$a_3 = \frac{-84C_1 + 36C_2 - 4C_3}{16 X_m^4}$$

$$a_4 = \frac{30C_1 - 14C_2 + 2C_3}{16 X_m^6}$$

where
$$C_1 = \frac{\xi_m - a_1 X_m}{X_m}$$

$$C_2 = b - a_1$$

and
$$C_3 = \frac{2X_m b}{1 - X_m}$$

APPENDIX B

APPLICATION OF SURFACE BOUNDARY CONDITION IN REGION OF SHEARED CYLINDRICAL COORDINATES

The boundary condition in the sheared cylindrical coordinates is

$$V - y_B' \, U = 0 \tag{B1}$$

where

$$U = 1 + \phi_\xi - y_B' \, \phi_\eta \tag{B2}$$

$$V = \phi_\eta \tag{B3}$$

and y_B' is the body slope.

This boundary condition (B1) can be rearranged to give:

$$\phi_\eta = \frac{y_B'}{y + y_B'^2} (1 + \phi_\xi) \tag{B4}$$

Let

$$\frac{y_B'}{y + y_B'^2} = w_2$$

and introduce $\phi_\eta = g\phi_Y$ and $\phi_\xi = f\phi_X$ to get:

$$g\phi_Y = w_2(1 + f\phi_X)$$

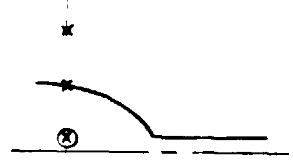
Let

$$w_2 \, (1 + f\phi_X) = DPO$$

so that

$$g\phi_Y = DPO \tag{B5}$$

First consider "ordinary" dummy points which lie inside the body and above the axis (i.e. $\eta\big|_{y=-\Delta Y} > -y_B$ or $Y\big|_{\eta=-Y_B} < -\Delta Y$) as shown in the following sketch:



The values of the potential function at ordinary dummy points are computed by first letting $\phi_Y = \dfrac{\phi_{i,\,JMAX-1} - \phi_{i,\,JMAX+1}}{2\Delta Y}$

which can be put into the boundary condition (B5) to give

$$\phi_{i,JMAX+1} = \phi_{i,JMAX-1} - \frac{2\Delta Y}{g} \, DPO \tag{B6}$$

This result can be expressed in the more general form (which will be needed later):

$$\phi_{i,JMAX+1} = w_3 \, \phi_{i,JMAX-1} + w_4 \, \phi_{i,JMAX} - w_5 \, DPO \tag{B7}$$

by letting $w_3 = 1$, $w_4 = 0$, and $w_5 = \dfrac{2\Delta Y}{g}$

In cases where the physical location of the dummy point is below the axis, the boundary condition is handled differently. Because the flow field is axisymmetric, the potential at a point below the axis is the same as that for a point an equal distance above the axis, as shown in the following sketches:

Let Y_1 be the value of the computational coordinate at the dummy point whose potential is desired. A Taylor series expansion for ϕ at this point (which is the same as $\phi_{i,JMAX+1}$) yields:

$$\phi_{i,\ JMAX+1} = \phi_{i,\ JMAX} + Y_1\ \phi_Y + \frac{Y_1^2}{2}\ \phi_{YY}$$

also

$$\phi_{i,\ JMAX-1} = \phi_{i,\ JMAX} + \Delta Y \phi_Y + \frac{\Delta Y^2}{2}\ \phi_{YY}$$

Eliminate ϕ_{YY} from these equations and solve for $\phi_{i,JMAX+1}$ to get

$$\phi_{i,JMAX+1} = \frac{Y_1^2}{\Delta Y^2}\ \phi_{i,JMAX-1} + \left(1 - \frac{Y_1^2}{\Delta Y^2}\right)\phi_{i,JMAX} + Y_1\left(1 - \frac{Y_1}{\Delta Y}\right)\phi$$

Now since $\phi_Y = \dfrac{DPO}{g}$, this can be put into the form

$$\phi_{i,JMAX+1} = w_3\ \phi_{i,JMAX-1} + w_4\ \phi_{i,JMAX} - w_5\ DPO$$

where

$$w_3 = \left(\frac{Y_1}{\Delta Y}\right)^2,\quad w_4 = 1 - \left(\frac{Y_1}{\Delta Y}\right)^2,\quad w_5 = -\ \frac{Y_1}{g}\left(1 - \frac{Y_1}{\Delta Y}\right)$$

If Y_a is the (negative) value of the computational coordinate that corresponds to the location of the axis, then $Y_1 = \Delta Y + 2Y_a$.

Y_a can be found from the stretching function. The stretching function is $\eta = \frac{AY}{(1-Y)^\alpha}$ or $\frac{\eta}{A} = Y\,(1-Y)^{-\alpha}$ which can be expanded in a series for small Y to give:

$$\frac{\eta}{A} = Y + \alpha Y^2 + \frac{\alpha(\alpha+1)}{2}\, Y^3 + \frac{\alpha(\alpha+1)(\alpha+2)}{6}\, Y^4 + \ldots$$

A reversion of this series gives

$$Y = \frac{\eta}{A} - \alpha\left(\frac{\eta}{A}\right)^2 + \frac{\alpha(3\alpha-1)}{2}\left(\frac{\eta}{A}\right)^3 - \frac{\alpha(16\alpha^2 - 12\alpha + 2)}{6}\left(\frac{\eta}{A}\right)^4 + \ldots$$

Putting $\eta = -y_B$ into this gives the value of Y_a.

APPENDIX C

PROGRAM LISTING

```
      OVERLAY(JERRY,0,0)
      PROGRAM RAXBOD3(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE4)
*************************************************************************
*                                                                       *
* RELAXATION SOLUTION OF EXACT EQUATION FOR DISTURBANCE VELOCITY        *
*              POTENTIAL FOR AXISYMMETRIC TRANSONIC FLOW                *
*              (COORDINATE INPUT VERSION)                               *
*                                                                       *
*      PROGRAMMED BY JERRY C. SOUTH, JR. AND JAMES D. KELLER            *
*                                                                       *
*                                                                       *
*************************************************************************
      CALL OVERLAY(5HJERRY,1,0)
      CALL OVERLAY(5HJERRY,2,0)
      STOP
      END
      OVERLAY(JERRY,1,0)
      PROGRAM ONE0
C
C     *******************************************************************
C
C     IMPORTANT. WHEN I-DIMENSION IS CHANGED, ID MUST BE SET EQUAL TO
C     NEW I-DIMENSION.
C
C     FINAL VALUE OF IMAX, AFTER ALL GRID-HALVING IS COMPLETED, IS
C     IMAX(FINAL)=(IMAX(INPUT)-1)*(2**MHALF)+1
C     SIMILARLY FOR JMAX(FINAL).
C
C     1ST DIMENSION OF P-ARRAY MUST BE AT LEAST AS BIG AS IMAX(FINAL)
C     2ND DIMENSION OF P-ARRAY MUST BE AT LEAST AS BIG AS JMAX(FINAL)+1
C     19 I-ARRAYS DIMENSIONED AT LEAST AS BIG AS IMAX(FINAL)
C     12 J-ARRAYS DIMENSIONED AT LEAST AS BIG AS JMAX(FINAL)
C
C     XS AND YS ARE SONIC PT. COORDS. NO NEED TO CHANGE DIMENSION UNLESS
C     MORE THAN 398 SONIC PTS ARE EXPECTED(VERY UNLIKELY).  SUBROUTINE
C     SONLIN PREVENTS CALCULATION OF MORE THAN 398 SONIC PTS.
C
C     IXY IS THE NUMBER OF INPUT COORDINATES USED TO DESCRIBE THE BODY.
C     COMMON BLOK6 CONTAINS 9 ARRAYS DIMENSIONED AT LEAST AS BIG AS IXY.
C     PROGRAM ONE1 CONTAINS 4 ARRAYS AT LEAST AS BIG AS IXY AND
C     5 ARRAYS AT LEAST AS BIG AS IXY+1.
C
C     *******************************************************************
C
```


ORIGINAL PAGE IS
OF POOR QUALITY

```
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPO(81),F(81),AK(81),S(81),LS(81),FM(81)
      COMMON AN(81),G(81),GH(81),CB(81),D(81),X1(81),X2(81),H(81),HR(81)
     1,HRP(81),HRM(81),HRMM(81)
      COMMON XS(400; YS(400)
      COMMON ID,ANMAX,DNDYO,YMAX,CD,RMSQ,JSKP,TSP
      COMMON /BLOK1/ XST
      COMMON /BLOK2/ PI,RAD
      COMMON /BLOK3/ IMAX,JMAX,C2,RF1,DPM,IDP,JDP,RPM,IR,JR,NS,GM102
     1,AOSQ,DXSQ,DXDY,DYSQ,DX2,DY2,KLOSE
      COMMON /BLOK4/ GMSQ,GOGM1,TOGMSQ,CPO,KSTAR
      COMMON /BLOK5/ JM1,DY,I1,JSUP,JSON,GF3,ISUB,ISUP,SUMRP
      COMMON /BLOK6/ XO(100),YO(100),XOP(100),XOPP(100),XOPPP(100),
     * YOP(100),YOPP(100),YOPPP(100),SOO(100),IXY,DYDXN,DYDXT
      COMMON /BLOK7/ SMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,XIO,XI1,CXM,
     * DX,X10,XREF
      COMMON /BLOK8/ ALF
      COMMON /BLOK9/ N
      DIMENSION DESC(8)
      DATA PI/3.14159265358979/,RAD/57.2957795130823/
C
C     ***************************************************************
C
C     --- WARNING --- WARNING --- WARNING --- WARNING --- WARNING ---
C     DON'T FORGET TO CHANGE ID WHEN I-DIMENSION IS CHANGED....
C
C     ***************************************************************
C
      ID=81
C
      CALL SECOND (T1)
      WRITE(6,270) T1
C
C     ***************************************************************
C
C     FOLLOWING 4 INSTRUCTIONS ESTABLISH TIME TO START CLEANUP OPERA-
C     TIONS. WHEN CPU TIME (INCLUDING COMPILE TIME) COMES WITHIN TSAF
C     SECONDS OF THE TIME LIMIT, TL, ITERATION IS STOPPED AND CLEANUP
C     STARTS. JPARAMS IS AN LRC SUBROUTINE THAT RETURNS JOB TIME LIMIT
C     IN D(11). SECOND IS AN LRC SUBROUTINE USED TO MONITOR THE CURRENT
C     TIME.
C
C     ***************************************************************
C
      TSAF=30.
      KTL=1
      CALL JPARAMS(D)
      TL=D(11)
   10 READ(5,290) DESC
      IF(EOF(5)) 20,30
   20 IF (NPLOT.EQ.1) RETURN
      STOP
   30 CONTINUE
      READ(5,300)IXY
      READ(5,330)(XO(I),I=1,IXY)
      READ(5,330)(YO(I),I=1,IXY)
      READ(5,330)DYDXN,DYDXT,YMAX,XREF
      READ(5,300)IMAX,JMAX,MIT,MHALF,KLOSE,NPLOT
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      READ(5,330)RF1,COVERG,QF3
      READ(5,330)DNDYO,ALF,DXIDXO,XM,CXM,DXIDXM
      READ(5,330)GAM,AMINF
      WRITE(6,280)DESC
      WRITE(6,720)(I,XO(I),YO(I),I=1,IXY)
      WRITE(6,420)DYDXN,DYDXT,YMAX,XREF
      TSAF=TSAF+NPLOT*30.
      THETAN=ATAN(DYDXN)
      THETAT=ATAN(DYDXT)
      DYDSN=SIN(THETAN)
      DXDSN=COS(THETAN)
      DYDST=SIN(THETAT)
      DXDST=COS(THETAT)
      IF(DYDXN.LT.900.)GO TO 31
      DYDSN=1.
      DXDSN=0.
   31 CONTINUE
      IF(ABS(DYDXT).LT.900.)GO TO 32
      DYDST=-1.
      DXDST=0.
   32 CONTINUE
      CALL FIT(IXY,XO,YO,SOO,XOP,XOPP,YOP,YOPP,DYDSN,DXDSN,DYDST,DXDST)
      CALL SPLIF(1,IXY,SOO,XO,XOP,XOPP,XOPPP,1,DXDSN,1,DXDST,IND)
      CALL SPLIF(1,IXY,SOO,YO,YOP,YOPP,YOPPP,1,DYDSN,1,DYDST,IND)
      NHALF=0
      ANMAX=1.E+08
      JSKP=1
      JPAGE=31
      N=0
   40 IF (JMAX/JSKP.LE.JPAGE) GO TO 50
      JSKP=JSKP+1
      GO TO 40
   50 CONTINUE
      X1(1)=0.
      X2(1)=X1(1)
      RMSQ=YMAX**2
      GM1=GAM-1.
      GM1O2=.5*GM1
      GOGM1=GAM/GM1
      AMSQ=AMINF**2
      GMSQ=GM1O2*AMSQ
      AOSQ=GM1O2+1./AMSQ
      TOGMSQ=2./(GAM*AMSQ)
      PSTAR=(2.*(1.+GMSQ)/(1.+GAM))**GOGM1
      CPSTAR=TOGMSQ*(PSTAR-1.)
      CPO=TOGMSQ*((1.+GMSQ)**GOGM1-1.)
      KSTAR=4.5+30.*(CPO-CPSTAR)
      IF (KSTAR.GT.100) KSTAR=100
   60 CALL SECOND (T1)
      WRITE(6,320)IMAX,JMAX,MIT,MHALF,KLOSE,NPLOT
     * ,RF1,COVERG,QF3,DNDYO,ALF,DXIDXO,XM,CXM,DXIDXM,GAM,AMINF
C
C     ***********************************************************************
C
C     OVERLAY(1,1) SETS UP THE TANGENTIAL COORDINATES
C
C     ***********************************************************************
C
      CALL OVERLAY(5HJERRY,1,1,6HRECALL)
C
```

```
C
C     ******************************************************************
C
C     OVERLAY(1,2) CALLS NTRANF AND W1W2
C
C     ******************************************************************
C
      CALL OVERLAY(5HJERRY,1,2,6HRECALL)
      DXSQ=1./DX**2
      RCHEK=100.*DXSQ
      DXDY=.5/(DX*DY)
      DYSQ=1./DY**2
      DY2=.5/DY
      DX2=.5/DX
      JM1=JMAX-1
      KPOINT=(IMAX-1)*(JMAX-1)
      POINTS=KPOINT
      WRITE(6,470)
      DO 90 I=1,IMAX
      LS(I)=0
      TD=THET(I)*RAD
      TBD=THETB(I)*RAD
      WRITE(6, 480) I,S(I),XB(I),YB(I),TD,TBD,AK(I),F(I)
   90 CONTINUE
      WRITE(6,451)ALF
      WRITE(6,450)
      WRITE(6, 460) (J,AN(J),G(J),GM(J),J=1,JMAX)
      CALL SECOND (T)
      T=T-T1
      WRITE(6,430)T
      IF (NHALF.GT.0) GO TO 100
      CALL ESTIM (P,ID,IMAX,JMAX)
  100 IT=0
      DO 110 I=1,I1
  110 DPO(I)=ST(I)
      IF (KLOSE.EQ.1) GO TO 130
      I2=I1+1
      I3=IMAX-1
      DO 120 I=I2,I3
  120 DPO(I)=W2(I)*(1.+F(I)*DX2*(P(I+1,JMAX)-P(I-1,JMAX)))
      DPI=3.*P(IMAX,JMAX)-4.*P(IMAX-1,JMAX)+P(IMAX-2,JMAX)
      DPO(IMAX)=W2(IMAX)*(1.+F(IMAX)*DX2*DPI)
  130 DO 140 I=1,IMAX
  140 P(I,JMAX+1)=W3(I)*P(I,JMAX-1)+W4(I)*P(I,JMAX)-W5(I)*DPO(I)
      WRITE(6,490)
      CALL SECOND (TO)
      SUM1=1.E+07
      SUM=0.
      COVR=COVERG/FLOAT(IMAX-1)**2
  150 CALL SECOND (T1)
      JSUP=0
      IF (AMINF.GE.1.) JSUP=1
      JSON=0
      IF (ABS(AMINF-1.).LE.1.E-06) JSON=1
C
C     ******************************************************************
C
C     OVERLAY(1,3) IS THE MIXED FLOW POTENTIAL ITERATION LOOP
C
C     ******************************************************************
C
```

ORIGINAL PAGE IS OF POOR QUALITY

```
      CALL OVERLAY(5HJERRY,1,3,6HRECALL)
      IT=IT+1
      RAVG=SUMRP/POINTS
      CALL SECOND (T)
      TI=T-T1
      WRITE(6,500)IT,DPM,IDP,JDP,RPM,IR,JR,ISUB,ISUP,RAVG,RF1,QF3,NS,TI
C
C     ****************************************************************
C
C     CHECK FOR TIME LIMIT.
C
C     ****************************************************************
C
      IF (TL-T.GT.TSAF) GO TO 160
      WRITE(6,370)T,TL,RPM,COVR
      KTL=2
      GO TO 180
  160 CONTINUE
C
C     ****************************************************************
C
C     CHECK FOR DIVERGENCE.
C
C     ****************************************************************
C
      IF(RPM.LT.RCHEK) GO TO 161
      WRITE(6,610)
      GO TO 10
  161 CONTINUE
C
C     ****************************************************************
C
C     CHECK FOR CONVERGENCE OR ITERATION LIMIT
C
C     ****************************************************************
C
      IF(RPM.GE.COVR)GO TO 171
      WRITE(6,700)RPM,COVR
      GO TO 180
  171 CONTINUE
      IF(IT.LT.MIT)GO TO 172
      WRITE(6,310)MIT,RPM,COVR
      GO TO 180
  172 CONTINUE
C
C     ****************************************************************
C
C     INCREASE PHI-ST DAMPING COEFFICIENT OR DECREASE RF1 IF AVERAGE
C     MAXIMUM CORRECTION OF LAST 10 CYCLES HAS INCREASED OVER PREVIOUS 1
C
C     ****************************************************************
C
      SUM=SUM+DPM
      IF (MOD(IT,10).NE.0) GO TO 150
      IF (SUM1.GT.SUM) GO TO 173
      QF3=QF3+.1*ISUP
      RF1=RF1*(1.-.1*ISUB)
      SUM1=1.E+07
      IF(ISUP.EQ.1)WRITE(6,680)QF3
      IF(ISUB.EQ.1)WRITE(6,710)RF1
```

```
      GO TO 174
  173 SUM1=SUM
  174 SUM=0.
      GO TO 150
  180 CALL SECOND (T)
C
C     ******************************************************************
C
C     THE FOLLOWING STATEMENTS CALL FOR PREPARATION AND PRINTING OF
C     CP, MACH NO., DRAG, ROUGH CP PLOT, MACH NO. CHART OF FLOW FIELD,
C     SONIC LINE CALCULATION, AND WRITING ON DISC FOR CALCOMP PLOTS.
C
C     ******************************************************************
C
      T1=T-T0
      WRITE(6, 510) T1,IT,NHALF
C
C     ******************************************************************
C
C     OVERLAY(1,4) CALLS CPBODY, DRAG, AND CPPLOT.
C
C     ******************************************************************
C
      CALL OVERLAY(5HJERRY,1,4,6HRECALL)
C
C     ******************************************************************
C
C     OVERLAY(1,5) CALLS MCHART AND SONLIN.
C
C     ******************************************************************
C
      CALL OVERLAY(5HJERRY,1,5,6HRECALL)
      CALL SECOND (T1)
      T1=T1-T
      WRITE(6,600)T1
      IF (NPLOT.NE.1) GO TO 220
      WRITE(4) DESC
      WRITE(4) IMAX,JMAX,IT,KLOSE,N
      WRITE(4)CPSTAR,AMINF,DPM,XREF,DXIDXO,DNDYO,QF3
      IF(KLOSE.EQ.1)GO TO 211
      WRITE(4) CXM,XM,XIM,DXIDXM
  211 CONTINUE
      WRITE(4)(AN(J),J=2,JMAX)
      WRITE(4)(ST(I),I=1,IMAX)
      WRITE(4)(CT(I),I=1,IMAX)
      WRITE(4) (XB(I),I=1,IMAX)
      WRITE(4) (YB(I),I=1,IMAX)
      WRITE(4) (CP(I),I=1,IMAX)
      IF (N.EQ.0) GO TO 220
      WRITE(4) (XS(I),I=1,N)
      WRITE(4) (YS(I),I=1,N)
  220 CONTINUE
      IF(KTL.EQ.2)GO TO 20
      IF (NHALF.EQ.MHALF) GO TO 10
      NHALF=NHALF+1
C
C     ******************************************************************
C
C     OVERLAY(1,6) IS THE GRID REFINEMENT ROUTINE.
C
```

```
C     ******************************************************************
C
      CALL OVERLAY(5HJERRY,1,6,6HRECALL)
      MIT=2*MIT
      DX=.5*DX
  250 IF (JMAX/JSKP.LE.JPAGE) GO TO 260
      JSKP=JSKP+1
      GO TO 250
  260 CONTINUE
      GO TO 60
  270 FORMAT (1H1///,16H COMPUTING TIME=,F6.1,8H SECONDS/)
  280 FORMAT (1H1,8A10)
  290 FORMAT (8A10)
  300 FORMAT (16I5)
  310 FORMAT(/* ----DID NOT CONVERGE IN*I4* CYCLES.---- RMAX=*
     * E9.2*, COVR=*E9.2/)
  320 FORMAT(6H1IMAX=I3/6H JMAX=I3/5H MIT=I4/7H MHALF=I1
     * /7H KLOSE=I1/7H NPLOT=I1/5H RF1=F5.3
     * /8H COVERG=E9.2/5H QF3=E9.2/7H DNDYO=E10.3
     * /5H ALF=F4.2/8H DXIDXO=E10.3/4H XM=E10.3
     * /5H CXM=E10.3/8H DXIDXM=E10.3/5H GAM=F4.2
     * /7H AMINF=F6.4)
  330 FORMAT (8E10.3)
  370 FORMAT(/* MUST STOP ITERATIONS, CLOSE TO TIME LIMIT.*/
     * * COMPUTING TIME =*F6.1* TIME LIMIT=*F6.1/
     * * RMAX=*E9.2*, COVR=*E9.2)
  420 FORMAT(/* DYDXN=*F10.4,/* DYDXT=*F10.4,/* YMAX=*F10.4,/* XREF=*
     * ,F10.4)
  430 FORMAT (/,44H CPU SECONDS FOR BODY GEOMETRY COMPUTATIONS=,F6.3/)
  450 FORMAT (10X,1HJ,4X,2HAN,10X,1HG,11X,2HGH/)
  451 FORMAT(/,*1----- NORMAL COORD. S  ETCH FOR ALF=*F6.3* ------*/)
  460 FORMAT (I12,3E12.4)
  470 FORMAT (1H1,9X,1HI,4X,1HS,11X,1HX,11X,1HY,10X,4HTHET,8X,5HTHETB,8
     1X,2HAK,10X,1HF//)
  480 FORMAT (I12,8E12.4)
  490 FORMAT(1H1,2X,2HIT,3X5HDPMAX,5X2HID,2X2HJD,3X4HRMAX,6X2HIR,2X2HJR,
     1 1X4HISUB,1X4HISUP,3X4HRAVG,6X3HRF1,4X3HQF3,6X2HNS,
     2 3X7HSEC/CYC/)
  500 FORMAT(I5,E11.3,2I4,E11.3,2I4,2I5,E11.3,2F7.3,I6,F9.3)
  510 FORMAT(13H0CPU SECONDS=,F7.2,4H FOR,I4,19H ITERATIONS, NHALF=I1/)
  600 FORMAT (47H0CPU SECONDS TO COMPUTE AND PLOT CP AND MCHART=,F7.3/)
  610 FORMAT(///* ------------------DIVERGENCE, RMAX EXCEEDS RCHEK.*
     1 * GO DIRECTLY TO JAIL, DO NOT PASS GO, DO NOT COLLECT $200.----
     2 ------------------*///)
  680 FORMAT(/* QF3 INCREASED TO*F6.3* BECAUSE 10-CYCLE AVERAGE OF*
     1 * RMAX INCREASED.*/)
  700 FORMAT(/* ----CONVERGENCE----, RMAX=*E9.2*, COVR=*E9.2/)
  710 FORMAT(/* RF1 DECREASED TO*F6.3* BECAUSE 10-CYCLE AVG FOR*
     1 * RMAX INCREASED.*/)
  720 FORMAT(/* INPUT COORDINATES*/4X1HI4X1HX9X1HY/(I5,2F10.6))
      END
      SUBROUTINE FIT(N,X,Y,S,X1,X2,Y1,Y2,DY1,DX1,DY2,DX2)
      DIMENSION X(1),Y(1),S(1),X1(1),X2(1),Y1(1),Y2(1)
      RES=1.0E-07
      TOL=.0625*RES
      K=0
      KMAX=500
      S(1)      = 0.
      M         = N  -1
      DO 22 I=1,M
```

```
      VAL        = X(I+1)  -X(I)
      DUM        = Y(I+1)  -Y(I)
   22 S(I+1)     = S(I)  +SQRT(VAL**2  +DUM**2)
   31 CALL  SPLIF(1,M,S,X,X1,X2,X2,1,DX1,1,DX2,IND)
      CALL  SPLIF(1,N,S,Y,Y1,Y2,Y2,1,DY1,1,DY2,IND)
      ERR        = 0.
      DUM        = 0.
      DO 32 I=1,M
      S0         = S(I+1)  -DUM
      DUM        = S(I+1)
      S1         = S(I+1)  -S(I)
      X3         = (X2(I+1)  -X2(I))/S1
      Y3         = (Y2(I+1)  -Y2(I))/S1
      CALL  ARCL (S1,S0,X1(I),X2(I),X3,Y1(I),Y2(I),Y3,R,IND,TOL)
      VAL        = ABS(S1  -S0)
      IF (VAL  -ERR) 32,32,33
   33 ERR        = VAL
   32 S(I+1)     = S(I)  +S1
      K=K+1
      IF(K.LE.KMAX)GO TO 34
      WRITE(6,9901)
      RETURN
   34 CONTINUE
      IF (ERR  -RES) 41,41,31
   41 RETURN
 9901 FORMAT(* FIT FAILED TO CONVERGE*)
      END
      SUBROUTINE  SPLIF(M,N,S,F,FP,FPP,FPPP,KM,VM,KN,VN,IND)
C     SPLINE FIT - JAMESON
      DIMENSION   S(1),F(1),FP(1),FPP(1),FPPP(1)
      IND        = 0
      K          = IABS(N  -M)
      IF (K  -1) 81,81,1
    1 K          = (N  -M)/K
      I          = M
      J          = M  +K
      DS         = S(J)  -S(I)
      D          = DS
      IF (DS) 11,81,11
   11 DF         = (F(J)  -F(I))/DS
      IF (KM  -2) 12,13,14
   12 U          = .5
      V          = 3.*(DF  -VM)/DS
      GO TO 25
   13 U          = 0.
      V          = VM
      GO TO 25
   14 U = -1.
      V          = -DS*VM
      GO TO 25
   21 I          = J
      J          = J  +K
      DS         = S(J)  -S(I)
      IF (D*DS) 81,81,23
   23 DF         = (F(J)  -F(I))/DS
      B          = 1./(DS  +DS  +U)
      U          = B*DS
      V          = B*(6.*DF  -V)
   25 FP(I)      = U
      FPP(I)     = V
```

```
      U          = (2.  -U)*DS
      V          = 6.*DF  +DS*V
      IF (J  -N) 21,31,21
   31 IF (KN  -2) 32,33,34
   32 V          = (6.*VN  -V)/U
      GO TO 35
   33 V          = VN
      GO TO 35
   34 V          = (DS*VN  +FPP(I))/(1.  +FP(I))
   35 B          = V
      D          = DS
   41 DS         = S(J)  -S(I)
      U          = FPP(I)  -FP(I)*V
      FPPP(I)    = (V  -U)/DS
      FPP(I)     = U
      FP(I)      = (F(J)  -F(I))/DS  -DS*(V  +U  +U)/6.
      V          = U
      J          = I
      I          = I  -K
      IF (J  -M) 41,51,41
   51 FPPP(N)    = FPPP(N-1)
      FPP(N)     = B
      FP(N)      = DF  +D*(FPP(N-1)  +B  +B)/6.
      I ^        = 1
   81 RETURN
      END
      SUBROUTINE  ARCL (S,STEP,X1,X2,X3,Y1,Y2,Y3,M,N,TOL)
C     CALCULATES ARC LENGTH USING FIRST THREE DERIVIATIVES OF X AND Y
      DP         = STEP
      P          = .5*DP
      N          = 1
      S          = SQRT(X1**2  +Y1**2)
      XX         = X1  +STEP*(X2  +.5*STEP*X3)
      YY         = Y1  +STEP*(Y2  +.5*STEP*Y3)
      S          = S  +SQRT(XX**2  +YY**2)
      XX         = X1  +P*(X2  +.5*P*X3)
      YY         = Y1  +P*(Y2  +.5*P*Y3)
      SUM        = SQRT(XX**2  +YY**2)
      SUM        = SUM*DP*2./3.
      S          = SUM  +S*DP/6.
      DO 12 I=2,M
      S1         = S
      S          = .5*(S  -.5*SUM)
      DP         = .5*DP
      P          = .5*DP
      XX         = X1  +P*(X2  +.5*P*X3)
      YY         = Y1  +P*(Y2  +.5*P*Y3)
      SUM        = SQRT(XX**2  +YY**2)
      N          = 2*N
      L          = N  -1
      DO 14 K=1,L
      P          = P  +DP
      XX         = X1  +P*(X2  +.5*P*X3)
      YY         = Y1  +P*(Y2  +.5*P*Y3)
   14 SUM        = SUM  +SQRT(XX**2  +YY**2)
      SUM        = SUM*DP*2./3.
      S          = S  +SUM
      ERR        = S/S1  -1.
      IF (ABS(ERR)  -TOL) 21,21,12
   12 CONTINUE
```

```
   21 RETURN
      END
      SUBROUTINE ESTIM (P,ID,IMAX,JMAX)
C
C     ------GIVES INITIAL ESTIMATE OF POTENTIAL AS ZERO PERTURBATION----
C
      DIMENSION P(ID,1)
      DO 40 I=1,IMAX
      DO 40 J=1,JMAX
   40 P(I,J)=0.
      RETURN
      END
      OVERLAY(JERRY,1,1)
      PROGRAM ONE1
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPC(81),F(81),AK(81),S(81)
      COMMON /BLOK1/ XST
      COMMON /BLOK2/ PI,RAD
      COMMON /BLOK3/ IMAX,DUMMY(17),KLOSE
      COMMON /BLOK5/ JM1,DY,I1
      COMMON /BLOK6/ XO(100),YO(100),XOP(100),XOPP(100),XOPPP(100),
     * YOP(100),YOPP(100),YOPPP(100),SOO(100),IXY,DYDXN,DYDXT
      COMMON /BLOK7/ SMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,XIO,XI1,CXM,
     * DX,X10,XREF
      DIMENSION XB1(100),YB1(100),XB2(100),YB2(100)
      DIMENSION D1(101),D2(101),D3(101),D4(101),D5(101)
      IF(KLOSE.EQ.0)GO TO 100
      I1=IMAX
      SMAX=SOO(IXY)
      A=(3.*SMAX-DXIDXO)/2.
      B=4.*(SMAX-A)
      DX=1./(IMAX-1)
      XX=0.
      DO 1 I=1,IMAX
      S(I)=.5*SMAX+(XX-.5)*(A+B*(XX-.5)**2)
      DXIDX=A+3.*B*(XX-.5)**2
      F(I)=1./DXIDX
      XX=XX+DX
    1 CONTINUE
      CALL INTPL(1,IMAX,S,XB,XB1,XB2,1,IXY,SOO,XO,XOP,XOPP,XOPPP)
      CALL INTPL(1,IMAX,S,YB,YB1,YB2,1,IXY,SOO,YO,YOP,YOPP,YOPPP)
      DO 4 I=1,IMAX
      AK(I)=SQRT(XB2(I)**2+YB2(I)**2)
      IF(XB1(I).LE.1.)GO TO 2
      WRITE(6,9901)XB1(I),I
      XB1(I)=1.
    2 CONTINUE
      IF(ABS(YB1(I)).LE.1.)GO TO 3
      WRITE(6,9902)YB1(I),I
      YB1(I)=SIGN(1.,YB1(I))
    3 CONTINUE
      THETX=SIGN(ACOS(XB1(I)),YB1(I))
      THETY=ASIN(YB1(I))
      THET(I)=.5*(THETX+THETY)
    4 CONTINUE
      THET(1)=.5*PI
      THET(IMAX)=-.5*PI
      RETURN
```

```
100 CONTINUE
    CALL HORTAN(IXY,XO,YO,SOO,XOP,XOPP,XOPPP,YOP,YOPP,YOPPP,SHT,XHT
   * ,YHT,IOBHT)
    XST=XO(IXY)
    S1=SHT
    X10=XHT
    CALL SETUPO(IMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,CXM,DX,X10)
    DX=1./(IMAX-1)
    XX=0.
    DO 101 I=1,IMAX
    CALL SDRIVO(XX,SS,DXIDX,XIM,A4,DXIDXO,DXIDXM,A2,A3,CXM)
    S(I)=SS
    F(I)=1./DXIDX
    IF(S(I).LE.S1)I1=I
    XX=XX+DX
101 CONTINUE
    CALL INTPL(1,I1,S,XB,XB1,XB2,1,IXY,SOO,XO,XOP,XOPP,XOPPP)
    CALL INTPL(1,I1,S,YB,YB1,YB2,1,IXY,SOO,YO,YOP,YOPP,YOPPP)
    DO 104 I=1,I1
    AK(I)=SQRT(XB2(I)**2+YB2(I)**2)
    IF(XB1(I).LE.1.)GO TO 102
    WRITE(6,9901)XB1(I),I
    XB1(I)=1.
102 CONTINUE
    IF(ABS(YB1(I)).LE.1.)GO TO 103
    WRITE(6,9902)YB1(I),I
    YB1(I)=SIGN(1.,YB1(I))
103 CONTINUE
    THETX=SIGN(ACOS(XB1(I)),YB1(I))
    THETY=ASIN(YB1(I))
    THET(I)=.5*(THETX+THETY)
104 CONTINUE
    THET(1)=.5*PI
    IF(IOBHT.LT.IXY)GO TO 105
    I2=I1
    GO TO 111
105 CONTINUE
    IOT=IOBHT+1
    D1(IOT)=XHT
    D2(IOT)=YHT
    DO 107 I=IOT,IXY
    D1(I+1)=XO(I)
    D2(I+1)=YO(I)
107 CONTINUE
    IXYP1=IXY+1
    CALL SPLIF(IOT,IXYP1,D1,D2,D3,D4,D5,1,0.,1,DYDX1,IND)
    IMAXM1=IMAX-1
    I1P1=I1+1
    DO 108 I=I1P1,IMAXM1
    DS=S(I)-SHT
    XB(I)=XHT+DS
    IF(XB(I).GT.XO(IXY))GO TO 108
    I2=I
108 CONTINUE
    CALL INTPL(I1P1,I2,XB,YB,YB1,YB2,IOT,IXYP1,D1,D2,D3,D4,D5)
    DO 110 I=I1P1,I2
    AK(I)=-YB2(I)
    THETB(I)=ATAN(YB1(I))
    THET(I)=0.
110 CONTINUE
```

```
  111 CONTINUE
      I2P1=I2+1
      DO 112 I=I2P1,IMAX
      DS=S(I)-SHT
      XB(I)=XHT+DS
      YB(I)=YO(IXY)
      AK(I)=0.
      THET(I)=0.
      THETB(I)=0.
  112 CONTINUE
      RETURN
 9901 FORMAT(* XB1=*E16.8* AT I=*I3)
 9902 FORMAT(* YB1(I)=*E16.8* AT I=*I3)
      END
      SUBROUTINE  INTPL(MI,NI,SI,FI,FIP,FIPP,M,N,S,F,FP,FPP,FPPP)
C     INTERPOLATION USING TAYLOR SERIES - JAMESON
      DIMENSION SI(1),FI(1),FIP(1),FIPP(1),S(1),F(1),FP(1),FPP(1)
     - ,FPPP(1)
      K          = IABS(N  -M)
      K          = (N  -M)/K
      I          = M
      MIN        = MI
      NIN        = NI
      D          = S(N)  -S(M)
      IF (D*(SI(NI)  -SI(MI))) 11,13,13
   11 MIN        = NI
      NIN        = MI
   13 KI         = IABS(NIN  -MIN)
      IF (KI) 21,21,15
   15 KI         = (NIN  -MIN)/KI
   21 II         = MIN  -KI
   31 II         = II  +KI
      SS         = SI(II)
   33 I          = I  +K
      IF (I  -N) 35,37,35
   35 IF (D*(S(I)  -SS)) 33,33,37
   37 CONTINUE
      I          = I  -K
      SS         = SS  -S(I)
      FIPP(II)=FPP(I)+SS*FPPP(I)
      FIP(II)=FP(I)+SS*(FPP(I)+SS*FPPP(I)*.5)
      FI(II)=F(I)+SS*(FP(I)+.5*SS*(FPP(I)+SS*FPPP(I)/3.))
      IF (II  -NIN) 31,41,31
   41 RETURN
      END
      SUBROUTINE SETUPO(IMAX,S1,XM,XIM,A4,A1,BB,A2,A3,CXM,DX,X10)
      XIM=S1+XM-X10
      DX=1./(IMAX-1)
      C1=XIM/CXM-A1
      C2=BB-A1
      C3=2.*CXM*BB/(1.-CXM)
      X2=CXM**2
      X4=X2**2
      X6=X4*X2
      A2=(70.*C1-22.*C2+2.*C3)/16./X2
      A3=(-84.*C1+36.*C2-4.*C3)/16./X4
      A4=(30.*C1-14.*C2+2.*C3)/16./X6
      RETURN
      END
      SUBROUTINE SDRIVO (XX,S,DXIDX,XIM,A4,A1,BB,A2,A3,CXM)
```

```
C
C     ******CALCULATES S AND DXIDX AS FUNCTIONS OF X
C
      IF (XX.GT.CXM) GO TO 10
      X2=XX**2
      X4=X2**2
      X6=X4*X2
      T1=A2*X2
      T2=A3*X4
      T3=A4*X6
      S=XX*(A1+T1+T2+T3)
      DXIDX=A1+3.*T1+5.*T2+7.*T3
      RETURN
   10 IF(ABS(XX-1.).LE.1.E-06) GO TO 20
      T=XX-CXM
      T1=1./(1.-T/(1.-CXM))
      S=XIM+BB*T*T1
      DXIDX=BB*T1**2
      RETURN
   20 S=1.E+30
      DXIDX=1.E+30
      RETURN
      END
      SUBROUTINE HORTAN(N,X,Y,S,XP,XPP,XPPP,YP,YPP,YPPP,SHT,XHT,YHT,IH)
      DIMENSION X(1),Y(1),S(1),XP(1),XPP(1),XPPP(1),YP(1),YPP(1),YPPP(1)
      DIMENSION SH(1),XH(1),YH(1),DUM(1)
      DO 1 I=1,N
      IF(YP(I).LT.0.)GO TO 2
      IH=I
    1 CONTINUE
    2 CONTINUE
      SHT=S(IH)+YP(IH)/(YP(IH)-YP(IH+1))*(S(IH+1)-S(IH))
      SH(1)=SHT
      CALL INTPL(1,1,SH,XH,DUM,DUM,1,N,S,X,XP,XPP,XPPP)
      CALL INTPL(1,1,SH,YH,DUM,DUM,1,N,S,Y,YP,YPP,YPPP)
      XHT=XH(1)
      YHT=YH(1)
      RETURN
      END
      OVERLAY(JERRY,1,2)
      PROGRAM ONE2
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPO(81),F(81),AK(81),S(81),LS(81),FM(81)
      COMMON AN(81),G(81),GH(81),CB(81),D(81),X1(81),X2(81),M(81),MR(81)
     1,MRP(81),MRM(81),MRMM(81)
      COMMON XS(400),YS(400)
      COMMON ID,ANMAX,DNDYO,YMAX,CD,RMSQ,JSKP
      COMMON /BLOK2/PI,RAD/BLOK3/IMAX,JMAX,C2,RF1,DPM,IDP,J
     1DP,RPM,IR,JR,NS,GM102,AOSQ,DXSQ,DXDY,DYSQ,DX2,DY2,KLOSE
      COMMON /BLOK5/ JM1,DY,I1,JSUP,JSON,QF3,ISUB,ISUP,SUMRP
      COMMON /BLOK7/ SMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,XIO,XI1,CXM,
     * DX
      COMMON/BLOK8/ALF
      CALL NTRANF (AN,ANMAX,JMAX,DNDYO,DY,G,GH,ALF)
      CALL W1W2(THET,THETB,YB,YBP,W1,W2,W3,W4,W5,ST,CT,G(JMAX),DNDYO
     * ,DY,I1,IMAX,KLOSE,ALF)
      RETURN
      END
```

```
      SUBROUTINE NTRANF (X,XMAX,JMAX,DNDYO,DY,G,GH,ALF)
C
C     ------COMPUTES STRETCHING OF NORMAL COORDINATE------
C
      DIMENSION X(1), G(1), GH(1)
      B=1./DNDYO
      IF (XMAX.GE.1.E+06) GO TO 10
      A=B-1./XMAX
      K=0
      GO TO 20
   10 K=1
      A=B
   20 DY=1./(JMAX-1)
      DO 50 J=1,JMAX
      ZETA=1.-(J-1)*DY
      IF (J*K.EQ.1) GO TO 30
      AA=(1.-ZETA)**ALF
      X(J)=ZETA/B/AA
      GO TO 40
   30 G(1)=0.
      GO TO 50
   40 G(J)=B*AA*(1.-ZETA)/(1.-(1.-ALF)*ZETA)
      IF (J.EQ.1) GO TO 50
      GH(J-1)=.5*(G(J)+G(J-1))
   50 CONTINUE
      AA=(1.+DY)**ALF
      GJP1=B*AA*(1.+DY)/(1.+(1.-ALF)*DY)
      GH(JMAX)=.5*(GJP1+G(JMAX))
      RETURN
      END
      SUBROUTINE W1W2(THET,THETB,YB,YBP,W1,W2,W3,W4,W5,ST,CT,G,DNDYO
     * ,DY,I1,IMAX,KLOSE,ALF)
C
C     ------CALCULATES YBP(I),W1(I),W2(I),ST(I),CT(I)------
C
      DIMENSION THET(1), THETB(1), YB(1), YBP(1), W1(1), W2(1), W3(1), W
     14(1), W5(1), ST(1), CT(1)
      DO 10 I=1,IMAX
      ST(I)=SIN(THET(I))
   10 CT(I)=COS(THET(I))
      DO 20 I=1,I1
      YBP(I)=0.
      W1(I)=1.
      W2(I)=0.
      W3(I)=1.
      W4(I)=0.
      W5(I)=2.*DY/G
   20 THETB(I)=THET(I)
      IF (KLOSE.EQ.1) RETURN
      I1P1=I1+1
      DO 30 I=I1P1,IMAX
      YBP(I)=TAN(THETB(I))
      W1(I)=1.+YBP(I)**2
      W2(I)=YBP(I)/W1(I)
      Y1=-DY
      IF(YB(I).GE.DNDYO)GO TO 25
      YBOA=YB(I)/DNDYO
      YA=-YBOA-ALF*YBOA**2-ALF*(3.*ALF-1.)/2.*YBOA**3
     * -ALF*(16.*ALF**2-12.*ALF+2.)/6.*YBOA**4
      IF(ABS(YA).GT.DY)GO TO 25
```

```
      Y1=DY*2.*YA
   25 CONTINUE
      Y1ODY=Y1/DY
      W3(I)=Y1ODY**2
      W4(I)=1.-W3(I)
      W5(I)=-Y1*(1.-Y1ODY)/G
   30 CONTINUE
      RETURN
      END
      OVERLAY(JERRY,1,3)
      PROGRAM ONE3
C
C     ------SOLUTION OF POTENTIAL EQN. BY COLUMN RELAXATION------
C
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPO(81),F(81),AK(81),S(81),LS(81),FM(81)
      COMMON AN(81),G(81),GH(81),CB(81),D(81),X1(81),X2(81),M(81),HR(81)
     1,HRP(81),HRM(81),HRMM(81)
      COMMON /BLOK3/ IMAX,JMAX,C2,RF1,DPM,IDP,JDP,RPM,IR,JR,NS,GM102
     1,AOSQ,DXSQ,DXDY,DYSQ,DX2,DY2,KLOSE
      COMMON /BLOK5/ JM1,DY,I1,JSUP,JSON,QF3,ISUB,ISUP,SUMRP
      SUMRP=0.
      QF1=1./RF1
      JO=2+JSUP
      IF (JO.EQ.2) JO=2+JSON
      JP1=JMAX+1
      J=JMAX
      DO 10 I=1,I1
   10 DPO(I)=ST(I)
      IF (KLOSE.EQ.1) GO TO 30
      I1P1=I1+1
      IMAXM1=IMAX-1
      DO 20 I=I1P1,IMAXM1
   20 DPO(I)=W2(I)*(1.+F(I)*DX2*(P(I+1,J)-P(I-1,J)))
      DPI=3.*P(IMAX,JMAX)-4.*P(IMAX-1,JMAX)+P(IMAX-2,JMAX)
      DPO(IMAX)=W2(IMAX)*(1.+F(IMAX)*DX2*DPI)
   30 CONTINUE
C
C     ------START A CYCLE AT I=1(STAG. PT.)------
C
      DPM=0.
      RPM=DPM
      I=1
      J=JO
      NS=0
      KS=0
      I1M1=I1-1
      A6=0.
      B1=0.
      B4=0.
      GO TO 230
   40 IMM=I-2
      IF (I.EQ.2) IMM=2
C
C     ------COMPUTE QUANTITIES DEPENDING ON I ALONE------
C
      FD=F(I)*DX2
      FD1=F(I)*DXSQ
```

```
      FD2=F(I)*DXDY
      KS=0
      CC=0.
      DD=CC
C
C     ------COMPUTE QUANTITIES DEPENDING ON I AND J ------
C
   50 CONTINUE
      HRMM(J)=HRM(J)
      HRM(J)=HR(J)
      HR(J)=HRP(J)
      IF (I.GT.I1M1) GO TO 60
      HRP(J)=1./(1.+AK(I+1)*AN(J))
      GO TO 70
   60 HRP(J)=1.
      A5=0.
   70 CONTINUE
      S1=FD1*HR(J)
      S2=FD2*G(J)*HR(J)
      S3=G(J)*DYSQ
      S4=G(J)*DY2
      S5=FD*HR(J)
      HF=F(I)*HR(J)
      TIM=F(I-1)*HRM(J)
      FHM=.5*(HF+TIM)
      AKH=AK(I)*HR(J)
      RR=1./(YB(I)+AN(J)*CT(I))
C
C     ------COMPUTE PHI-DIFFERENCES FOR VELOCITY COMPONENTS,NOTE
C     INCREASING J MEANS DECREASING NORMAL(ZETA OR N) COORDINATE.----
C
      DPI=P(I+1,J)-X1(J)
      DPJ=P(I,J-1)-P(I,J+1)
      DPJJ=GH(J-1)*(P(I,J-1)-P(I,J))-GH(J)*(P(I,J)-P(I,J+1))
      U=CT(I)+DPI*S5-YBP(I)*DPJ*S4
      V=-ST(I)+DPJ*S4
      VB=V-YBP(I)*U
      L=1
      IF (V.LT.0.) L=-1
      IF (J.EQ.JMAX) L=-1
      T=L
      UU=U*U
      VV=V*V
      QQ=UU+VV
      AA=A0SQ-GM1O2*QQ
      AR=1./AA
      T4=1.-UU*AR
      UV=U*V
      UVAR=UV*AR
      A4=(AKH*T4+RR*CT(I))*S4
      IF (I.GT.I1) GO TO 80
      A5=(2.*AKH*UVAR+RR*ST(I))*S5
   80 CONTINUE
      FH=.5*(HF+F(I+1)*HRP(J))
      DPIJ=P(I+1,J-1)-P(I+1,J+1)+P(I-1,J+1)-P(I-1,J-1)
      DPII=FH*(P(I+1,J)-P(I,J))-FHM*(P(I,J)-P(I-1,J))
      B1=0.
      D4=0.
      A6=0.
      IF (QQ.LT.AA) GO TO 120
```

```
C
C     ------BACKWARD SECOND DIFFS FOR SUPERSONIC FLOW------
C
      KS=KS+1
      QR=1./QQ
      S1=S1*QR
      S2=S2*QR
      S3=S3*QR
      AUU=UU*S1
      AVV=VV*S1
      UB=U+YBP(I)*V
      BUVN=S2*V*UB
      CUU=UB*UB*S3
      CVV=VB*VB*S3
      IF (QF3.LE.1.E-06) GO TO 91
      FAC=SQRT(ABS(1.-QQ/AA))
      B1=-QF3*FAC*ABS(VB)*G(J)*DXDY
      B4=QF3*FAC*U*FD1
      A6=B4*(P(I-1,J)-X1(J))
   91 CONTINUE
      DPNN=AVV*DPII-BUVN*DPIJ+CUU*DPJJ
      KM=(J+J-1+L)/2
      JM=J-L
      IF (JM.GT.1) GO TO 100
      JMM=1
      KMM=1
      GO TO 110
  100 KMM=KM+L
      JMM=JM+L
  110 CONTINUE
      FHMM=.5*(TIM+F(IMM)*HRMM(J))
      DPII=FHM*(P(I,J)-P(I-1,J))-FHMM*(P(I-1,J)-X2(J))
      DPIJ=P(I,JM)-P(I,J)+P(I-1,J)-X1(JM)
      BUVS=-4.*S2*T+U*VB
      A2S=GH(KMM)+GH(KM)
      DPJJ=GH(KMM)*P(I,JMM)-A2S*P(I,JM)+GH(KM)*P(I,J)
      DPSS=AUU*DPII-BUVS*DPIJ+CVV*DPJJ
      A1S=1.-QQ*AR
      B2=.5*A1S*(A2S*CVV-BUVS)
      B3=B1+B2
      A=(1.-T)*B3-CUU*GH(J-1)-A4
      C=(1.+T)*B3-CUU*GH(J)+A4
      B=-A-C+A1S*(BUVS-2.*AUU*FHM)+AVV*FHM+B4
      RP=A1S*DPSS+DPNN+A4*DPJ+A5*DPI+A6
      ARP=ABS(RP-A6)
      SUMRP=SUMRP+ARP
      IF (ARP.LE.RPM) GO TO 140
      ISUP=1
      ISUB=0
      IR=I
      JR=J
      RPM=ARP
      GO TO 140
  120 A1=T4*S1
      A2=(T4*YBP(I)+UVAR)*S2
      A3=(W1(I)-AR*VB**2)*S3
      RP=A1*DPII-A2*DPIJ+A3*DPJJ+A4*DPJ+A5*DPI
      ARP=ABS(RP)
      SUMRP=SUMRP+ARP
      IF (ARP.LE.RPM) GO TO 130
```

```
      ISUB=1
      ISUP=0
      IR=I
      JR=J
      RPM=ARP
C
C     ------COMPUTE TRIDIAGONAL COEFFS------
C
  130 A=-A3*GH(J-1)-A4
      C=-A3*GH(J)+A4
      B=-A-C+QF1*A1*(FH+FHM)
  140 CONTINUE
      B=1./(B-A*CC)
      CC=B*C
      DD=B*(RP-A*DD)
      IF (J.EQ.JMAX) GO TO 150
      CB(J)=CC
      D(J)=DD
      J=J+1
      GO TO 50
  150 DP=DD
      IF (ABS(DP).LE.DPM) GO TO 160
      DPM=ABS(DP)
      IDP=I
      JDP=J
  160 X2(J)=X1(J)
      X1(J)=P(I,J)
      P(I,J)=P(I,J)+DP
      DO 190 JJ=J0,JM1
      J=J-1
      DP=D(J)-CB(J)*DP
      IF (ABS(DP).LE.DPM) GO TO 180
      DPM=ABS(DP)
      IDP=I
      JDP=J
  180 X2(J)=X1(J)
      X1(J)=P(I,J)
  190 P(I,J)=P(I,J)+DP
      J=J0
      LS(I)=KS
      NS=NS+KS
      P(I,JMAX+1)=W3(I)*P(I,JMAX-1)+W4(I)*P(I,JMAX)-W5(I)*DPO(I)
C
C     ------CHECK I FOR END OF CYCLE,IF BODY IS CLOSED,I=IMAX IS SYMMETR
C     AXIS, IF BODY IS OPEN, I=IMAX IS EITHER NOT COMPUTED(SUBSONIC FREE
C     STREAM,P(IMAX,J)=0.) OR EXTRAPOLATED(SUPERSONIC FREE STREAM)-----
C
      IF (I.EQ.IMAX) RETURN
      I=I+1
      IF (I.EQ.IMAX) GO TO 200
      GO TO 40
  200 IF (KLOSF.EQ.1) GO TO 230
      IF (JSON+JSUP.LT.1) GO TO 220
      DO 210 J=J0,JP1
  210 P(I,J)=3.*(P(I-1,J)-P(I-2,J))+P(I-3,J)
  220 RETURN
C
C     ------SPECIAL EQNS FOR SYMMETRY AXIS,I=1 OR IMAX------
C
  230 CC=0.
```

```
      DD=CC
      S1=2.*DXSQ*F(I)**2
      S3=AK(I)*DY2
      IF (I.EQ.1) GO TO 250
      DO 240 JJ=2,JMAX
  240 HR(JJ)=HRP(JJ)
      IM=I-1
      L=-1
      TA=0.
      TC=1.
      IF (I.EQ.IMAX) GO TO 270
  250 IM=2
      L=1
      TA=1.
      TC=0.
      DO 260 JJ=2,JMAX
      HR(JJ)=1./(1.+AK(1)*AN(JJ))
      HRP(JJ)=1./(1.+AK(2)*AN(JJ))
      HRM(JJ)=HRP(JJ)
  260 X1(JJ)=P(2,JJ)
  270 DPII=P(IM,J)-P(I,J)
      DPJ=P(I,J-1)-P(I,J+1)
      V=-ST(I)+DY2*G(J)*DPJ
      VV=V*V
      AA=AOSQ-GM1O2*VV
C
C     ------COMPUTE COEFFS OF DIFF EQ. AT SYMMETRY AXIS------
C
      A1=2.*HR(J)
      A3=S3*A1*G(J)
      A1=A1*S1*HR(J)
      A2=(1.-VV/AA)*G(J)*DYSQ
      B1=0.
      KSUP=0
      KSUB=0
      IF (J.EQ.JM1.AND.I.EQ.IMAX) GO TO 290
      IF (VV.GE.AA) GO TO 300
  290 DPJJ=GH(J-1)*P(I,J-1)-(GH(J-1)+GH(J))*P(I,J)+GH(J)*P(I,J+1)
      GO TO 310
  300 CONTINUE
      IF (QF3.LE.1.E-06) GO TO 301
      FAC=SQRT(ABS(1-VV/AA))
      B1=-ABS(V)*FAC*G(J)*2.*DXDY*QF3
  301 CONTINUE
      KSUP=1
      KS=KS+1
      KM=(J+J-1-L)/2
      KMM=KM-L
      JM=J-L
      JMM=JM-L
      A2S=GH(KMM)+GH(KM)
      DPJJ=GH(KMM)*P(I,JMM)-A2S*P(I,JM)+GH(KM)*P(I,J)
      B=A2S*A2+B1
      A=TA*B-A3
      C=TC*B+A3
      B=-B+A1
      GO TO 320
  310 A=-A2*GH(J-1)-A3
      C=-A2*GH(J)+A3
      B=-A-C+QF1*A1
```

```
      KSUB=1
  320 RP=A1*DPII+A2*DPJJ+A3*DPJ
      ARP=ABS(RP)
      SUMRP=SUMRP+ARP
      IF (ARP.LT.RPM) GO TO 330
      RPM=ABS(RP)
      ISUB=KSUB
      ISUP=KSUP
      IR=I
      JR=J
  330 CONTINUE
      B=1./(B-A*CC)
      CC=B*C
      DD=B*(RP-A*DD)
      IF (J.EQ.JMAX) GO TO 150
      CB(J)=CC
      D(J)=DD
      J=J+1
      GO TO 270
      END
      OVERLAY(JERRY,1,4)
      PROGRAM ONE4
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPO(81),F(81),AK(81),S(81),LS(81),FM(81)
      COMMON AN(81),G(81),GH(81),CB(81),D(81),X1(81),X2(81),M(81),HR(81)
     1,HRP(81),HRM(81),HRMM(81)
      COMMON XS(400),YS(400)
      COMMON ID,ANMAX,DNDYO,YMAX,CD,RMSQ,JSKP,TSP
      COMMON /BLOK3/IMAX,JMAX,C2,KF1,DPM,IDP,J
     !DP,RPM,IR,JR,NS,GM102,AOSQ,DXSG,DXDY,DYSQ,DX2,DY2,KLOSE
      COMMON /BLOK4/ GMSQ,GOGM1,TOGMSQ,CPO,KSTAR
      COMMON /BLOK7/ SMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,XIO,XI1,CXM,
     * DX,X10,XREF
      CALL CPBODY (P,F,W1,YBP,DPO,CT,LS,CP,FM,ID,IMAX,JMAX,GM102
     * ,AOSQ,DX2,KLOSE,GMSQ,GOGM1,TOGMSQ)
      WRITE(6,570)
      WRITE(6, 580) (I,S(I),XB(I),YB(I),CP(I),FM(I),I=1,IMAX)
      CALL DRAG(CP,YB,THET,THETB,F,RMSQ,IMAX,DX)
      CALL CPPLOT (S,XB,YB,CP,IMAX,CPO,KSTAR)
      RETURN
  570 FORMAT (1H1,8X,1HI,6X,2HSB,8X,2HXB,8X,2HYB,8X,2HCP,8X,1HM/)
  580 FORMAT (I10,3F10.3,2F10.5)
      END
      SUBROUTINE CPBODY (P,F,W1,YBP,DPO,CT,LS,CP,FM,ID,IMAX,JMAX,GM102
     * ,AOSQ,DX2,KLOSE,GMSQ,GOGM1,TOGMSQ)
C
C     ------COMPUTES SURFACE PRESSURE COEFFICIENT AND MACH NO.------
C
      DIMENSION P(ID,1), F(1), W1(1), YBP(1), DPO(1), CT(1), LS(1), CP(1
     1), FM(1)
C
      J=JMAX
      Q=0.
      DO 60 I=1,IMAX
      IP=I+1
      IPP=I+2
      IF (I.EQ.IMAX-1) IPP=I
      IM=I-1
```

```
      IMM=I-2
      IF(I.EQ.1)GO TO 40
      IF (I.EQ.IMAX) GO TO 50
      IF (I.EQ.2) IMM=2
      DJ=P(IP,J)-P(IM,J)
      GO TO 30
   20 CONTINUE
      DJ=3.*P(I,J)-4.*P(IM,J)+P(IMM,J)
C     ------COMPUTE SURFACE VELOCITY------
   30 U=CT(I)+DJ*F(I)*DX2-YBP(I)*DPO(I)
      Q=SQRT(W1(I))*U
   40 QQ=Q*Q
      AA=AOSQ-GM1O2*QQ
C
C     ------SURFACE MACH NO.------
      FM(I)=SQRT(QQ/AA)
C
C     ------PRESSURE RATIO------
      POPINF=(1.+GMSQ*(1.-QQ))**GOGM1
C
C     ------PRESSURE COEFF.------
      CP(I)=TOGMSQ*(POPINF-1.)
      GO TO 60
C
C     ------IF I=IMAX IS NOT A SYMMETRY AXIS,USE BCKWD DIFF FOR DPSB----
   50 IF (KLOSE.EQ.0) GO TO 20
C
C     ------I=IMAX IS A SYMMETRY AXIS------
      Q=0.
      GO TO 40
   60 CONTINUE
      RETURN
      END
      SUBROUTINE DRAG(CP,R,THET,THETB,F,RMSQ,IMAX,DX)
C
C     ------COMPUTES DRAG COEFFICIENT BY INTEGRATION OF SURFACE PRESSURE
C
      DIMENSION CP(1),R(1),THET(1),THETB(1),F(1)
C
C
C     ------TRAPEZOIDAL INTEGRATION------
C
      SUM=0.
      DO 10 I=2,IMAX
   10 SUM=SUM+(CP(I)*R(I)+CP(I-1)*R(I-1))*(R(I)-R(I-1))
      CDTRAP=SUM/RMSQ
      WRITE(6,540)CDTRAP
      IF (MOD(IMAX,2).NE.0) GO TO 15
      WRITE(6,9901)
      RETURN
   15 CONTINUE
C
C     ------SIMPSON INTEGRATION (ONLY IF IMAX ODD) ------
C
      SUM=0.
      IMAXM1=IMAX-1
      DO 20 I=2,IMAXM1,2
   20 SUM=SUM+CP(I)*R(I)*SIN(THETB(I))/(F(I)*COS(THET(I)-THETB(I)))
      SUM=2.*SUM
      IMAXM2=IMAX-2
```

```
      DO 30 I=3,IMAXM2,2
   30 SUM=SUM+CP(I)*R(I)*SIN(THETB(I))/(F(I)*COS(THET(I)-THETB(I)))
      CDSIMP=4.*DX*SUM/(3.*RMSQ)
      WRITE(6,550)CDSIMP
      RETURN
  540 FORMAT (////47H   DRAG COEFFICIENT BY TRAPEZOIDAL INTEGRATION=,F8.
     15)
  550 FORMAT (//43H   DRAG COEFFICIENT BY SIMPSON INTEGRATION=,F8.5)
 9901 FORMAT(/* NO DRAG BY SIMPSON INTEGRATION BECAUSE IMAX IS EVEN*)
      END
      SUBROUTINE CPPLOT (S,XB,YB,CP,IMAX,CPO,KSTAR)
      DIMENSION S(1), XB(1), YB(1), CP(1), KODE(4), LINE(100)
      DATA KODE/1H ,1H+,1HO,1H*/
      WRITE(6,50)
      DO 10 L=1,100
   10 LINE(L)=KODE(1)
      LINE(KSTAR)=KODE(4)
      DO 40 I=1,IMAX
      K=4.5+30.*(CPO-CP(I))
      IF(K.GT.100) GO TO 30
      LINE(K)=KODE(2)
   30 WRITE(6,70) I,XB(I),YB(I),CP(I),LINE
      LINE(K)=KODE(1)
   40 LINE(KSTAR)=KODE(4)
      RETURN
C
C
   50 FORMAT (33H1PLOT OF CP AT UNEQUAL INCREMENTS///3X,1HI,5X,2HXB
     * ,8X,2HYB,6X,2HCP//)
   70 FORMAT (I4,2F10.3,F8.4,100A1)
      END
      OVERLAY(JERRY,1,5)
      PROGRAM ONES
      COMMON P(81,82)
      COMMON XB(81),YB(81),CP(81)
      COMMON THET(81),THETB(81),ST(81),CT(81),W1(81),W2(81),W3(81)
     * ,W4(81),W5(81),YBP(81),DPO(81),F(81),AK(81),S(81),LS(81),FM(81)
      COMMON AN(81),G(81),GH(81),CB(81),D(81),X1(81),X2(81),M(81),HR(81)
     1,HRP(81),HRM(81),HRMM(81)
      COMMON XS(400),YS(400)
      COMMON ID,ANMAX,DNDYO,YMAX,CD,RMSQ,JSKP
      COMMON /BLOK2/PI,RAD
      COMMON /BLOK3/ IMAX,JMAX,C2,RF1,DPM,IDP,JDP,RPM,IR,JR,NS,GM102
     1,AOSQ,DXSQ,DXDY,DYSQ,DX2,DY2,KLOSE
      COMMON /BLOK5/ JM1,DY,I1,JSUP,JSON,QF3,ISUB,ISUP,SUMRP
      COMMON /BLOK7/ SMAX,S1,XM,XIM,A4,DXIDXO,DXIDXM,A2,A3,XIO,XI1,CXM,
     * DX
      COMMON /BLOK9/ N
      CALL MCHART (P,AK,AN,F,G,YBP,DPO,ST,CT,LS,M,ID,JSKP
     * ,IMAX,JMAX,GM102,AOSQ,DX2,DY2,KLOSE,I1)
      CALL SONLIN (P,F,ST,CT,XB,YB,AK,FM,YBP,D,AN,G,M,XS,YS,ID,N
     * ,IMAX,JMAX,GM102,AOSQ,DX2,DY2,KLOSE,I1,JSUP,JSON)
      IF (N.EQ.0) RETURN
      WRITE(6,50)N
      WRITE(6,60)
      WRITE(6,80) (XS(I),I=1,N)
      WRITE(6,70)
      WRITE(6,80) (YS(I),I=1,N)
      WRITE (6,40)
      RETURN
```

```
 40 FORMAT (//)
 50 FORMAT (7H1     N=,I3//)
 60 FORMAT (18H  XS(K), K=1,...,N/)
 70 FORMAT (///18H  YS(K), K=1,...,N/)
 80 FORMAT(1X,8E10.3)
    END
    SUBROUTINE MCHART (P,AK,AN,F,G,YBP,DPO,ST,CT,LS,M,ID,JSKP
   * ,IMAX,JMAX,GM102,AOSQ,DX2,DY2,KLOSE,I1)
C
C   ------PLOTS CHART OF LOCAL MACH NUMBER, HORIZONTAL ROWS ARE I=CONS
C
    DIMENSION P(ID,1), AK(1), AN(1), F(1), G(1), YBP(1), DPO(1), ST(1)
   1, CT(1), LS(1), M(1)
C
    WRITE(6,170)
    I=1
    DO 10 K=1,JMAX,JSKP
 10 M(K)=JMAX+1-K
    WRITE(6,200) (M(K),K=1,JMAX,JSKP)
    M(JMAX)=100./SQRT(AOSQ-GM102)
    WRITE(6,180)
 20 J=JMAX
    IF (I.EQ.1) GO TO 30
    IM=I-1
    IMM=I-2
    IF (I.EQ.2) IMM=2
    FD=F(I)*DX2
    IF (I.EQ.IMAX) GO TO 30
    IP=I+1
    IPP=I+2
    IF (I.EQ.IMAX-1) IPP=I
 30 JP=J+1
    JM=J-1
    DPJ=G(J)*DY2*(P(I,JM)-P(I,JP))
    IF (I.EQ.1) GO TO 140
    IF (I.EQ.IMAX) GO TO 150
    GO TO 60
 40 DPI=3.*P(I,J)-4.*P(IM,J)+P(IMM,J)
    GO TO 70
 60 DPI=P(IP,J)-P(IM,J)
 70 IF (I.GT.I1) GO TO 80
    U=CT(I)+FD*DPI/(1.+AK(I)*AN(J))
    GO TO 90
 80 U=1.+FD*DPI-YBP(I)*DPJ
 90 V=-ST(I)+DPJ
    UU=U*U
    VV=/*V
    QQ=UU+VV
    AA=AOSQ-GM102*QQ
    IF (AA.GT.0.) GO TO 100
    WRITE(6,160)I,J,UU,VV,QQ,AA
    RETURN
100 CONTINUE
    K=JMAX+1-J
    M(K)=100.*SQRT(QQ/AA)
    IF (J.EQ.2) GO TO 110
    J=J-JSKP
    IF (J.LT.2) GO TO 110
    GO TO 30
110 CONTINUE
```

```
      WRITE(6,190) I,(M(K),K=1,JMAX,JSKP)
      IF (I.EQ.IMAX) GO TO 120
      I=I+1
      GO TO 20
  120 CONTINUE
      WRITE(6,180)
      DO 130 K=1,JMAX,JSKP
  130 M(K)=JMAX+1-K
      WRITE(6,200) (M(K),K=1,JMAX,JSKP)
      RETURN
  140 U=0.
      GO TO 90
  150 IF (KLOSE.EQ.0) GO TO 40
      GO TO 140
C
C
  160 FORMAT (////*        NEGATIVE SPEED OF SOUND OCCURRED IN MCHART AT*
     1* POINT I=*I4,4H, J=,I4//6H   UU=,E11.4,4H VV=E11.4
     2 * QQ=*,E11.4,4H AA=,E11.4////)
  170 FORMAT(/*1     MACH NO. CHART*/)
  180 FORMAT (/)
  190 FORMAT (I4,4H//  ,31I4)
  200 FORMAT (8X,31I4)
      END
      SUBROUTINE SONLIN (P,F,ST,CT,XB,YB,AK,FM,YBP,FJM,AN,G,M,XS,YS,ID,N
     * ,IMAX,JMAX,GM1O2,AOSQ,DX2,DY2,KLOSE,I1,JSUP,JSON)
C
C     -----CALCULATES XS,YS COORDINATES OF SONIC LINE-------
C
      DIMENSION P(ID,1), F(1), ST(1), CT(1), XB(1), YB(1)  AK(1), FM(1),
     1 FJM(1), XS(1), YS(1), YBP(1), AN(1), G(1), M(1)
C
      D(QQ)=AOSQ-GM1O2*QQ
      AMACH(QQ)=SQRT(QQ/D(QQ))
C
      DO 10 J=1,JMAX
   10 M(J)=0
      N=0
      J=JMAX
   20 I=1
   30 DPI=0.
      GO TO 50
   40 DPI=(P(I+1,J)-P(I-1,J))*F(I)*DX2
   50 DPJ=(P(I,J-1)-P(I,J+1))*G(J)*DY2
      HR=1./(1.+AK(I)*AN(J))
      IF (I.GT.I1) HR=1.
      U=CT(I)+DPI*HR-DPJ*YBP(I)
      V=-ST(I)+DPJ
      QQ=U*U+V*V
      DA=D(QQ)
      IF (DA.GT.0.) GO TO 60
      WRITE(6,190) I,J,N,P(I+1,J),P(I-1,J),DPI,P(I,J-1),P(I,J+1),DPJ,U,V
     * ,QQ,DA
      RETURN
   60 CONTINUE
      FM(I)=AMACH(QQ)
      IF (I.EQ.1) GO TO 110
      H1=FM(I-1)-1.
      H=FM(I)-1.
      HS=H-H1
```

```
      IF (HS.GE.0.) GO TO 100
      IF (JSON.EQ.0) GO TO 70
      IF (I.EQ.IMAX.AND.KLOSE.EQ.0) GO TO 120
   70 CONTINUE
      N=N+1
      IF (N.LE.398) GO TO 90
   80 WRITE(6,200)
      RETURN
   90 CONTINUE
      X=XB(I)-AN(J)*ST(I)
      X1=XB(I-1)-AN(J)*ST(I-1)
      Y=YB(I)+AN(J)*CT(I)
      Y1=YB(I-1)+AN(J)*CT(I-1)
      H2=-H1/(H-H1)
      XS(N)=X1+H2*(X-X1)
      YS(N)=Y1+H2*(Y-Y1)
      M(J)=M(J)+1
      IF(N.EQ.398)GO TO 80
  100 IF (I.EQ.IMAX) GO TO 120
  110 I=I+1
      IF (I.EQ.IMAX) GO TO 130
      GO TO 40
  120 IF (M(J).EQ.0) GO TO 140
      J=J-1
      IF (J.EQ.1) GO TO 140
      IF (J.GT.2) GO TO 20
      IF (JSON.EQ.1) RETURN
      GO TO 20
  130 IF (KLOSE.EQ.1) GO TO 30
      DPI=(3.*P(I,J)-4.*P(I-1,J)+P(I-2,J))*F(I)*DX2
      GO TO 50
  140 IF (JSUP.EQ.0) RETURN
      I=1
  150 J=JMAX
  160 V=-ST(I)+G(J)*DY2*(P(I,J-1)-P(I,J+1))
      QQ=V*V
      DA=D(QQ)
      IF (DA.GT.0.) GO TO 170
      WRITE(6,190) I,J,N,P(I,J-1),P(I,J+1),V,QQ,DA
      RETURN
  170 FJM(J)=AMACH(QQ)
      IF (J.EQ.JMAX) GO TO 180
      H1=FJM(J+1)-1.
      H=FJM(J)-1
      HS=H*H1
      IF (HS.GE.0.) GO TO 180
      IF (N.GE.398) GO TO 80
      N=N+1
      X=XB(I)-AN(J)*ST(I)
      X1=XB(I)-AN(J+1)*ST(I)
      H2=-H1/(H-H1)
      XS(N)=X1+H2*(X-X1)
      YS(N)=0.
      IF (KLOSE.EQ.0) RETURN
      IF (I.EQ.IMAX) GO TO 180
      I=IMAX
      GO TO 150
  180 J=J-1
      IF (J.GT.1) GO TO 160
      IF (I.EQ.IMAX) RETURN
```

```
      I=IMAX
      GO TO 150
C
  190 FORMAT (/,* NEGATIVE SQUARE OF SOUND SPEED CALCULATED IN SUBRO*
     1*UTINE SONLIN*//1X,3I3,10E12.4/)
  200 FORMAT (/,*      NO. OF SONIC PTS. EXCEEDS 398. SONIC PT. CALCULA*
     1*TIONS TERMINATED.*)
      END
      OVERLAY(JERRY,1,6)
      PROGRAM ONE6
      COMMON P(81,82)
      COMMON /BLOK3/IMAX,JMAX,DUM(16),KLOSE
C
C     ------HALVES MESH SIZE IN BOTH DIRECTIONS AND USES 4TH-ORDER
C        INTERPOLATION TO DISTRIBUTE POTENTIAL OVER NEW MESH------
C
C
C     ------RENUMBER I-INDEX SUCH THAT I-ODD=KNOWN P,I-EVEN=UNKNOWN P---
      IP=IMAX+1
      M=2*IMAX+1
      DO 10 J=1,JMAX
      DO 10 K=1,IMAX
      M1=M-2*K
      M2=IP-K
   10 P(M1,J)=P(M2,J)
      IMAX=2*IMAX-1
C
C     ------RENUMBER J-INDEX SIMILARLY------
      IMAX=M-2
      JP=JMAX+1
      N=2*JMAX+1
      DO 20 I=1,IMAX,2
      DO 20 K=1,JMAX
      N1=N-2*K
      N2=JP-K
   20 P(I,N1)=P(I,N2)
      JMAX=N-2
      M=IMAX-1
      N=JMAX-1
C
C     ------NOW FILL IN ODD J-ROWS,BUT TREAT I=2 AND I=IMAX-1 FIRST TO
C        ACCOUNT FOR SYMMETRY OR END CONDITION------
C
      DO 30 J=1,JMAX,2
   30 P(2,J)=.5625*P(1,J)+.5*P(3,J)-.0625*P(5,J)
      IF (KLOSE.EQ.1) GO TO 50
C
C     ------I=IMAX   IS NOT A SYMMETRY AXIS, SO USE NONCENTRAL INTERP.--
      DO 40 J=1,JMAX,2
   40 P(M,J)=.3125*(P(M+1,J)-P(M-3,J))+.9375*P(M-1,J)+.0625*P(M-5,J)
      GO TO 70
C
C     ------I=IMAX   IS A SYMMETRY AXIS------
   50 DO 60 J=1,JMAX,2
   60 P(M,J)=.5625*P(M+1,J)+.5*P(M-1,J)-.0625*P(M-3,J)
   70 M=M-2
      DO 80 J=1,JMAX,2
      DO 80 I=4,M,2
   80 P(I,J)=.5625*(P(I+1,J)+P(I-1,J))-.0625*(P(I+3,J)+P(I-3,J))
C
```

```
C     ------NOW ALL I-INDICES ARE KNOWN ON ALL ODD J. FILL IN ALL EVEN J
C        AFTER FIRST TREATING J=2 AND JMAX-1 BY NONCENTRAL INTERP.------
C
      DO 90 I=1,IMAX
   90 P(I,2)=.3125*(P(I,1)-P(I,5))+.9375*P(I,3)+.0625*P(I,7)
      DO 100 I=1,IMAX
  100 P(I,N)=.3125*(P(I,N+1)-P(I,N-3))+.9375*P(I,N-1)+.0625*P(I,N-5)
      N=N-2
      DO 110 I=1,IMAX
      DO 110 J=4,N,2
  110 P(I,J)=.5625*(P(I,J+1)+P(I,J-1))-.0625*(P(I,J+3)+P(I,J-3))
      RETURN
      END
      OVERLAY(JERRY,2,0)
      PROGRAM TWOO
      DIMENSION XB(200), YB(200), CP(200), DESC(8)
      DIMENSION XS(400), YS(400)
      DIMENSION AN(100),ST(200),CT(200),D1(200),D2(200)
      CALL PSEUDO
      CALL LEROY
      REWIND 4
   10 READ (4) DESC
      IF(EOF(4)) 20,30
   20 CALL CALPLT (0,0,999)
      RETURN
   30 CONTINUE
      READ (4) IMAX,JMAX,IT,KLOSE,NSL
      READ(4)CPSTAR,AMINF,DPM,XREF,DXIDXO,DNDYO,QF3
      IF(KLOSE.EQ.1)GO TO 31
      READ(4)CXM,XM,XIM,DXIDXM
   31 CONTINUE
      READ (4)(AN(J),J=2,JMAX)
      READ (4)(ST(I),I=1,IMAX)
      READ (4)(CT(I),I=1,IMAX)
      READ (4) (XB(I),I=1,IMAX)
      READ (4) (YB(I),I=1,IMAX)
      READ (4) (CP(I),I=1,IMAX)
      IF (NSL.EQ.0) GO TO 40
      READ (4) (XS(I),I=1,NSL)
      READ (4) (YS(I),I=1,NSL)
   40 CONTINUE
      CALL GRID(IMAX,JMAX,XB,YB,ST,CT,AN,D1,D2,XREF,KLOSE)
      CALL PLOT (IMAX,JMAX,KLOSE,XB,YB,CP,DESC,IT,AMINF,CPSTAR,DPM,XR
     1EF,DXIDXO,DNDYO,CXM,XM,XIM,DXIDXM,QF3)
      IF (NSL.EQ.0) GO TO 10
      CALL SONPLT (XB,YB,XS,YS,NSL,IMAX,XREF,KLOSE)
      GO TO 10
      END
      SUBROUTINE GRID(IMAX,JMAX,XB,YB,ST,CT,AN,D1,D2,XREF,KLOSE)
      DIMENSION XB(1),YB(1),AN(1),D1(1),D2(1),ST(1),CT(1)
      DXR=5./XREF
      XSHIFT=3.
      YSHIFT=2.5
      IM=IMAX-1+KLOSE
      DO 2 J=2,JMAX
      DO 1 I=1,IM
      D1(I)=(XB(I)-AN(J)*ST(I))*DXR + XSHIFT
      D2(I)=(YB(I)+AN(J)*CT(I))*DXR + YSHIFT
    1 CONTINUE
      CALL DRAW(D1,D2,IM)
```

```
    2 CONTINUE
      DO 3 I=1,IM
      D2(I)=-YB(I)*DXR+YSHIFT
    3 CONTINUE
      CALL DRAW(D1,D2,IM)
      DO 5 I=1,IM
      DO 4 J=2,JMAX
      D1(J-1)=(XB(I)-AN(J)*ST(I))*DXR + XSHIFT
      D2(J-1)=(YB(I)+AN(J)*CT(I))*DXR + YSHIFT
    4 CONTINUE
      CALL DRAW(D1,D2,JMAX-1)
    5 CONTINUE
      CALL NFRAME
      RETURN
      END
      SUBROUTINE PLOT (IMAX,JMAX,KLOSE,XB,YB,CP,DESC,IT,AMINF,CPSTAR,
     1 DPM,XREF,DXIDXO,DNDYO,CXM,XM,XIM,DXIDXM,QF3)
      DIMENSION T(30),LBLE(8),NAME(13)
      DIMENSION XB(1), YB(1), CP(1), DESC(1)
      DATA NREAD/0/
      DATA NAME/2HM=,7H, IMAX=,7H, JMAX=,5H, IT=,6H, DPM=,7HDXIDXO=,
     1 8H, DNDYO=,5H CXM=,5H, XM=,6H, XIM=,9H, DXIDXM=,6H   CD=
     * ,6H, QF3=/
      NREAD=NREAD+1
      IF (NREAD.GT.1) GO TO 10
      CALL JPARAMS(T)
      YPG=7.
      YDV=0.
      YTIC=-1.
   10 CONTINUE
      CALL CALPLT(2.0,2.5,-3)
      IM=IMAX-1+KLOSE
      XB(IM+1)=0.
      YB(IM+1)=XB(IM+1)
      XB(IM+2)=.2
      YB(IM+2)=XB(IM+2)
      YMAX=0.
      DXR=1./XREF
      DO 20 I=1,IM
      XB(I)=XB(I)*DXR
      YB(I)=YB(I)*DXR
      IF (YB(I)-YMAX.LE.0.) GO TO 20
      YMAX=YB(I)
   20 CONTINUE
      NBOD=0
      IF (YMAX*5..GE.1.3+1.E-06) NBOD=2
      IF (YMAX*5..GE.2.5+1.E-06) NBOD=1
      CP(IM+1)=1.5
      CP(IM+2)=-.5
      BL=CP(IM+1)
      TL=BL+YPG*CP(IM+2)
      PYD=(BL-CPSTAR)/CP(IM+2)
      PYU=-PYD
      DO 40 I=1,IM
      D=(CP(I)-CP(IM+1))/CP(IM+2)+2.5
      IF (D.LE.10.) GO TO 30
      CP(I)=CP(IM+1)+7.5*CP(IM+2)
      GO TO 40
   30 IF (D.GE.0.) GO TO 40
      CP(I)=CP(IM+1)-2.5*CP(IM+2)
```

```
   40 CONTINUE
      NSYM=22
      NLINE=1
      CALL AXES (-.5,0.,90.,YPG,CP(IM+1),CP(IM+2),YTIC,YDV,2HCP,.20,
     1+2)
      IF (CPSTAR.LT.TL.OR.CPSTAR.GT.BL) GO TO 50
C
C     DRAW LINE FOR CPSTAR
C
      CALL CALPLT (-.5,PYU,3)
      CALL CALPLT (.28,PYU,2)
C
C     PLOT  CP
C
   50 CALL LINPLT (XB,CP,IM,1,NLINE,NSYM,1,0)
      IF (NBOD.EQ.1) GO TO 70
C
C     PLOT BODY
C
      CALL LINPLT (XB,YB,IM,1,0,0,1,0)
      IF (NBOD.EQ.2) GO TO 70
      DO 60 I=1,IM
   60 YB(I)=-YB(I)
      CALL LINPLT (XB,YB,IM,1,0,0,1,0)
   70 CONTINUE
C
C     ADD LABELS
C
      CALL NOTATE(-.5,-1.39,.14,DESC,0.,80)
      ENCODE(50,80,LBLE)NAME(1),AMINF,NAME(2),IMAX,NAME(3),JMAX,NAME(4),
     1IT,NAME(5),DPM
      CALL NOTATE(-.5,-1.64,.14,LBLE,0.,50)
      ENCODE(39,90,LBLE)NAME(6),DXIDXO,NAME(7),DNDYO,NAME(13),QF3
      CALL NOTATE(-.5,-1.89,.14,LBLE,0.,39)
      IF(KLOSE.EQ.0)GO TO 72
      DY=-2.14
   71 ENCODE(17,100,LBLE)T(1),T(23)
      CALL NOTATE(-.5,DY,.14,LBLE,0.,17)
      GO TO 73
   72 ENCODE(54,110,LBLE)NAME(8),CXM,NAME(9),XM,NAME(10),XIM,NAME(11),
     1DXIDXM
      CALL NOTATE(-.5,-2.14,.14,LBLE,0.,54)
      DY=-2.39
      GO TO 71
   73 CONTINUE
      CALL NFRAME
      RETURN
C
   80 FORMAT(A2,F5.3,A7,I3,A7,I3,A5,I4,A6,E8.2)
   90 FORMAT(A7,F5.2,A8,E8.2,A6,F5.2)
  100 FORMAT(A7,A10)
  110 FORMAT(A5,E8.2,A5,E8.2,A6,E8.2,A9,F5.2)
      END
      SUBROUTINE BONPLT (XB,YB,XS,YS,NSL,IMAX,XREF,KLOSE)
C
C     -----SCALES AND PLOTS BODY AND SONIC LINES-----
C
      DIMENSION XB(200), YB(200), XS(400), YS(400)
C
      IM=IMAX-1+KLOSE
```

```
      DXR=1./XREF
      DO 10 I=1,NSL
      XS(I)=XS(I)*DXR
   10 YS(I)=YS(I)*DXR
      CALL DSCALE (XS,NSL,XSMAX,XSMIN)
      CALL DSCALE (YS,NSL,YSMAX,YSMIN)
      XMAX=XSMAX
      IF (XB(IM).GT.XMAX) XMAX=XB(IM)
      XMIN=0.
      IF (XSMIN.LT.0.) XMIN=XSMIN
      DO 20 I=1,IM
   20 YB(I)=ABS(YB(I))
      CALL DSCALE (YB,IM,YBMAX,YBMIN)
      DX=XMAX-XMIN
      DY=YSMAX
      L=1
      DXR=1.
   30 IF (DX*DXR.LE.2.4+1.E-08) GO TO 60
C ***********************************************************************
C
C----- FOLLOWING CARD GIVES FURTHER SIZE REDUCTION IF YOU REMOVE COMMENT
C     GO TO (40,50), L
C
C ***********************************************************************
      IF(L.EQ.2) GO TO 60
   40 DXR=.5*DXR
      L=2
      GO TO 30
   50 DXR=.4*DXR
      L=3
   60 IF (DY*DXR.LE.1.5+1.E-08) GO TO 90
      GO TO (70,80,90), L
   70 DXR=.5*DXR
      L=2
      GO TO 60
   80 DXR=.4*DXR
      L=3
      GO TO 60
   90 K=0
      DO 110 I=1,NSL
      IF (L.EQ.1) GO TO 100
      XS(I)=XS(I)*DXR
      YS(I)=YS(I)*DXR
  100 IF (YS(I)-1.5.GT.1.E-08) GO TO 110
      K=K+1
      XS(K)=XS(I)
      YS(K)=YS(I)
  110 CONTINUE
      IF (L.EQ.1) GO TO 130
      DO 120 I=1,IM
      XB(I)=DXR*XB(I)
  120 YB(I)=DXR*YB(I)
      XMIN=XMIN*DXR
      YBMAX=YBMAX*DXR
      XMAX=XMAX*DXR
  130 XB(IM+1)=0.
      YB(IM+1)=XB(IM+1)
      XS(K+1)=XB(IM+1)
      YS(K+1)=XB(IM+1)
      XB(IM+2)=.2
```

```
      YB(IM+2)=XB(IM+2)
      XS(K+2)=XB(IM+2)
      YS(K+2)=XB(IM+2)
      SKIP=5.*ABS(XMIN)+2.
      NSKIP=SKIP
      SKIP=NSKIP
      CALL CALPLT(SKIP,2.5,-3)
      CALL LINPLT (XB,YB,IM,1,0,0,1,0)
      IF (YBMAX*5..GT.2.5+1.E-08) GO TO 150
      DO 140 I=1,IM
  140 YB(I)=-YB(I)
      CALL LINPLT (XB,YB,IM,1,0,0,1,0)
  150 CALL LINPLT (XS,YS,K,1,-1,22,1,0)
      CALL NFRAME
      RETURN
      END
      SUBROUTINE DSCALE (X,N,XMAX,XMIN)
C
C     *****COMPUTES MAX AND MIN OF X-ARRAY*****
C
      DIMENSION X(1)
C
      XMIN=X(1)
      XMAX=XMIN
      DO 20 I=1,N
      IF (X(I).GE.XMIN) GO TO 10
      XMIN=X(I)
      GO TO 20
   10 IF (X(I).LE.XMAX) GO TO 20
      XMAX=X(I)
   20 CONTINUE
      RETURN
      END
```

APPENDIX D

SAMPLE CASES

The input for the sample cases is listed below. The output for these cases is on the following pages. Note that the sample cases are for only 25 cycles on the crude grid. In actual usage there might be more cycles and some grid refinements. The plotted output is shown in figures 1 to 8.

```
10-1 ELLIPSOID
  35
0.000000  0.012311  0.038155  0.072346  0.113710  0.161530  0.21521   0.27421
0.33799   0.40601   0.47777   0.55274   0.63041   0.71027   0.79181   0.87452
0.95789   1.04140   1.1246    1.2069    1.2876    1.3668    1.4435    1.5173
1.5877    1.6543    1.7164    1.7737    1.8255    1.8715    1.9110    1.9436
1.9688    1.987689  2.
.000000   .015643   .027360   .037344   .046312   .054495   .061977   .068792
.074949   .080447   .085280   .089440   .092919   .095711   .097809   .099210
.099911   .099914   .099221   .097837   .095770   .093029   .089627   .085580
.080906   .075626   .069768   .063360   .056439   .049047   .041241   .033100
.024765   .015643   0.
999.      999.      .1        2.
   21   21   25    0    1    1
1.4       1.0       0.
.5        1.3       .084
1.4       .995
10-1 ELLIPSOID WITH 20-PERCENT STING
  34
0.0000    0.012311  0.038155  0.072346  0.113710  0.161530  0.21521   0.27421
0.33799   0.40601   0.47777   0.55274   0.63041   0.71027   0.79181   0.87452
0.95789   1.00140   1.1246    1.2069    1.2876    1.3668    1.4435    1.5173
1.5877    1.6543    1.7164    1.7737    1.8255    1.8715    1.9110    1.9436
1.9688    1.9797959
.000000   .015643   .027360   .037344   .046312   .054495   .061977   .066792
.074949   .080447   .085280   .089440   .092919   .095711   .097809   .099210
.099911   .099914   .099221   .097837   .095770   .093029   .089627   .085560
.080906   .075626   .069768   .063360   .056439   .049047   .041241   .033100
.024765   .020000
9999.     -.48989795 .1       2.
   21   21   25    0    0    1
1.4       1.0       0.
.5        1.3       .084      1.9797959 .75       2.
1.4       .995
 SPHERE/15-DEG CONE/CYLINDER/15-DEG FLARE
  40
 .0        1.637E-03 7.003E-03 1.751E-02 3.564E-02 6.509E-02 1.107E-01 1.782E-01
 2.727E-01 3.969E-01 5.430E-01 7.106E-01 8.976E-01 1.105E+00 1.332E+00 1.576E+00
 1.842E+00 2.123E+00 2.421E+00 2.738E+00 3.065E+00 3.398E+00 3.735E+00 4.072E+00
 4.406E+00 4.734E+00 5.053E+00 5.361E+00 5.655E+00 5.934E+00 6.198E+00 6.448E+00
 6.684E+00 6.910E+00 7.133E+00 7.357E+00 7.594E+00 7.856E+00 8.156E+00 8.511E+00
 .0        4.043E-02 8.339E-02 1.312E-01 1.654E-01 2.467E-01 3.138E-01 3.827E-01
 4.453E-01 4.900E-01 5.294E-01 5.741E-01 6.242E-01 6.797E-01 7.406E-01 8.066E-01
 8.773E-01 9.524E-01 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E+00 1.000E+00
 1.028E+00 1.116E+00 1.202E+00 1.284E+00 1.363E+00 1.438E+00 1.509E+00 1.575E+00
 1.639E+00 1.699E+00 1.759E+00 1.819E+00 1.883E+00 1.953E+00 2.033E+00 2.074E+00
999.      .26795    2.        8.5
   21   21   25    0    0    1
1.4       1.0       0.
2.857142R6 1.3      2.        8.        .75       15.
1.4       .8
```

10-1 ELLIPSOID

INPUT COORDINATES

I	X	Y
1	0.000000	0.000000
2	.012311	.015643
3	.038155	.027360
4	.072346	.037344
5	.113710	.046312
6	.161530	.054495
7	.215210	.061977
8	.274210	.068792
9	.337990	.074949
10	.406010	.080447
11	.477770	.085280
12	.552740	.089440
13	.630410	.092919
14	.710270	.095711
15	.791810	.097809
16	.874520	.099210
17	.957690	.099911
18	1.041400	.099914
19	1.124600	.099221
20	1.206900	.097837
21	1.287800	.095770
22	1.366800	.093029
23	1.443500	.089627
24	1.517300	.085580
25	1.587700	.080906
26	1.654300	.075626
27	1.716400	.069768
28	1.773700	.063360
29	1.825500	.056439
30	1.871500	.049047
31	1.911000	.041241
32	1.943600	.033100
33	1.968800	.024765
34	1.987689	.015643
35	2.000000	0.000000

```
DYDXN=  999.0000
DYDXI=  999.0000
YMAX=      .1000
XREF=     2.0000

IMAX= 21
JMAX= 21
MIT=  25
MHALF=0
KLOSE=1
NPLOT=1
RF1=1.400
COVERG=  .10E+01
OF3= 0.
DNDYO=  .500E+00
ALF=1.30
DXIDXO=  .840E-01
XM=-0.
CXM=-0.
DXIDXM=-0.
GAM=1.40
AMINF= .9950
```

I	S	X	Y	THET	THETB	AK	F
1	0.	0.	0.	.9000E+02	.9000E+02	.8967E+02	.1190E+02
2	.1832E-01	.1011E-01	.1422E-01	.3252E+02	.3252E+02	.1901E+02	.1564E+01
3	.6295E-01	.5111E-01	.3153E-01	.1676E+02	.1676E+02	.2661E+01	.8803E+00
4	.1310E+00	.1174E+00	.4701E-01	.1073E+02	.1073E+02	.8574E+00	.6352E+00
5	.2194E+00	.2048E+00	.6063E-01	.7488E+01	.7488E+01	.4380E+00	.5117E+00
6	.3254E+00	.3101E+00	.7239E-01	.5444E+01	.5444E+01	.2618E+00	.4394E+00
7	.4460E+00	.4303E+00	.8219E-01	.3966E+01	.3966E+01	.1793E+00	.3939E+00
8	.5783E+00	.5624E+00	.8992E-01	.2787E+01	.2787E+01	.1370E+00	.3645E+00
9	.7194E+00	.7033E+00	.9550E-01	.1780E+01	.1780E+01	.1147E+00	.3461E+00
10	.8663E+00	.8502E+00	.9887E-01	.8682E+00	.8682E+00	.1036E+00	.3359E+00
11	.1016E+01	.1000E+01	.1000E+00	.5879E-02	.5879E-02	.9904E-01	.3326E+00
12	.1166E+01	.1150E+01	.9887E-01	-.8677E+00	-.8677E+00	.1035E+00	.3359E+00
13	.1313E+01	.1297E+01	.9550E-01	-.1780E+01	-.1780E+01	.1151E+00	.3461E+00
14	.1454E+01	.1438E+01	.8992E-01	-.2786E+01	-.2786E+01	.1367E+00	.3645E+00
15	.1586E+01	.1570E+01	.8218E-01	-.3966E+01	-.3966E+01	.1766E+00	.3939E+00
16	.1707E+01	.1690E+01	.7239E-01	-.5449E+01	-.5449E+01	.2588E+00	.4394E+00
17	.1813E+01	.1795E+01	.6063E-01	-.7477E+01	-.7477E+01	.4404E+00	.5117E+00
18	.1901E+01	.1863E+01	.4702E-01	-.1059E+02	-.1059E+02	.9311E+00	.6352E+00
19	.1969E+01	.1949E+01	.3156E-01	-.1636E+02	-.1636E+02	.2813E+01	.8803E+00
20	.2014E+01	.1990E+01	.1423E-01	-.3254E+02	-.3254E+02	.1936E+02	.1564E+01
21	.2032E+01	.2000E+01	.1432E-13	-.9000E+02	-.9000E+02	.8982E+02	.1190E+02

----- NORMAL COORD, STRETCH FOR ALF= 1,300 -----

J	AN	G	GH
1	0.	0.	.7920E-03
2	.2334E+02	.1584E-02	.4738E-02
3	.8979E+01	.7893E-02	.1409E-01
4	.5006E+01	.2030E-01	.3005E-01
5	.3241E+01	.3981E-01	.5357E-01
6	.2274E+01	.6732E-01	.8549E-01
7	.1674E+01	.1037E+00	.1266E+00
8	.1272E+01	.1496E+00	.1778E+00
9	.9873E+00	.206[illegible]E+00	.2398E+00
10	.7755E+00	.2735E+00	.3134E+00
11	.6156E+00	.3532E+00	.3993E+00
12	.4895E+00	.4[illegible]5E+00	.4985E+00
13	.3885E+00	.5515E+00	.6118E+00
14	.3060E+00	.6720E+00	.7399E+00
15	.2385E+00	.8078E+00	.8839E+00
16	.1817E+00	.9600E+00	.1045E+01
17	.1337E+00	.1129E+01	.12[illegible]E+01
18	.9264E-01	.1317E+01	.1420E+01
19	.5734E-01	.1524E+01	.1638E+01
20	.2672E-01	.1751E+01	.1876E+01
21	0.	.2000E+01	.2136E+01

CPU SECONDS FOR BODY GEOMETRY COMPUTATIONS= .109

IT	DPMAX	ID	JD	RMAX	IR	JR	ISUB	IBUP	RAVG	RF1	OF3	NB	SEC/CYC
1	,917E-02	3	21	,738E+03	21	21	1	0	,249E+01	1,400	0,000	0	,216
2	,562E-02	4	21	,668E+03	1	21	1	0	,326E+01	1,400	0,000	57	,239
3	,386E-02	5	21	,550E+03	1	21	1	0	,281E+01	1,400	0,000	117	,256
4	,298E-02	5	21	,463E+03	1	21	1	0	,233E+01	1,400	0,000	136	,257
5	,239E-02	17	21	,353E+03	1	21	1	0	,184E+01	1,400	0,000	149	,261
6	,198E-02	17	21	,270E+03	1	21	1	0	,147E+01	1,400	0,000	153	,260
7	,165E-02	17	21	,217E+03	1	21	1	0	,122E+01	1,400	0,000	156	,261
8	,141E-02	16	21	,181E+03	1	21	1	0	,103E+01	1,400	0,000	157	,262
9	,124E-02	16	21	,155E+03	1	21	1	0	,891E+00	1,400	0,000	158	,268
10	,108E-02	16	21	,135E+03	1	21	1	0	,776E+00	1,400	0,000	160	,261
11	,952E-03	16	21	,118E+03	1	21	1	0	,681E+00	1,400	0,000	158	,267
12	,840E-03	16	21	,105E+03	1	21	1	0	,603E+00	1,400	0,000	159	,268
13	,741E-03	16	21	,933E+02	1	21	1	0	,535E+00	1,400	0,000	161	,266
14	,650E-03	16	21	,838E+02	1	21	1	0	,476E+00	1,400	0,000	161	,263
15	,570E-03	16	21	,756E+02	1	21	1	0	,426E+00	1,400	0,000	161	,267
16	,5[illegible]-03	8	21	,686E+02	1	21	1	0	,383E+00	1,400	0,000	161	,264
17	,456E-03	8	21	,625E+02	1	21	1	0	,345E+00	1,400	0,000	161	,262
18	,413E-03	8	21	,571E+02	1	21	1	0	,312E+00	1,400	0,000	161	,263
19	,373E-03	8	21	,524E+02	1	21	1	0	,284E+00	1,400	0,000	162	,266
20	,339E-03	8	21	,482E+02	1	21	1	0	,258E+00	1,400	0,000	163	,264
21	,3[illegible]E-03	9	21	,444E+02	1	21	1	0	,233E+00	1,400	0,000	164	,263
22	,283E-03	9	21	,411E+02	1	21	1	0	,211E+00	1,400	0,000	164	,274
23	,260E-03	9	21	,381E+02	1	21	1	0	,192E+00	1,400	0,000	164	,270
24	,238E-03	9	21	,354E+02	1	21	1	0	,174E+00	1,400	0,000	164	,263
25	,219E-03	9	21	,329E+02	1	21	1	0	,159E+00	1,400	0,000	164	,265

----DID NOT CONVERGE IN 25 CYCLES,---- RMAX= ,33E+02, COVR= ,25E-02

CPU SECONDS= 6,61 FOR 25 ITERATIONS, NHALF=0

I	SB	XB	YB	CP	M
1	0,000	0,000	0,000	1,27261	0,00000
2	,018	,010	,014	,52903	,69183
3	,063	,051	,032	,23626	,85794
4	,131	,117	,047	,04816	,96649
5	,219	,205	,061	-,01530	1,00414
6	,325	,310	,072	-,04232	1,02038
7	,446	,430	,082	-,05810	1,02994
8	,578	,562	,090	-,06862	1,03646
9	,719	,70[illegible]	,095	-,07636	1,04106
10	,866	,850	,099	-,08163	1,04428
11	1,016	1,000	,100	-,08578	1,04682
12	1,166	1,150	,099	-,08945	1,04907
13	1,313	1,297	,095	-,09318	1,05137
14	1,454	1,438	,090	-,09961	1,05532
15	1,586	1,570	,082	-,09859	1,05469
16	1,707	1,690	,072	-,06725	1,03550
17	1,813	1,795	,061	-,01765	1,00567
18	1,901	1,883	,047	,04976	,96554
19	1,969	1,949	,032	,23052	,86120
20	2,014	1,990	,014	,51904	,69755
21	2,032	2,000	,000	1,27261	0,00000

DRAG COEFFICIENT BY TRAPEZOIDAL INTEGRATION= ,01635

DRAG COEFFICIENT BY SIMPSON INTEGRATION= ,01608

PLOT OF CP AT UNEQUAL INCREMENTS

I	XB	YB	CP
1	0,000	0,000	1,2726
2	,010	,014	,5290
3	,051	,032	,2363
4	,117	,047	,0482
5	,205	,061	-,0153
6	,310	,072	-,0423
7	,430	,082	-,0581
8	,562	,090	-,0688
9	,703	,095	-,0764
10	,850	,099	-,0816
11	1,000	,100	-,0858
12	1,150	,099	-,0894
13	1,297	,095	-,0932
14	1,438	,090	-,0996
15	1,570	,082	-,0986
16	1,690	,072	-,0672
17	1,795	,061	-,0179
18	1,883	,047	,0498
19	1,949	,032	,2305
20	1,990	,014	,5190
21	2,000	,000	1,2726

MACH NO. CHART

	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
1//	0	85	90	93	95	96	97	97	98	98	98	98	99	99	99	99	99	99	99	99	99
2//	69	84	90	93	95	96	97	97	98	98	98	98	99	99	99	99	99	99	99	99	99
3//	85	86	88	90	91	93	94	95	96	96	97	97	98	98	98	99	99	99	99	99	99
4//	96	95	95	95	95	95	95	95	96	96	97	97	97	98	98	98	99	99	99	99	99
5//	100	99	98	98	98	98	97	97	97	97	97	97	98	98	98	98	99	99	99	99	99
6//	102	101	100	100	100	100	99	99	99	99	98	98	98	98	98	98	99	99	99	99	99
7//	102	102	102	101	101	101	100	100	100	100	100	99	99	99	99	99	99	99	99	99	99
8//	103	103	102	102	102	102	101	101	101	101	100	100	100	100	99	99	99	99	99	99	99
9//	104	103	103	103	102	102	102	102	101	101	101	101	100	100	100	100	99	99	99	99	99
10//	104	104	103	103	103	103	102	102	102	102	101	101	101	101	100	100	100	99	99	99	99
11//	104	104	104	103	103	103	103	102	102	102	102	102	101	101	101	100	100	100	99	99	99
12//	104	104	104	104	103	103	103	103	103	102	102	102	102	101	101	101	100	100	99	99	99
13//	105	104	104	104	104	103	103	103	103	102	102	102	102	102	101	101	101	100	99	99	99
14//	105	105	104	104	104	104	103	103	103	103	103	102	102	102	101	101	100	100	99	99	99
15//	105	105	104	104	104	103	103	103	102	102	102	101	101	101	100	100	99	99	99	99	99
16//	103	102	102	102	101	101	101	100	100	100	100	99	99	99	99	99	99	99	99	99	99
17//	100	99	99	98	98	98	98	97	97	97	97	98	98	98	98	98	99	99	99	99	99
18//	96	95	95	95	95	95	95	95	96	96	97	97	98	98	98	99	99	99	99	99	99
19//	86	87	89	90	92	93	94	95	96	96	97	98	98	98	99	99	99	99	99	99	99
20//	69	85	90	93	95	96	97	97	98	98	98	99	99	99	99	99	99	99	99	99	99
21//	0	85	91	93	95	96	97	97	98	98	98	99	99	99	99	99	99	99	99	99	99
	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

```
N= 36

XS(K), K=1,.....,N

 .195E+00  .181E+01  .228E+00  .179E+01  .252E+00  .178E+01  .269E+00  .177E+01
 .281E+00  .176E+01  .292E+00  .176E+01  .313E+00  .176E+01  .332E+00  .175E+01
 .350E+00  .175E+01  .368E+00  .175E+01  .387E+00  .175E+01  .417E+00  .175E+01
 .448E+00  .175E+01  .484E+00  .176E+01  .535E+00  .177E+01  .602E+00  .178E+01
 .704E+00  .178E+01  .882E+00  .179E+01


YS(K), K=1,.....,N

 .591E-01  .587E-01  .902E-01  .883E-01  .124E+00  .120E+00  .161E+00  .157E+00
 .204E+00  .199E+00  .253E+00  .248E+00  .312E+00  .305E+00  .382E+00  .373E+00
 .465E+00  .456E+00  .568E+00  .558E+00  .696E+00  .685E+00  .859E+00  .847E+00
 .107E+01  .106E+01  .136E+01  .134E+01  .176E+01  .175E+01  .237E+01  .235E+01
 .334E+01  .332E+01  .510E+01  .508E+01


CPU SECONDS TO COMPUTE AND PLOT CP AND MCHART=    .383


10-1 ELLIPSOID WITH 20-PERCENT STING

INPUT COORDINATES
   I      X          Y
   1  0.000000   0.000000
   2   .012311    .015643
   3   .038155    .027360
   4   .072346    .037344
   5   .113710    .046312
   6   .161530    .054495
   7   .215210    .061977
   8   .274210    .068792
   9   .337490    .074949
  10   .406010    .080447
  11   .477770    .085280
  12   .552740    .089440
  13   .630410    .092419
  14   .710270    .095711
  15   .791810    .097609
  16   .874520    .099210
  17   .957690    .099911
  18  1.041400    .099914
  19  1.124600    .099221
  20  1.206900    .097837
  21  1.287800    .095770
  22  1.366800    .093029
  23  1.443500    .089627
  24  1.517300    .085580
  25  1.587700    .080906
  26  1.654300    .075626
  27  1.716400    .069766
  28  1.773700    .063360
  29  1.825500    .056439
  30  1.871500    .049047
  31  1.911000    .041241
  32  1.943600    .033100
  33  1.968800    .024765
  34  1.979796    .020000

DYDXN= 9999.0000
DYDXT=     -.4899
YMAX=       .1000
XREF=      2.0000
```

```
IMAX= 21
JMAX= 21
MIT=  25
MHALF=0
KLOSE=0
NPLOT=1
RF1=1,400
COVERG=  ,10E+01
QF3= 0,
DNDYO=  ,500E+00
ALF=1,30
DXIDXO=  ,840E-01
XM=  ,198E+01
CXM=  ,750E+00
DXIDXM=  ,200E+01
GAM=1,40
AMINF= ,9950
```

I	S	X	Y	THET	THETB	AK	F
1	0,	0,	0,	,9000E+02	,9000E+02	,6967E+02	,1190E+02
2	,6441E-02	,1581E-02	,5874E-02	,5921E+02	,5921E+02	,6364E+02	,4567E+01
3	,2604E-01	,1699E-01	,1837E-01	,2781E+02	,2781E+02	,1181E+02	,1651E+01
4	,7056E-01	,5843E-01	,3366E-01	,1533E+02	,1533E+02	,2526E+01	,6247E+00
5	,1490E+00	,1351E+00	,5020E-01	,9765E+01	,9765E+01	,7493E+00	,5128E+00
6	,2666E+00	,2516E+00	,6632E-01	,6431E+01	,6431E+01	,3361E+00	,3636E+00
7	,4236E+00	,4062E+00	,8061E-01	,4199E+01	,4199E+01	,1667E+00	,2640E+00
8	,6163E+00	,6004E+00	,9167E-01	,2496E+01	,2496E+01	[illegible]	[illegible]
9	[illegible]	,8187E+00	,9834E-01	,1057E+01	,1057E+01	,1051E+00	[illegible]
10	,1066E+01	,1050E+01	,9968E-01	0,	-,2843E+00	,1002E+00	[illegible]
11	[illegible]	,1277E+01	,9609E-01	0,	-,1651E+01	,1132E+00	,2274E+00
12	,1500E+01	,1464E+01	,8750E-01	0,	-,3167E+01	,1503E+00	[illegible]
13	,1674E+01	,1658E+01	,7534E-01	0,	-,4991E+01	,2376E+00	,3255E+00
14	,1807E+01	,1791E+01	,6120E-01	0,	-,7372E+01	,4418E+00	[illegible]
15	,1906E+01	,1890E+01	,4562E-01	0,	-,1105E+02	,1060E+01	,5862E+00
16	,1996E+01	,1980E+01	,2000E-01	0,	-,2610E+02	,1258E+02	,5000E+00
17	,2121E+01	,2105E+01	,2000E-01	0,	0,	0,	,3200E+00
18	,2324E+01	,2313E+01	,2000E-01	0,	0,	0,	,1800E+00
19	,2746E+01	,2730E+01	,2000E-01	0,	0,	0,	,6000E-01
20	,3996E+01	,3980E+01	,2000E-01	0,	0,	0,	,2000E-01
21	,1000E+31	,1000E+31	,2000E-01	0,	0,	0,	,1000E-29

----- NORMAL COORD, STRETCH FOR ALF= 1,300 -----

J	AN	G	GH
1	0,	0,	,7920E-03
2	,2334E+02	,1584E-02	,4738E-02
3	,8979E+01	,7893E-02	,1409E-01
4	,5006E+01	,2030E-01	,3005E-01
5	,3241E+01	,3981E-01	,5357E-01
6	,2274E+01	,6732E-01	,8549E-01
7	,1674E+01	,1037E+00	,1266E+00
8	,1272E+01	,1496E+00	,1778E+00
9	,9873E+00	,2060E+00	,2398E+00
10	,7765E+00	,2736E+00	,3134E+00
11	,6156E+00	,3532E+00	,3993E+00
12	,4895E+00	,4455E+00	,4985E+00
13	,3885E+00	,5515E+00	,6118E+00
14	,3064E+00	,6720E+00	,7399E+00
15	,2385E+00	,8078E+00	,8839E+00
16	,1817E+00	,9600E+00	,1045E+01
17	,1337E+00	,1129E+01	,1223E+01
18	,9264E-01	,1317E+01	,1420E+01
19	,5734E-01	,1524E+01	,1638E+01
20	,2672E-01	,1751E+01	,1876E+01
21	0,	,2000E+01	,2136E+01

CPU SECONDS FOR BODY GEOMETRY COMPUTATIONS= ,112

IT	DPMAX	ID	JD	RMAX	IR	JR	ISUB	ISUP	RAVG	RF1	QFS	NS	SEC/CYC
1	,914E-02	4	21	,939E+02	1	21	1	0	,627E+00	1,400	0,000	0	,209
2	,579E-02	15	21	,703E+03	1	21	1	0	,202E+01	1,400	0,000	55	,227
3	,519E-02	2	21	,572E+03	1	21	1	0	,186E+01	1,400	0,000	101	,237
4	,577E-02	1	21	,950E+03	1	21	1	0	,276E+01	1,400	0,000	113	,239
5	,417E-02	1	21	,678E+03	1	21	1	0	,199E+01	1,400	0,000	114	,242
6	,285E-02	1	20	,256E+03	1	21	1	0	,870E+00	1,400	0,000	115	,236
7	,135E-02	1	20	,606E+02	1	21	1	0	,321E+00	1,400	0,000	114	,245
8	,138E-02	7	21	,794E+02	1	21	1	0	,334E+00	1,400	0,000	113	,244
9	,120E-02	7	21	,143E+03	1	21	1	0	,474E+00	1,400	0,000	113	,247
10	,104E-02	7	21	,154E+03	1	21	1	0	,489E+00	1,400	0,000	113	,247
11	,899E-03	7	21	,122E+03	1	21	1	0	,397E+00	1,400	0,000	113	,241
12	,778E-03	7	21	,880E+02	1	21	1	0	,300E+00	1,400	0,000	112	,245
13	,675E-03	7	21	,708E+02	1	21	1	0	,247E+00	1,400	0,000	113	,240
14	,601E-03	8	21	,659E+02	1	21	1	0	,226E+00	1,400	0,000	114	,240
15	,539E-03	8	21	,625E+02	1	21	1	0	,210E+00	1,400	0,000	112	,246
16	,481E-03	8	21	,564E+02	1	21	1	0	,189E+00	1,400	0,000	113	,247
17	,430E-03	8	21	,491E+02	1	21	1	0	,166E+00	1,400	0,000	113	,242
18	,383E-03	8	21	,430E+02	1	21	1	0	,146E+00	1,400	0,000	113	,242
19	,341E-03	8	21	,384E+02	1	21	1	0	,131E+00	1,400	0,000	113	,243
20	,304E-03	8	21	,349E+02	1	21	1	0	,118E+00	1,400	0,000	114	,247
21	,271E-03	8	21	,316E+02	1	21	1	0	,107E+00	1,400	0,000	114	,246
22	,242E-03	9	21	,285E+02	1	21	1	0	,959E-01	1,400	0,000	114	,240
23	,221E-03	9	21	,257E+02	1	21	1	0	,862E-01	1,400	0,000	114	,240
24	,200E-03	9	21	,232E+02	1	21	1	0	,777E-01	1,400	0,000	114	,241
25	,182E-03	9	21	,210E+02	1	21	1		,704E-01	1,400	0,000	114	,244

----DID NOT CONVERGE IN 25 CYCLES,---- RMAX= ,21E+02, COVR= ,25E-02

CPU SECONDS= 6,10 FOR 25 ITERATIONS, NHALF=0

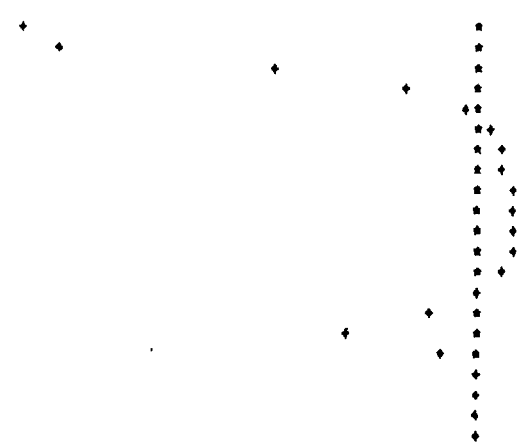
I	BB	XB	YB	CP	M
1	0,000	0,000	0,000	1,27261	0,00000
2	,006	,002	,006	1,18736	,21394
3	,026	,017	,018	,58442	,65988
4	,071	,058	,034	,20426	,87617
5	,149	,135	,050	,03982	,97140
6	,267	,252	,066	-,02344	1,00902
7	,424	,408	,081	-,05237	1,02646
8	,616	,600	,092	-,06930	1,03675
9	,835	,819	,098	-,07896	1,04265
10	1,066	1,050	,100	-,08511	1,04641
11	1,293	1,277	,096	-,09114	1,05011
12	1,500	1,484	,087	-,09826	1,05449
13	1,674	1,658	,075	-,06614	1,03483
14	1,807	1,791	,061	,00611	,99136
15	1,906	1,890	,046	,12420	,92216
16	1,996	1,980	,020	,38754	,77228
17	2,121	2,105	,020	,11500	,92748
18	2,329	2,313	,020	,02152	,98221
19	2,746	2,730	,020	,00480	,99214
20	3,996	3,980	,020	,00046	,99472
21	**********	**********	,020	0,00000	,99500

DRAG COEFFICIENT BY TRAPEZOIDAL INTEGRATION= ,03738

DRAG COEFFICIENT BY SIMPSON INTEGRATION= -,03446

PLOT OF CP AT UNEQUAL INCREMENTS

I	XB	YB	CP
1	0,000	0,000	1,2726
2	,002	,006	1,1874
3	,017	,018	,5844
4	,058	,034	,2043
5	,135	,050	,0396
6	,252	,066	-,0234
7	,408	,081	-,0524
8	,600	,092	-,0693
9	,819	,098	-,0790
10	1,050	,100	-,0851
11	1,277	,096	-,0911
12	1,484	,087	-,0983
13	1,658	,075	-,0661
14	1,791	,061	,0061
15	1,890	,046	,1242
16	1,980	,020	,3875
17	2,105	,020	,1150
18	2,313	,020	,0215
19	2,730	,020	,0048
20	3,980	,020	,0005
21	***********	,020	0,0000



MACH NO, CHART

	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
1//	0	88	91	94	95	96	97	98	98	98	99	99	99	99	99	99	99	99	99	99	99
2//	21	65	90	93	95	96	97	97	98	98	98	99	99	99	99	99	99	99	99	99	99
3//	65	80	86	89	92	93	95	96	97	97	98	98	98	99	99	99	99	99	99	99	99
4//	87	87	89	90	91	92	93	94	95	96	96	97	98	98	98	99	99	99	99	99	99
5//	97	96	95	95	95	95	95	96	96	96	96	97	97	98	98	98	99	99	99	99	99
6//	100	100	99	99	99	98	98	98	98	98	98	98	98	98	98	98	99	99	99	99	99
7//	102	102	101	101	101	100	100	100	100	99	99	99	99	99	99	99	99	99	99	99	99
8//	103	103	102	102	102	102	101	101	101	101	100	100	100	100	99	99	99	99	99	99	99
9//	104	103	103	103	103	102	102	102	102	101	101	101	101	100	100	100	99	99	99	99	99
10//	104	104	104	103	103	103	103	102	102	102	102	102	101	101	101	100	100	100	99	99	99
11//	105	104	104	104	103	103	103	103	103	102	102	102	102	101	101	101	100	100	99	99	99
12//	105	105	104	104	104	104	103	103	103	103	102	102	102	102	101	101	101	100	99	99	99
13//	103	102	102	102	102	101	101	101	101	101	101	101	100	100	100	100	100	100	99	99	99
14//	99	98	98	97	97	97	97	97	97	97	97	98	98	98	98	98	99	99	99	99	99
15//	92	92	92	93	94	94	95	95	96	96	96	97	97	98	98	98	99	99	99	99	99
16//	78	84	87	89	91	92	93	94	95	95	96	97	97	97	98	98	99	99	99	99	99
17//	92	93	94	95	95	96	96	96	97	97	97	97	98	98	98	98	99	99	99	99	99
18//	98	98	98	98	98	98	98	98	98	98	98	98	98	98	98	99	99	99	99	99	99
19//	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
20//	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
21//	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

N= 36

XS(K), K=1,....,N

,224E+00	,176E+01	,244E+00	,175E+01	,269E+00	,174E+01	,292E+00	,173E+01
,311E+00	,172E+01	,326E+00	,172E+01	,340E+00	,172E+01	,353E+00	,171E+01
,367E+00	,171E+01	,387E+00	,171E+01	,414E+00	,171E+01	,441E+00	,170E+01
,469E+00	,170E+01	,502E+00	,170E+01	,549E+00	,170E+01	,631E+00	,171E+01
,759E+00	,171E+01	,925E+00	,169E+01				

YS(K), K=1,....,N

,625E-01	,640E-01	,922E-01	,927E-01	,125E+00	,124E+00	,163E+00	,160E+00
,206E+00	,202E+00	,256E+00	,250E+00	,314E+00	,308E+00	,383E+00	,376E+00
,467E+00	,458E+00	,570E+00	,560E+00	,697E+00	,686E+00	,860E+00	,847E+00
,107E+01	,106E+01	,136E+01	,134E+01	,176E+01	,174E+01	,237E+01	,234E+01
,334E+01	,331E+01	,510E+01	,506E+01				

CPU SECONDS TO COMPUTE AND PLOT CP AND MCHART= ,383

SPHERE/15-DEG CONE/CYLINDER/15-DEG FLARE

INPUT COORDINATES

I	X	Y
1	0.000000	0.000000
2	.001637	.040430
3	.007003	.083390
4	.017510	.131200
5	.035640	.185400
6	.065090	.246700
7	.110700	.313800
8	.178200	.382700
9	.272700	.445300
10	.396900	.490000
11	.543800	.529400
12	.710600	.574100
13	.897600	.624200
14	1.105000	.679700
15	1.332000	.740600
16	1.578000	.606600
17	1.842000	.877300
18	2.123000	.952400
19	2.421000	1.000000
20	2.738000	1.000000
21	3.065000	1.000000
22	3.398000	1.000000
23	3.735000	1.000000
24	4.072000	1.000000
25	4.406000	1.028000
26	4.734000	1.116000
27	5.053000	1.202000
28	5.361000	1.284000
29	5.655000	1.363000
30	5.934000	1.438000
31	6.198000	1.509000
32	6.448000	1.575000
33	6.684000	1.639000
34	6.910000	1.699000
35	7.133000	1.759000
36	7.357000	1.819000
37	7.594000	1.683000
38	7.856000	1.953000
39	8.156000	2.033000
40	8.511000	2.074000

DYDXN= 999.0000
DYDXT= .2680
YMAX= 2.0000
XREF= 8.5000

IMAX= 21
JMAX= 21
MIT= 25
MHALF=0
KLOSE=0
NPLOT=1
RF1=1.400
COVERG= .10E+01
QF3= 0.
DNDYO= .286E+01
ALF=1.30
DXIDXO= .200E+01
XM= .800E+01
CXM= .750E+00
DXIDXM= .150E+02
GAM=1.40
AMINF= .8000

I	B	X	Y	THET	THETB	AK	F
1	0.	0.	0.	.9000E+02	.9000E+02	.1999E+01	.5000E+00
2	.1074E+00	.1148E-01	.1065E+00	.7771E+02	.7771E+02	.2000E+01	.4098E+00
3	.2580E+00	.6508E-01	.2467E+00	.6046E+02	.6046E+02	.1994E+01	.2693E+00
4	.4903E+00	.2217E+00	.4156E+00	.3393E+02	.3393E+02	.2120E+01	.1761E+00
5	.8336E+00	.5437E+00	.5294E+00	.1481E+02	.1481E+02	.8703E-01	.1233E+00
6	.1305E+01	.9986E+00	.6512E+00	.1498E+02	.1498E+02	.2152E-02	.9319E-01
7	.1904E+01	.1578E+01	.8066E+00	.1489E+02	.1489E+02	.2955E-01	.7568E-01
8	.2618E+01	.2270E+01	.9821E+00	.9323E+01	.9323E+01	.5403E+00	.6563E-01
9	.3416E+01	.3066E+01	.1000E+01	0.	.4194E+00	.8329E-01	.6055E-01
10	.4254E+01	.3904E+01	.9991E+00	0.	-.1833E+00	-.6287E-01	.5933E-01
11	.5084E+01	.4735E+01	.1116E+01	0.	.1620E+02	.2172E+00	.6173E-01
12	.5860E+01	.5510E+01	.1324E+01	0.	.1505E+02	.1912E-03	.6798E-01
13	.6548E+01	.6199E+01	.1509E+01	0.	.1465E+02	.5006E-01	.7602E-01
14	.7148E+01	.6798E+01	.1669E+01	0.	.1481E+02	.8646E-02	.8835E-01
15	.7707E+01	.7357E+01	.1819E+01	0.	.1515E+02	-.5544E-01	.8731E-01
16	.8350E+01	.8000E+01	.1996E+01	0.	.1619E+02	.4308E+00	.6667E-01
17	.9287E+01	.8937E+01	.2074E+01	0.	0.	0.	.4267E-01
18	.1085E+02	.1050E+02	.2074E+01	0.	0.	0.	.2400E-01
19	.1397E+02	.1362E+02	.2074E+01	0.	0.	0.	.1067E-01
20	.2335E+02	.2300E+02	.2074E+01	0.	0.	0.	.2667E-02
21	.1000E+31	.1000E+31	.2074E+01	0.	0.	0.	.1000E-29

----- NORMAL COORD. STRETCH FOR ALF= 1.300 -----

J	AN	G	GH
1	0.	0.	.1386E-03
2	.1334E+03	.2772E-03	.8292E-03
3	.5131E+02	.1381E-02	.2466E-02
4	.2860E+02	.3552E-02	.5259E-02
5	.1852E+02	.6967E-02	.9374E-02
6	.1299E+02	.1178E-01	.1496E-01
7	.9567E+01	.1814E-01	.2216E-01
8	.7270E+01	.2619E-01	.3112E-01
9	.5642E+01	.3605E-01	.4196E-01
10	.4437E+01	.4786E-01	.5484E-01
11	.3518E+01	.6160E-01	.6968E-01
12	.2797E+01	.7797E-01	.8724E-01
13	.2220E+01	.9652E-01	.1071E+00
14	.1751E+01	.1176E+00	.1295E+00
15	.1363E+01	.1414E+00	.1547E+00
16	.1038E+01	.1680E+00	.1828E+00
17	.7637E+00	.1976E+00	.2141E+00
18	.5294E+00	.2305E+00	.2486E+00
19	.3277E+00	.2667E+00	.2866E+00
20	.1527E+00	.3065E+00	.3282E+00
21	0.	.3500E+00	.3738E+00

CPU SECONDS FOR BODY GEOMETRY COMPUTATIONS= .110

IT	DPMAX	ID	JD	RMAX	IR	JR	ISUB	ISUP	RAVG	RF1	RF3	NS	SEC/CYC
1	,102E+00	4	21	,128E+02	2	21	1	0	,209E+00	1.400	0,000	0	,204
2	,690E-01	2	21	,248E+02	1	21	1	0	,206E+00	1.400	0,000	0	,213
3	,607E-01	1	21	,204E+02	1	21	1	0	,197E+00	1.400	0,000	0	,214
4	,594E-01	1	20	,146E+02	1	21	1	0	,188E+00	1.400	0,000	0	,211
5	,462E-01	1	21	,152E+02	1	21	1	0	,169E+00	1.400	0,000	0	,210
6	,345E-01	1	21	,116E+02	1	21	1	0	,132E+00	1.400	0,000	0	,213
7	,283E-01	14	21	,717E+01	1	21	1	0	,987E-01	1.400	0,000	0	,211
8	,256E-01	14	21	,485E+01	1	21	1	0	,780E-01	1.400	0,000	0	,216
9	,232E-01	14	21	,391E+01	1	21	1	0	,668E-01	1.400	0,000	0	,212
10	,211E-01	14	21	,344E+01	1	21	1	0	,603E-01	1.400	0,000	0	,210
11	,193E-01	14	21	,312E+01	1	21	1	0	,557E-01	1.400	0,000	0	,214
12	,176E-01	14	21	,286E+01	1	21	1	0	,517E-01	1.400	0,000	0	,214
13	,162E-01	14	21	,260E+01	1	21	1	0	,479E-01	1.400	0,000	0	,219
14	,148E-01	14	21	,235E+01	1	21	1	0	,441E-01	1.400	0,000	0	,214
15	,137E-01	14	21	,212E+01	1	21	1	0	,404E-01	1.400	0,000	0	,217
16	,126E-01	14	21	,192E+01	1	21	1	0	,371E-01	1.400	0,000	0	,214
17	,117E-01	14	21	,174E+01	1	21	1	0	,342E-01	1.400	0,000	0	,210
18	,108E-01	14	21	,159E+01	1	21	1	0	,318E-01	1.400	0,000	0	,211
19	,100E-01	14	21	,147E+01	1	21	1	0	,297E-01	1.400	0,000	0	,217
20	,934E-02	14	21	,137E+01	1	21	1	0	,280E-01	1.400	0,000	0	,218
21	,870E-02	14	21	,129E+01	1	21	1	0	,265E-01	1.400	0,000	0	,217
22	,812E-02	14	21	,123E+01	1	21	1	0	,251E-01	1.400	0,000	0	,216
23	,759E-02	14	21	,117E+01	1	21	1	0	,238E-01	1.400	0,000	0	,216
24	,710E-02	14	21	,112E+01	1	21	1	0	,225E-01	1.400	0,000	0	,216
25	,665E-02	14	21	,106E+01	1	21	1	0	,213E-01	1.400	0,000	0	,211

----DID NOT CONVERGE IN 25 CYCLES,---- RMAX= ,11E+01, COVR= ,25E-02

CPU SECONDS= 5,42 FOR 25 ITERATIONS, NHALF=0

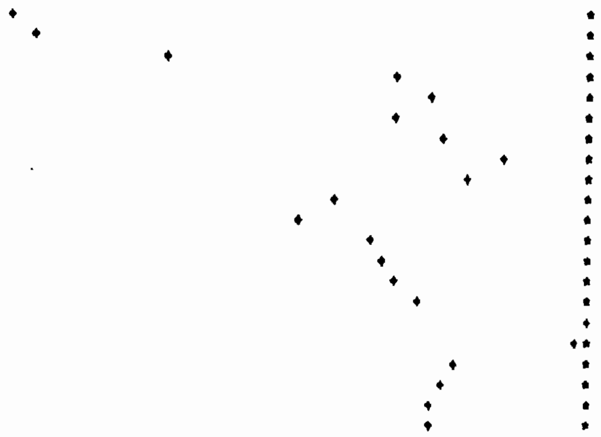
I	SB	XB	YB	CP	M
1	0,000	0,000	0,000	1,17040	0,00000
2	,107	,011	,107	1,09484	,17941
3	,258	,065	,247	,72935	,44960
4	,490	,222	,416	,09728	,75609
5	,834	,544	,529	-,00286	,80129
6	1,305	,999	,651	,10179	,75405
7	1,904	1,578	,807	-,02264	,81022
8	2,618	2,270	,982	-,19547	,88861
9	3,416	3,066	1,000	-,08256	,83729
10	4,254	3,904	,999	,26338	,68051
11	5,084	4,735	1,116	,36698	,63244
12	5,860	5,510	1,324	,18223	,71760
13	6,548	6,199	1,509	,12378	,74411
14	7,148	6,798	1,669	,08965	,75954
15	7,707	7,357	1,819	,03537	,78404
16	8,350	8,000	1,996	-,42017	,99311
17	9,287	8,937	2,074	-,38472	,97633
18	10,850	10,500	2,074	-,06093	,82751
19	13,975	13,625	2,074	-,01423	,80642
20	23,350	23,000	2,074	-,00138	,80062
21**********************			2,074	0,00000	,80000

DRAG COEFFICIENT BY TRAPEZOIDAL INTEGRATION= ,04600

DRAG COEFFICIENT BY SIMPSON INTEGRATION= -,01718

PLOT OF CP AT UNEQUAL INCREMENTS

I	XB	YB	CP
1	0,000	0,000	1,1704
2	,011	,107	1,0948
3	,065	,247	,7293
4	,222	,416	,0973
5	,544	,529	-,0029
6	,999	,651	,1018
7	1,578	,807	-,0226
8	2,270	,982	-,1955
9	3,066	1,000	-,0826
10	3,904	,999	,2634
11	4,735	1,116	,3670
12	5,510	1,324	,1822
13	6,199	1,509	,1238
14	6,798	1,669	,0896
15	7,357	1,819	,0354
16	8,000	1,996	-,4202
17	8,937	2,074	-,3847
18	10,500	2,074	-,0609
19	13,625	2,074	-,0142
20	23,000	2,074	-,0014
21	***********	2,074	0,0000



MACH NO. CHART

	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
1//	0	36	53	62	68	72	74	76	77	78	78	79	79	79	79	79	79	79	79	79	80
2//	17	38	53	63	68	72	74	76	77	78	78	79	79	79	79	79	79	79	79	79	80
3//	44	48	57	64	68	72	74	75	77	77	78	78	79	79	79	79	79	79	79	79	80
4//	75	68	68	70	72	73	75	76	77	78	78	79	79	79	79	79	79	79	79	79	80
5//	80	76	74	74	74	74	74	75	76	76	77	78	78	79	79	79	79	79	80	80	80
6//	75	75	75	75	76	76	76	77	77	77	78	78	78	79	79	79	79	79	79	80	80
7//	81	80	79	79	78	78	78	78	78	78	78	78	78	79	79	79	79	79	79	80	80
8//	86	86	84	82	81	80	79	79	79	79	79	79	79	79	79	79	79	79	79	79	80
9//	83	82	81	80	79	79	78	78	78	78	78	78	79	79	79	80	80	80	80	80	80
10//	68	69	70	72	73	74	75	76	77	76	78	79	79	79	79	80	80	80	80	80	80
11//	63	65	66	68	70	71	73	74	75	76	77	78	79	79	79	79	80	80	80	80	80
12//	71	72	72	72	73	74	75	75	76	77	78	78	79	79	79	79	80	80	80	80	80
13//	74	74	74	75	75	76	76	77	78	78	79	79	79	79	79	79	79	79	80	80	80
14//	75	76	76	77	77	78	78	79	79	80	80	80	80	80	79	79	79	79	79	79	80
15//	78	79	80	81	81	82	82	82	82	81	81	80	80	80	80	79	79	79	79	79	80
16//	99	97	95	92	90	89	87	85	84	82	81	81	80	80	79	79	79	79	79	79	80
17//	[illegible]	95	93	92	90	88	87	85	84	83	82	81	81	80	80	80	79	79	79	79	80
18//	82	82	82	82	82	82	82	82	82	81	81	81	80	80	80	80	80	79	79	79	80
19//	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	79	79	80
20//	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80
21//	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80	80
	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

CPU SECONDS TO COMPUTE AND PLOT CP AND MCHART= ,279

REFERENCES

1. South, Jerry C. Jr.; and Jameson, Antony: Relaxation Solutions for Inviscid Axisymmetric Transonic Flow Over Blunt or Pointed Bodies. AIAA Computational Fluid Dynamics Conference (Palm Springs, Calif., July 1973, pp. 8-17).

2. Jameson, Antony: Iterative Solution of Transonic Flows Over Airfoils and Wings, Including Flows at Mach 1. Commun. Pure & Appl. Math., vol. 27, no. 3, May 1974, pp. 283-309.

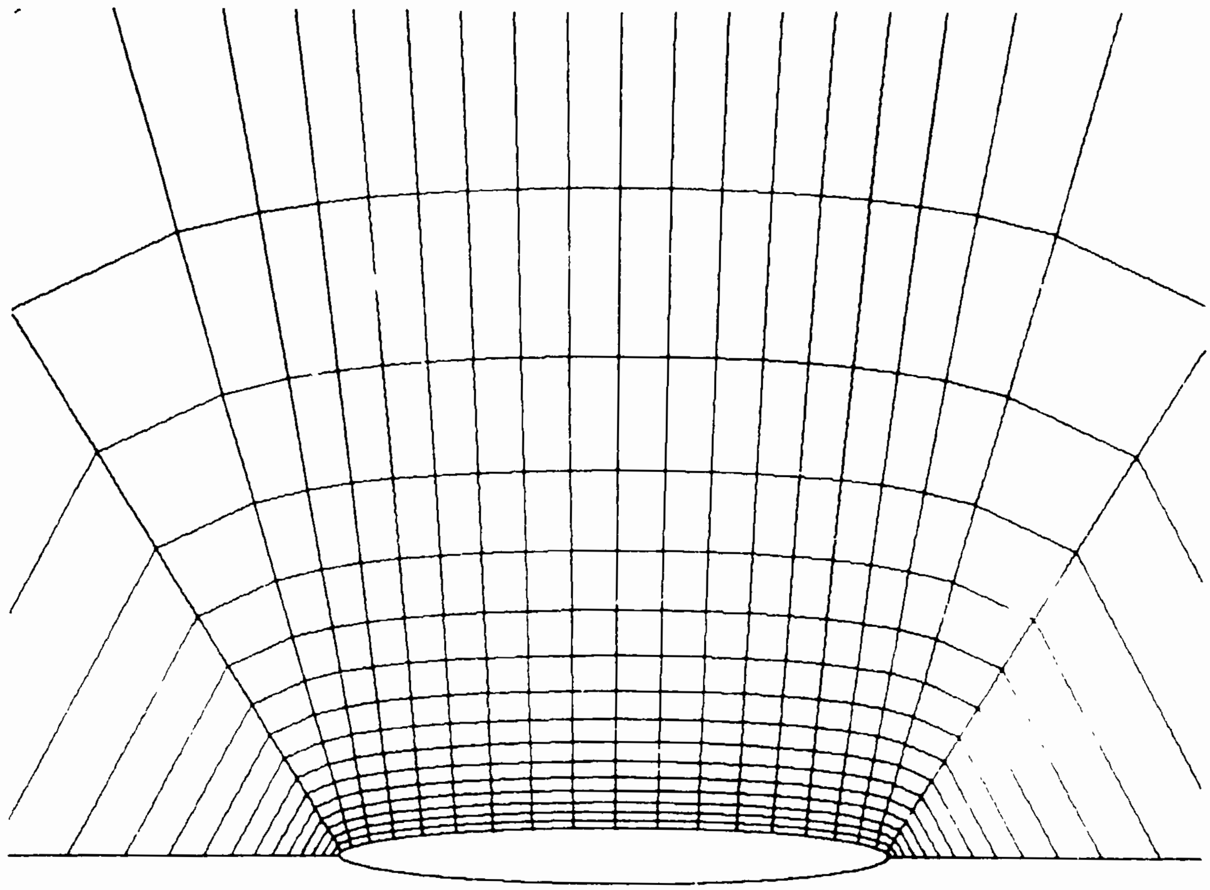


Figure 1. - Frame 1 of plotted output.

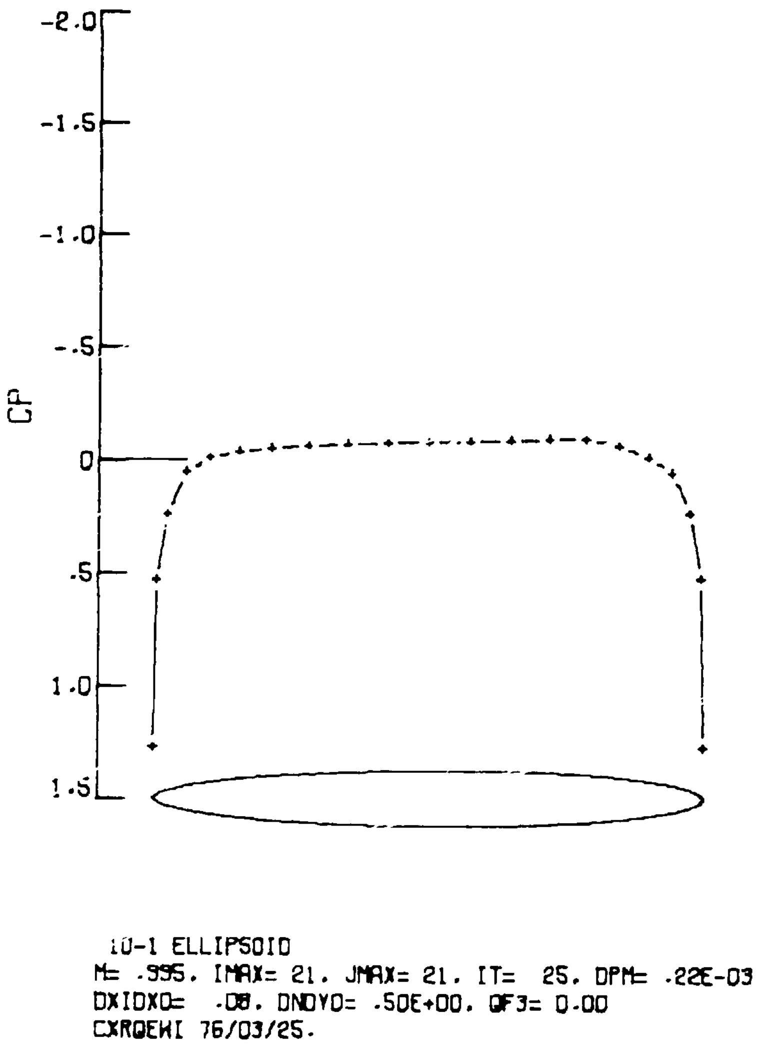
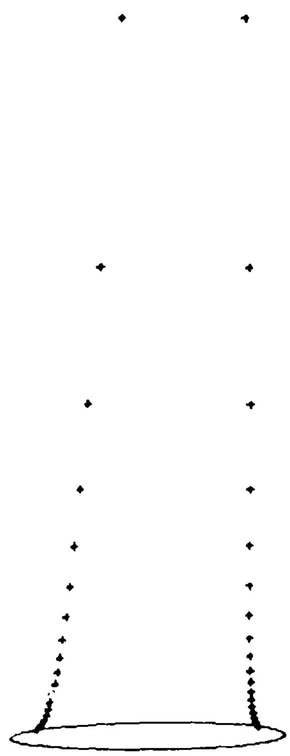


Figure 2. - Frame 2 of plotted output.



ORIGINAL PAGE IS
OF POOR QUALITY

Figure 3. - Frame 3 of plotted output.

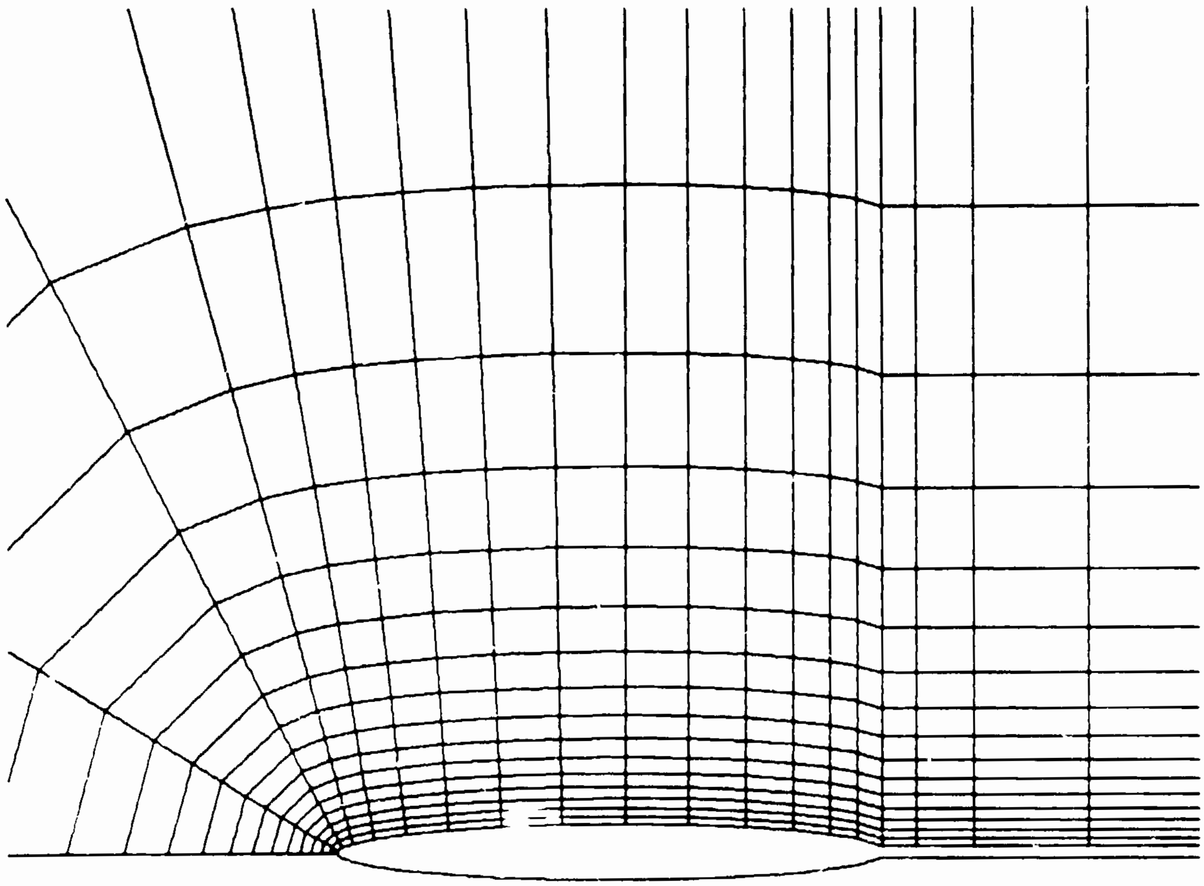


Figure 4. - Frame 4 of plotted output.

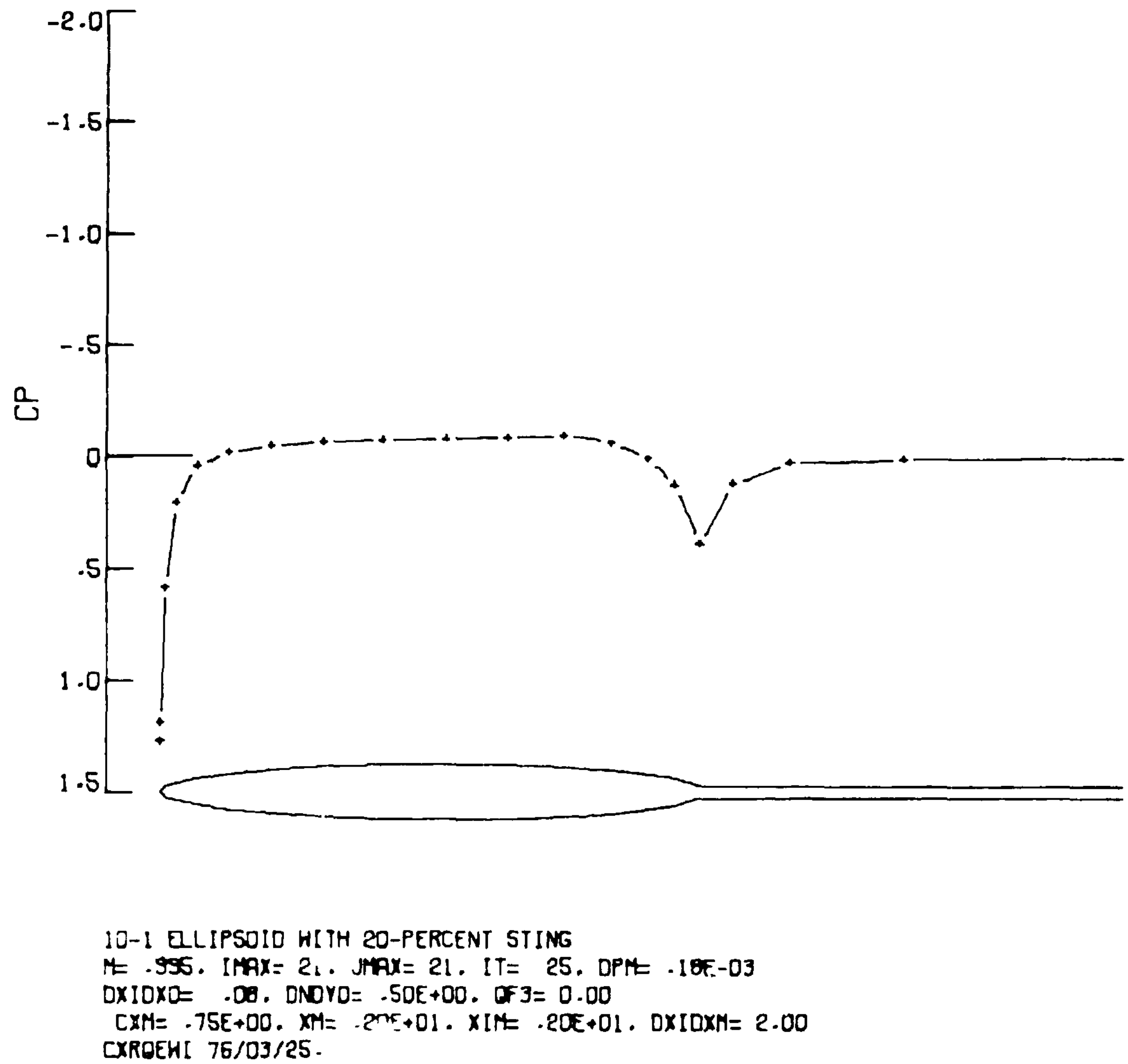


Figure 5. - Frame 5 of plotted output.

ORIGINAL PAGE IS
OF POOR QUALITY

Figure 6. - Frame 6 of plotted output.

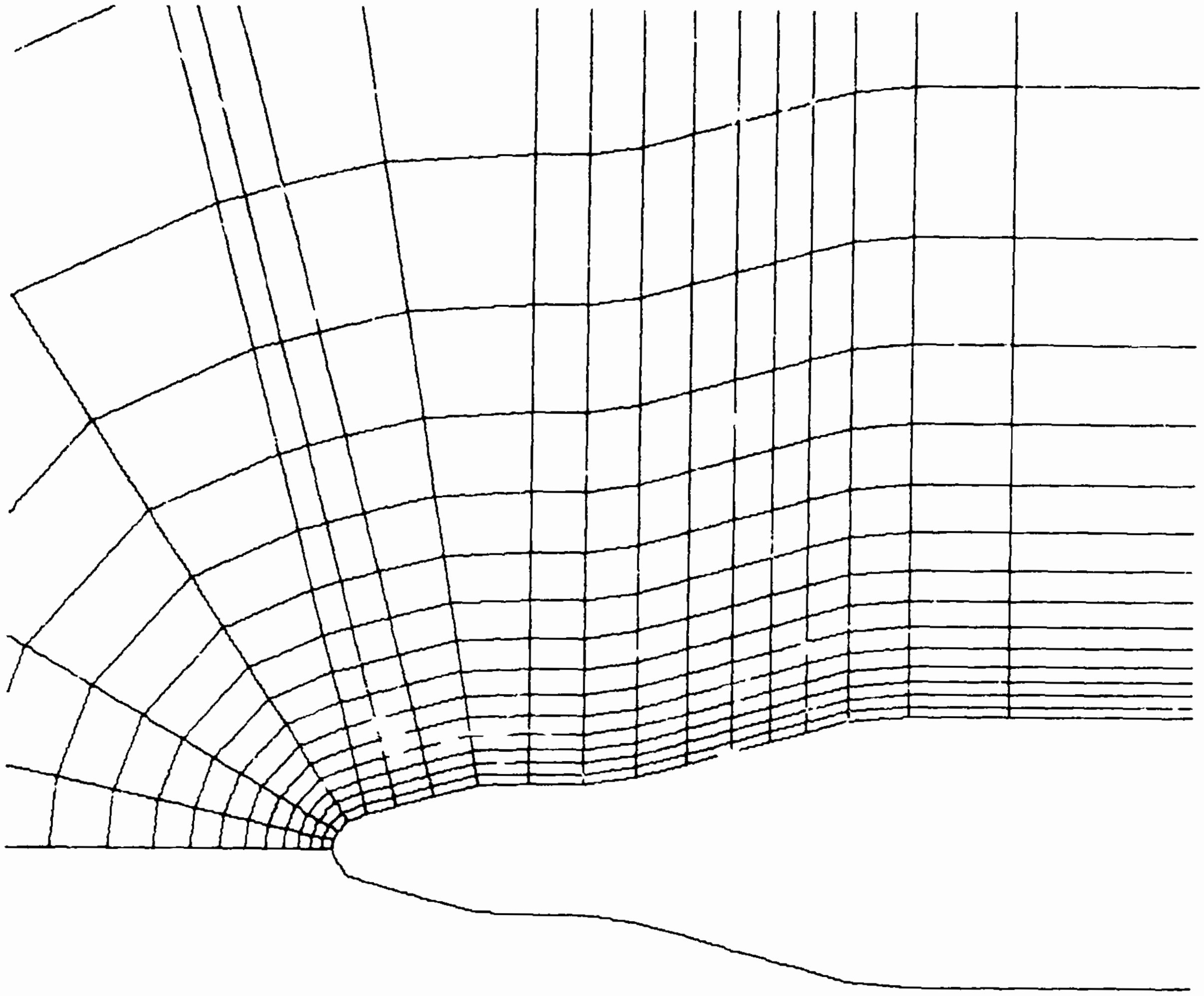


Figure 7. - Frame 7 of plotted output.

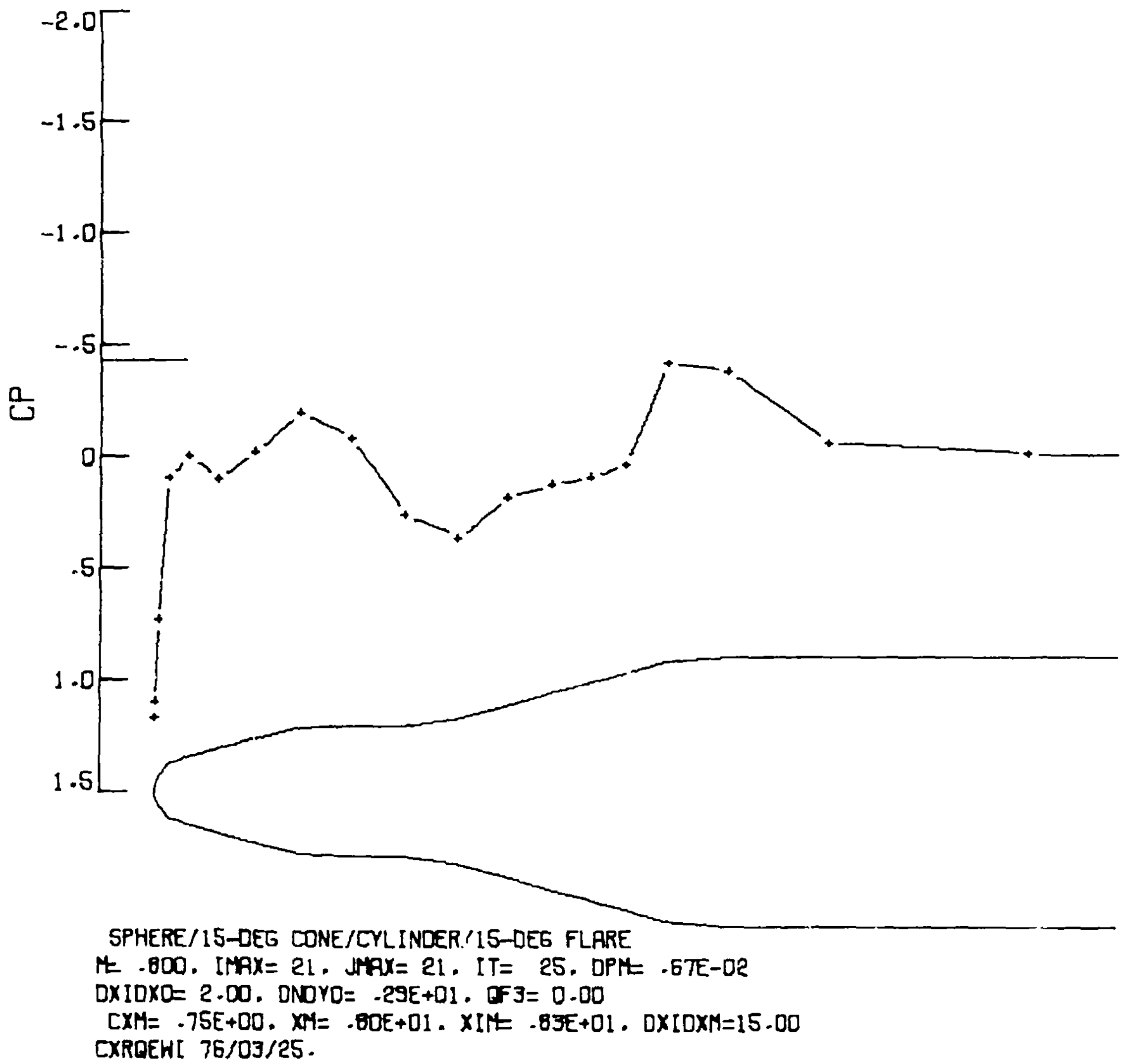


Figure 8. - Frame 8 of plotted output.