NASA TECHNICAL MEMORANDUM

NASA TM X-1598

NASA TM X-1598

PLOT3D — A PACKAGE OF FORTRAN SUBPROGRAMS TO DRAW THREE-DIMENSIONAL SURFACES

by R. Bruce Canright, Jr., and Paul Swigert

Lewis Research Center

Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JUNE 1968

NASA TM X-1598

PLOT3D - A PACKAGE OF FORTRAN SUBPROGRAMS TO DRAW THREE-DIMENSIONAL SURFACES

By R. Bruce Canright, Jr., and Paul Swigert

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

ABSTRACT

PLOT3D is a package of FORTRAN subprograms to draw three-dimensional surfaces of the form $z = f(x, y)$. The function f and the bounding values for x and y are the input to PLOT3D. The surface thus defined by PLOT3D may be drawn after arbitrary rotations. Output is by off-line incremental plotter or on-line microfilm recorder. PLOT3D is completely described along with its limitations, possible future modifications, and potential uses. Sample output and listings of the subprograms are included. Written entirely in FORTRAN IV, PLOT3D is readily adaptable to other hardware.

STAR Category 08

views are not perspective. PLOT3D is designed primarily to portray functions of two independent variables. The functions must be piecewise continuous and bounded.

This report contains a complete description of the PLOT3D package. In addition, potential applications are presented; examples of PLOT3D output are included; and the limitations of PLOT3D are discussed.

PLOT3D is implemented for both the IBM 7044/7094 Direct Couple System - California Computer Products Digital Incremental Plotter, controller model 780, printer model 765; and the IBM 360/67 - Control Data Corporation microfilm system model 280 (referred to at Lewis as the DD80). The subprograms for both are explained, and there is a sample user's program. Finally, possible future modifications are mentioned.

POSSIBLE APPLICATIONS FOR PLOT3D

It can be extremely difficult to get a "feel" for the behavior of complicated functions or the shape of empirical surfaces from tabulated values or conventional contour plots. Since plotting such surfaces is PLOT3D's purpose, PLOT3D may be of use anywhere these surfaces arise.

In engineering and physics, functions of two variables arise often, both experimentally and theoretically. PLOT3D can portray their behavior. For example, the phase diagrams and equations of state of thermodynamics can be studied. In mechanics, for example, energy surfaces and potential wells can be drawn (see fig. 1). Data points from experiments can be curve-fitted and drawn in three dimensions (see fig. 2, also ref. 8).

Partial differential equations in two independent variables occur frequently - for example, in heat flow, electricity and magnetism, fluid dynamics, and quantum mechanics. Maps of their analytic and numerical solutions are easy to make with PLOT3D (see fig. 3).

In numerical analysis and analytic geometry, problems arise which can be solved with the aid of PLOT3D: The behavior of functions near singularities can be pictured (see figs. 4(a), (b), and (c)); approximate roots of equations can be located; and the form of surfaces after rotations, contraction, and deformations can be pictured. Numerical solutions to equations of any sort can be curve-fitted and then mapped by PLOT3D.

LIMITATIONS OF PLOT3D

PLOT3D is designed to draw only functions; that is, for every point in the x and y ranges there must be a unique z value. It cannot draw, for example, a sphere.

PLOT3D - A PACKAGE OF FORTRAN SUBPROGRAMS TO DRAW THREE-DIMENSIONAL SURFACES

by R. Bruce Canright, Jr., and Paul Swigert

Lewis Research Center

SUMMARY

PLOT3D is a package of FORTRAN subprograms to draw three-dimensional surfaces of the form $z = f(x, y)$. The function f and the bounding values for x and y are the input to PLOT3D. The surface thus defined by PLOT3D may be drawn after arbitrary rotations. Output is by off-line incremental plotter or on-line microfilm recorder.

Plot3D is completely described along with its limitations, possible future modifications, and potential uses. Sample output and listings of the subprograms are included. Since PLOT3D is written entirely in FORTRAN IV, it is readily adaptable to other hardware.

INTRODUCTION

Several packages for plotting three-dimensional surfaces have been developed elsewhere. Each package has its strengths and limitations. Some of these programs are intended primarily to show the spatial relations among solids. This problem was perhaps studied first by Roberts, who considered convex polyhedra (ref. 1). Appel has developed powerful methods for drawing solids bounded by planes (ref. 2). Weiss has dealt with surfaces which are plane or quadric (ref. 3).

Other packages draw surfaces made up of data points. Perspective drawings can be made from a given viewpoint (refs. 4 and 5). Branstetter has designed a program to draw contour maps (ref. 6), and there are similar programs available commercially. Nelson has developed routines which, within certain restrictions, will plot both arrays of data and functions of two variables (ref. 7). All packages are machine dependent to some extent.

PLOT3D will plot any function of the form $z = f(x, y)$. It uses a given recipe to generate views of the surface after arbitrary rotations about the three spatial axes. The

The functions must be in rectangular coordinates. Also, the function must be expressed explicitly in the form $z = f(x, y)$. Where such a form is inconvenient or impossible, this restriction can sometimes be overcome by curve fitting methods.

Every line lying on the surface is drawn, not just those which are "visible" or "in front." In a sense, the views are projections of wire frame models onto a plane. They are not like wood models.

The output views may be difficult to study quantitatively. It requires careful choice of the input parameters to read values off the surface plots.

DESCRIPTION OF PLOT3D PACKAGE

The method used by the routines is simple. The given x and y ranges are divided to form a uniform rectangular grid. The values of the supplied function at the grid points (x, y) are calculated and stored; this defines the surface.

The surface is portrayed by connecting successive (y, z) points with straight-line segments for each x value on the grid and, in turn, connecting successive (x, z) points for each fixed y value on the grid. These lines are then projected by parallel projection onto the fixed yz-plane for plotting.

The function is evaluated only once, before the first view. Views of the surface are made after rotating it about the fixed spatial axes - first around the fixed x-axis through angle α (see fig. 5(a)), then around y through β (fig. 5(b)), and finally around z through γ(fig. 5(c), see also ref. 10). For this order it is seen that the matrix which performs the desired rotation is simply the product CBA where A, B, and C are the matrices defined in figure 5. The grid points and their function values are transformed and redrawn. Note that the three axes of rotation are fixed and that the order of rotations is crucial, because matrix multiplication is not in general commutative: for example, the rotation ABC does not give the same view as CBA.

Input parameters allow considerable flexibility in drawing the surface. The x and y ranges are determined by supplying maximum and minimum x and y values. The number of grid lines in each direction is supplied, as well as the number of points on each grid line. The more grid lines there are, the finer the mesh is. This fine mesh allows shading of areas with small surface gradients. Because straight-line segments are drawn between the points, the more points that are used, the better the representation of curvature is. The number of points is limited by the computer storage space and by the computation time. Finally, the surface may be rotated about the three fixed axes, as discussed earlier.

A general flow chart of the subprograms which make the drawings is given in figure 6. As the left side of this figure shows, SUBROUTINE PLOT3D sets up the rectangular grid and uses the supplied function to get the z values. At this point the figure axes coincide with the fixed axes; that is, the figure yz plane coincides with the fixed yz plotting plane.

ENTRY ROTATE to PLOT3D calls subroutines which draw the surface after arbitrary rotations about the three fixed axes, as the right side of figure 6 shows. A description of the routines called by PLOT3D and ROTATE follows.

SUBROUTINE SCALE, called once for each surface to be drawn, locates the minimum and maximum z values. SCALE then scales x, y, and z to fit into a cube 3.5 plotting units on a side, with the plotting plane defined as a square with 10 plotting units on a side.

The variables x, y, and z may be scaled in one of two ways. Each variable may be scaled according to the maximum magnitude it takes on, or all three variables may be scaled according to the largest of these three maxima. The first way of scaling fits the surface into a cube and may distort it considerably. The second way of scaling preserves the original relative magnitudes, but may shrink one or two axes so small as to be invisible. The bounding values are placed in COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX.

SUBROUNTINE AXIS, called once for each surface view, sets up and draws the three figure axes. These axes help to show the orientation of the surface with respect to the fixed spatial axes (see figs. 5 and 7). AXIS calls PHI to rotate these figure axes.

SUBROUTINE DRAW, called once for each view plotted, draws the line segments connecting (y, z) grid points. DRAW, like AXIS and WRITE, uses plotting routines which are machine dependent.

SUBROUTINE WRITE, called once for each view plotted, labels the figure with the bounding values from COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX. This subroutine also labels the figure with the accumulated angles α, β, and γ, and with an alphanumeric array supplied through COMMON/LABEL/ as explained in the section USING PLOT3D. The array in COMMON/LABEL/ is initially filled with blanks in a BLOCK DATA subprogram.

SUBROUTINE TRNMAT sets up the rotation matrix M (M = CBA where the matrices A, B, C are defined according to fig. 5) by using the supplied angles of rotation α, β, and γ. TRNMAT is called once for every rotated view, that is, every call to ROTATE. The rotation matrix is placed in COMMON/MATRIX/TMAT (3, 3).

SUBROUTINE PHI rotates the surface curves and figure axes by multiplying the position vector of each point by the rotation matrix in COMMON/MATRIX/TMAT (3, 3). It is called three times for each rotated view (see fig. 6).

The output devices and subprograms are dependent on the computer used to calculate and store the points to be plotted. PLOT3D has been written for both the IBM 7044/94 DCS and the IBM 360/67. On the 7044/94, PLOT3D uses the CalComp programs

SYMBOL, NUMBER, and PLT770 (ref. 8). Output is written onto tape which is then used by the off-line CalComp plotter. On the 360/67, PLOT3D uses the plotting routine LRCURV and several associated routines developed at Lewis for the on-line microfilm recorder. Presently, a positive film is produced, which gives white-on-black prints.

Subroutine listings for the IBM 7044/94 appear in appendix A, while listings of the subroutines for the IBM 360/67 are given in appendix B. The functions of the plotting programs used have been explained in comments cards so that similar routines at other facilities might be substituted.

USING PLOT3D

The first call must be CALL PLOT3D(XMIN, XMAX, YMIN, YMAX, S, NXPTS, NYPLS, NXPLS, NYPTS, FUNC, CUBE)

where

XMIN	minimum x value
XMAX	maximum x value
YMIN	minimum y value
YMAX	maximum y value
S	name of the array used by PLOT3D to hold all the coordinates of points to be plotted (It must be dimensioned at least 3*(NXPTS*NYPLS+NYPTS*NXPLS).)
NXPTS	number of points in each plane y = constant (These are connected by lines to represent a curve on the surface.)
NYPLS	number of planes y = constant
NXPLS	number of planes x = constant
NYPTS	number of points lying in each of these planes
FUNC	name of the function z = f(x, y) to be plotted (It must appear in an EXTERNAL statement in the user's calling program. The user must supply a function subprogram of the form FUNCTION FUNC(x, y).)
CUBE	LOGICAL variable (When CUBE = .TRUE., the three dimensions are scaled independently in order to fit the surface into a cube. When CUBE = .FALSE., the surface is not fit into a cube. See the discussion on SUBROUTINE SCALE, p. 4.)

The preceding parameters, except for S, FUNC, and CUBE are illustrated in figure 7. Views of the surface may now be made by the call CALL ROTATE (A, B, C, DRAWME) where

A	angle of rotation about the fixed x axis, deg
B	angle of rotation about the fixed y axis, deg
C	angle of rotation about the fixed z axis, deg
DRAWME	LOGICAL variable set to .TRUE. if the particular view is to be plotted, or set to .FALSE. if this view is not to be plotted

Positive angles are counterclockwise rotations, while negative angles are clockwise rotations. The surface is not returned to its original position after rotation. Labels may be placed below the views by placing an alphanumeric array in the common block/LABEL/. For the 7044/94, COMMON/LABEL/LAB(12), this array may contain up to 12 words of 6 characters each; for the 360/67, COMMON/LABEL/LAB(15), the array may contain up to 15 words of 4 characters each

A sample calling program with function subprogram for the 7094/44 follows. This program produced figures 4(a), (b), and (c). Note that for use on the 360/67 only the array LAB is changed.

```
      EXTERNAL POLES
      LOGICAL CUBE,DRAWME
      DIMENSION LABA(12),S(6000)
      COMMON/LABEL/ LAB(12)
      DATA LABA /2*6H      ,6HW = CA, 6HBS(1/Z, 6H + 1/(, 6HZ-1)**,
     * 6H2) . Z, 6H = X +, 6H I*Y ., 3*6H         /
      CUBE = .TRUE.
      DO 1 I=1,12
    1 LAB(I) = LABA(I)
      CALL PLOT3D(-3.0,3.0,-3.0,3.0,S,51,31,7,51,POLES,CUBE)
      DRAWME = .FALSE.
      CALL ROTATE(0.0,0.0,45.0,DRAWME)
      DRAWME = .TRUE.
      DO 2 I=1,3
      CALL ROTATE(0.0,20.0,0.0,DRAWME)
    2 CONTINUE
      STOP
      END
```

```
      FUNCTION POLES(X,Y)
C        FUNCTION WITH TWO POLES OF DIFFERENT ORDER
      COMPLEX Z
      IF(Y.NE.0.0) GO TO 1
      IF(X.NE.0.0 .AND. X.NE.1.0) GO TO 1
      FA = 20.
      GO TO 2
    1 Z = CMPLX(X,Y)
      FA = CABS(1.0/Z + 1.0/(Z-1.0)**2)
      IF(FA.GT.20.) FA = 20.
    2 POLES = FA
      RETURN
      END
```

The PLOT3D package requires about 950_{10} storage locations excluding the plot routines and the S array of coordinates. Execution time is roughly 1 to 3 seconds per view for the examples given in figures 1 to 4 and ~7500 points on the 360/67. There are no error returns in PLOT3D. For the 7044/94 computer, the execution time is less, but each view requires 1 to 2 minutes of plotter time.

CONCLUDING REMARKS

A package of FORTRAN subprogram has been designed to draw three-dimensional surfaces by means of either the 7044/7094 DCS and off-line CalComp plotter, or the 360/67 and on-line CDC microfilm recorder. The package is readily adaptable to other hardware. It is intended primarily for functions of two independent variables and is not much use in drawing, for example, complex three-dimensional solids.

The PLOT3D package can be of use in many areas of mathematics, physics, and engineering. It has already had successful applications here at Lewis.

PLOT3D could have new uses when new devices are available with the 360/67. With a scope and a conversational mode of operation, the user could draw, rotate, and alter many surfaces in a short time. Motion pictures have been made, which adds the fourth dimension - time.

Future modifications could increase the powers of PLOT3D. The projection of views onto the fixed plane could be made a point projection; that is, perspective views could be made. Functions in coordinate systems other than rectangular could be plotted. Parts of lines which are "hidden" could be determined and left off the view. Probably the only method to accomplish this would be a point-by-point visibility check, which is a time-consuming process.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, March 27, 1968,
129-02-01-05-22.

APPENDIX A

Routines for Direct Couple System

The following are listings of the subroutines contained in the PLOT3D package for the IBM 7044/94 DCS. These routines are used with an off-line CalComp digital incremental plotter. The standard CalComp plotting routines used are explained in the comments cards.

```
C
C
C         SUBROUTINE PLOT3D
C
C         PURPOSE
C           DRAW AND ROTATE THREE DIMENSIONAL SURFACES WITH THE CALCOMP
C           DIGITAL PLOTTER
C
C         USAGE
C           CALL PLOT3D(XMIN,XMAX,YMIN,YMAX,S,NXPTS,NYPLS,NXPLS,NYPTS,
C                F,CUBE)
C           CALL ROTATE(ALPHA,BETA,GAMMA,DRAWME)
C           COMMON /LABEL/LAB(12)
C
C         DESCRIPTION OF PARAMETERS
C           XMIN     -MINIMUM 'X' VALUE TO BE PLOTTED
C           XMAX     -MAXIMUM 'X' VALUE TO BE PLOTTED
C           YMIN     -MINIMUM 'Y' VALUE TO BE PLOTTED
C           YMAX     -MAXIMUM 'Y' VALUE TO BE PLOTTED
C           S        -ARRAY OF LENGTH 3*(NXPTS*NYPLS+NXPLS*NYPTS)
C           NXPTS    -NUMBER OF POINTS DESCRIBING LINES IN PLANES PARALLEL
C                     TO THE XZ PLANE
C           NYPLS    -NUMBER OF PLANES PARALLEL TO THE XZ PLANE
C           NXPLS    -NUMBER OF PLANES PARALLEL TO THE YZ PLANE
C           NYPTS    -NUMBER OF POINTS DESCRIBING LINES IN PLANES PARALLEL
C                     TO THE YZ PLANE
C           F        -FUNCTION SUBPROGRAM SUPPLIED BY THE USER TO DESCRIBE
C                     THE SURFACE TO BE DRAWN.  IT MUST BE OF THE FORM
C                     FUNCTION F(X,Y).
C           CUBE     -LOGICAL VARIABLE SET TO .TRUE. IF THE THREE
C                     DIRECTIONS ARE TO BE SCALED INDEPENDENTLY OF EACH
C                     OTHER, AND SET TO .FALSE. IF THE THREE DIRECTIONS
C                     ARE TO BE SCALED DEPENDENTLY
C           ALPHA    -ANGLE OF ROTATION ABOUT THE X AXIS GIVEN IN DEGREES
C           BETA     -ANGLE OF ROTATION ABOUT THE Y AXIS GIVEN IN DEGREES
C           GAMMA    -ANGLE OF ROTATION ABOUT THE Z AXIS GIVEN IN DEGREES
C           DRAWME   -LOGICAL VARIABLE SET TO .TRUE. IF THE ROTATED
C                    VIEW IS TO BE DRAWN, AND SET TO .FALSE. IF THIS
C                    VIEW IS NOT TO BE DRAWN.
C           LAB      -ARRAY OF BCD CHARACTERS USED TO LABEL THE DRAWING.
C                     (THIS ARRAY NEED NOT BE SUPPLIED)
C
C         REMARKS
C                    A CALL TO PLOT3D DEFINES AND SCALES THE SURFACE.
C                    SUBSEQUENT CALLS TO ROTATE ROTATE IT TO NEW FRAME OF
C           REFERENCE.  THE SURFACE IS NOT RETURNED TO THE ORIGINAL
C           POSITION AFTER A ROTATION.  THE SURFACE IS ROTATED ABOUT
C           THE STATIONARY XYZ AXES IN THIS ORDER+ FIRST ABOUT THE 'X'
C           AXIS THEN ABOUT 'Y' AND FINALLY ABOUT 'Z'.
C
```

```
C         SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           PLT770   (A CALCOMP SUBROUTINE)
C           NUMBER   (A CALCOMP SUBROUTINE)
C           SYMBOL   (A CALCOMP SUBROUTINE)
C           DRAW
C           WRITE
C           TRNMAT
C           PHI
C           SCALE
C           AXIS
C           NOTE-- PLT770 MOVES THE PEN TO (X,Y) ON THE PLOTTING
C                  SURFACE, WITH THE PEN UP OR DOWN.
C                  NUMBER DRAWS A NUMBER AT (X,Y).
C                   SYMBOL DRAWS A BCD SYMBOL AT (X,Y).
C                   A FUNCTION SUBPROGRAM SUPPLIED BY THE USER TO
C                   DESCRIBE THE SURFACE TO BE DRAWN.  THE NAME OF THIS
C                   FUNCTION SUBPROGRAM MUST APPEAR IN AN EXTERNAL
C                   STATEMENT OF THE USER'S CALLING PROGRAM.
C
C     ******************************************************************
C
      SUBROUTINE PLOT3D(XMIN,XMAX,YMIN,YMAX,S,NXPTS,NYPLS,
     * NXPLS,NYPTS,F,CUBE)
      LOGICAL CUBE
      DIMENSION S(1)
      DATA IK/0/
      IF (IK.NE.0) GO TO 1
      CALL WRITE
      IK=1
    1 FNX1 = NXPTS-1
      FNY1 = NYPLS-1
      FNX2 = NXPLS-1
      FNY2 = NYPTS-1
      L1 = NXPTS*NYPLS
      L2 = NXPLS*NYPTS
      N1=L1
      N2=N1+L1
      N3=N2+L1
      N4=N3+L2
      N5=N4+L2
      DX=XMAX-XMIN
      DY=YMAX-YMIN
C
C
      DO 2 I=1,NXPTS
      DO 2 J=1,NYPLS
      II = NXPTS*(J-1) + I
      I1=II
      I2=N1+II
      I3=N2+II
      S(I1)=XMIN+DX*FLOAT(I-1)/FNX1
      S(I2)=YMIN+DY*FLOAT(J-1)/FNY1
2     S(I3)=F(S(I1),S(I2))
C
C
      DO 3 I=1,NXPLS
      DO 3 J=1,NYPTS
      II = NXPLS*(J-1) + I
      I1=N3+II
```

```
      I2=N4+II
      I3=N5+II
      S(I1)=XMIN+DX*FLOAT(I-1)/FNX2
      S(I2)=YMIN+DY*FLOAT(J-1)/FNY2
3     S(I3)=F(S(I1),S(I2))
C
C
      N1=N1+1
      N2=N2+1
      N3=N3+1
      N4=N4+1
      N5=N5+1
      CALL SCALE(XMIN,XMAX,YMIN,YMAX,S(1),S(N1),S(N2),NXPTS,NYPLS,
     * S(N3),S(N4),S(N5),NXPLS,NYPTS,CUBE)
C
C
      CALL AXIS(0,.FALSE.)
C
C
      ASUM=0.0
      BSUM=0.0
      CSUM=0.0
      RETURN
C
C
      ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
      LOGICAL DRAWME
      CALL PLOT(0.0,0.0,-3)
      CALL PLOT(0.0,5.0,-3)
      CALL TRNMAT (ALPHA,BETA,GAMMA)
      CALL PHI  (S(1),S(N1),S(N2),NXPTS,NYPLS)
      CALL PHI (S(N3),S(N4),S(N5),NXPLS,NYPTS)
      CALL AXIS(1,DRAWME)
      ASUM=ASUM+ALPHA
      BSUM=BSUM+BETA
      CSUM=CSUM+GAMMA
      IF(.NOT.DRAWME) RETURN
      CALL DRAWS(S(1),S(N1),S(N2),NXPTS,NYPLS)
      CALL DRAW(S(N3),S(N4),S(N5),NXPLS,NYPTS)
C
C
      CALL WRITES(ASUM,BSUM,CSUM)
      CALL PLOT(15.0,-5.0,-3)
      RETURN
      END




      BLOCK DATA
      COMMON/LABEL/ LAB(12)
      DATA LAB /12*6H
      END
```

```
      SUBROUTINE DRAW    (X,Y,Z,NX,NY)
      DIMENSION X(NX,NY), Y(NX,NY), Z(NX,NY)
      DO 1 I=1,NX
      CALL PLOT (Y(I,1),Z(I,1),3)
      DO 1 J=2,NY
      CALL PLOT (Y(I,J),Z(I,J),2)
1     CONTINUE
      RETURN
      ENTRY DRAWS (X,Y,Z,NX,NY)
      DO 2 J=1,NY
      CALL PLOT (Y(1,J),Z(1,J),3)
      DO 2 I=2,NX
      CALL PLOT (Y(I,J),Z(I,J),2)
2     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WRITE
      COMMON /MAXES/ XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
      COMMON /XCPIDX/ NBL,RUNNO,CPID(8)
      COMMON /LABEL/ LAB(12)
      DATA BLOCK,ALPHA,BETA,GAMMA/5HBLOCK,6HALPHA=,5HBETA=,6HGAMMA=/
      DATA HXMIN,HXMAX,HYMIN,HYMAX,HZMIN,HZMAX/5HXMIN=,5HXMAX=,5HYMIN=,5
     1HYMAX=,5HZMIN=,5HZMAX=/
      CALL PLOT (0.0,0.0,-3)
      CALL PLOT (0.0,5.0,-3)
      CPID(5)=RUNNO
      CALL SYMBOL (0.0,-4.0,.25,CPID,90.0,30)
      CALL SYMBOL (0.0,-5.0,0.05,BLOCK,0.0,6)
      NBL1=NBL-1
      FNBL=NBL1
      CALL NUMBER (0.3,-5.0,0.05,FNBL,0.0,-1)
      WRITE (6,2) NBL1
      CALL PLOT(9.0,-5.0,-3)
      RETURN
      ENTRY WRITES(A1,B1,C1)
      A=AMOD(A1,360.0)
      B=AMOD(B1,360.0)
      C=AMOD(C1,360.0)
      CALL SYMBOL (0.0,-5.0,0.05,BLOCK,0.0,6)
      FNBL=NBL
      CALL NUMBER (0.3,-5.0,0.05,FNBL,0.0,-1)
      WRITE (6,2) NBL
      IF (LAB(1).EQ.ISTR) GO TO 1
      CALL SYMBOL (-3.0,-4.0,0.1,LAB,0.0,72)
1     CALL SYMBOL (-3.0,-4.25,0.1,ALPHA,0.0,6)
      CALL NUMBER (-2.4,-4.25,0.1,A,0.0,5)
      CALL SYMBOL (-3.0,-4.5,0.1,BETA,0.0,6)
      CALL NUMBER (-2.4,-4.5,0.1,B,0.0,5)
      CALL SYMBOL (-3.0,-4.75,0.1,GAMMA,0.0,6)
      CALL NUMBER (-2.4,-4.75,0.1,C,0.0,5)
      CALL SYMBOL (-1.0,-4.25,0.1,HXMIN,0.0,5)
      CALL NUMBER (-0.4,-4.25,0.1,XMIN,0.0,5)
      CALL SYMBOL (-1.0,-4.5,0.1,HYMIN,0.0,5)
```

```
      CALL NUMBER (-0.4,-4.5,0.1,YMIN,0.0,5)
      CALL SYMBOL (-1.0,-4.75,0.1,HZMIN,0.0,5)
      CALL NUMBER (-0.4,-4.75,0.1,ZMIN,0.0,5)
      CALL SYMBOL (1.0,-4.25,0.1,HXMAX,0.0,5)
      CALL NUMBER (1.6,-4.25,0.1,XMAX,0.0,5)
      CALL SYMBOL (1.0,-4.5,0.1,HYMAX,0.0,5)
      CALL NUMBER (1.6,-4.5,0.1,YMAX,0.0,5)
      CALL SYMBOL (1.0,-4.75,0.1,HZMAX,0.0,5)
      CALL NUMBER (1.6,-4.75,0.1,ZMAX,0.0,5)
      RETURN
C
2     FORMAT (120X,5HBLOCK,I5)
      END
```

```
      SUBROUTINE SCALE (XMIN,XMAX,YMIN,YMAX,X1,Y1,Z1,NX1,NY1,X2,Y2,Z2,NX
     12,NY2,CUBE)
      DIMENSION X1(NX1,NY1), Y1(NX1,NY1), Z1(NX1,NY1), X2(NX2,NY2), Y2(N
     1X2,NY2), Z2(NX2,NY2)
      COMMON /MAXES/ XMIN1,XMAX1,YMIN1,YMAX1,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      XMIN1=XMIN
      XMAX1=XMAX
      YMIN1=YMIN
      YMAX1=YMAX
      MAXX=(XMAX-XMIN)/3.5
      MAXY=(YMAX-YMIN)/3.5
      ZMAX=Z1(1,1)
      ZMIN=Z1(1,1)
      DO 1 I=1,NX1
      DO 1 J=1,NY1
      ZMAX=AMAX1(Z1(I,J),ZMAX)
1     ZMIN=AMIN1(Z1(I,J),ZMIN)
      DO 2 I=1,NX2
      DO 2 J=1,NY2
      ZMAX=AMAX1(Z2(I,J),ZMAX)
2     ZMIN=AMIN1(Z2(I,J),ZMIN)
      MAXZ=(ZMAX-ZMIN)/3.5
      IF (CUBE) GO TO 3
      MAXX=AMAX1(MAXX,MAXY,MAXZ)
      MAXY=MAXX
      MAXZ=MAXX
3     DO 4 I=1,NX1
      DO 4 J=1,NY1
      X1(I,J)=(X1(I,J)-(XMAX+XMIN)/2.)/MAXX
      Y1(I,J)=(Y1(I,J)-(YMAX+YMIN)/2.)/MAXY
4     Z1(I,J)=(Z1(I,J)-(ZMAX+ZMIN)/2.)/MAXZ
      DO 5 I=1,NX2
      DO 5 J=1,NY2
      X2(I,J)=(X2(I,J)-(XMAX+XMIN)/2.)/MAXX
      Y2(I,J)=(Y2(I,J)-(YMAX+YMIN)/2.)/MAXY
5     Z2(I,J)=(Z2(I,J)-(ZMAX+ZMIN)/2.)/MAXZ
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      LOGICAL DRAWME
      DIMENSION X(4,1), Y(4,1), Z(4,1)
      IF(I.NE.0) GO TO 1
      X(1,1) = 0.
      X(2,1) = 1.0
      X(3,1) = 0.
      X(4,1) = 0.
      Y(1,1) = 0.
      Y(2,1) = 0.
      Y(3,1) = 1.0
      Y(4,1) = 0.
      Z(1,1) = 0.
      Z(2,1) = 0.
      Z(3,1) = 0.
      Z(4,1) = 1.0
      RETURN
    1 CALL PHI(X,Y,Z,4,1)
      IF(DRAWME) GO TO 2
      RETURN
    2 YY = Y(1,1) + 4.0
      ZZ = Z(1,1) + 4.0
      CALL PLOT(YY,ZZ,3)
      CALL PLOT(Y(2,1)+4., Z(2,1)+4., 2)
      CALL SYMBOL(Y(2,1)+3.97143, Z(2,1)+3.95, .1, 1HX, 0.0, 1)
      CALL PLOT(YY,ZZ,3)
      CALL PLOT(Y(3,1)+4., Z(3,1)+4., 2)
      CALL SYMBOL(Y(3,1)+3.97143, Z(3,1)+3.95, .1, 1HY, 0.0, 1)
      CALL PLOT(YY,ZZ,3)
      CALL PLOT(Y(4,1)+4., Z(4,1)+4., 2)
      CALL SYMBOL(Y(4,1)+3.97143, Z(4,1)+3.95, .1, 1HZ, 0.0, 1)
      RETURN
      END
```

```
      SUBROUTINE TRNMAT (ALPHA,BETA,GAMMA)
      COMMON /MATRIX/ TMAT(3,3)
      A=ALPHA/57.2957795
      B=BETA/57.2957795
      C=GAMMA/57.2957795
      SINA=SIN(A)
      SINB=SIN(B)
      SINC=SIN(C)
      COSA=COS(A)
      COSB=COS(B)
      COSC=COS(C)
      TMAT(1,1)=COSC*COSB
      TMAT(1,2)=COSC*SINB*SINA-SINC*COSA
      TMAT(1,3)=COSC*SINB*COSA+SINC*SINA
      TMAT(2,1)=SINC*COSB
      TMAT(2,2)=SINC*SINB*SINA+COSC*COSA
      TMAT(2,3)=SINC*SINB*COSA-COSC*SINA
      TMAT(3,1)=-SINB
      TMAT(3,2)=COSB*SINA
      TMAT(3,3)=COSB*COSA
      RETURN
      END
```

```
      SUBROUTINE PHI (X,Y,Z,NX,NY)
      DIMENSION X(NX,NY), Y(NX,NY), Z(NX,NY)
      COMMON /MATRIX/ TMAT(3,3)
C
C
      DO 1 I=1,NX
      DO 1 J=1,NY
C
C
      XP=TMAT(1,1)*X(I,J)+TMAT(1,2)*Y(I,J)+TMAT(1,3)*Z(I,J)
      YP=TMAT(2,1)*X(I,J)+TMAT(2,2)*Y(I,J)+TMAT(2,3)*Z(I,J)
      ZP=TMAT(3,1)*X(I,J)+TMAT(3,2)*Y(I,J)+TMAT(3,3)*Z(I,J)
C
C
      X(I,J)=XP
      Y(I,J)=YP
1     Z(I,J)=ZP
C
C
      RETURN
      END
```

APPENDIX B

Routines for System 360

The following are listings of the subroutines contained in the PLOT3D package for the IBM System 360/67. These routines are used with an on-line CDC microfilm recorder. The internal routines which do the plotting are explained in the comments cards.

```
C
C
C
C         SUBROUTINE PLOT3D
C
C         PURPOSE
C           DRAW AND ROTATE THREE-DIMENSIONAL SURFACES WITH 360/67
C           DD80 MICROFILM RECORDER
C
C         USAGE
C           CALL PLOT3D(XMIN,XMAX,YMIN,YMAX,S,NXPTS,NYPLS,NXPLS,NYPTS,
C                F,CUBE)
C           CALL ROTATE(ALPHA,BETA,GAMMA,DRAWME)
C           COMMON /LABEL/ LAB(15)
C
C         DESCRIPTION OF PARAMETERS
C           XMIN    -MINIMUM 'X' VALUE TO BE PLOTTED
C           XMAX    -MAXIMUM 'X' VALUE TO BE PLOTTED
C           YMIN    -MINIMUM 'Y' VALUE TO BE PLOTTED
C           YMAX    -MAXIMUM 'Y' VALUE TO BE PLOTTED
C           S       -ARRAY OF LENGTH 3*(NXPTS*NYPLS+NXPLS*NYPTS)
C           NXPTS   -NUMBER OF POINTS DESCRIBING LINES IN PLANES PARALLEL
C                    TO THE XZ PLANE
C           NYPLS   -NUMBER OF PLANES PARALLEL TO THE XZ PLANE
C           NXPLS   -NUMBER OF PLANES PARALLEL TO THE YZ PLANE
C           NYPTS   -NUMBER OF POINTS DESCRIBING LINES IN PLANES PARALLEL
C                    TO THE YZ PLANE
C           F       -FUNCTION SUBPROGRAM SUPPLIED BY THE USER TO DESCRIBE
C                    THE SURFACE TO BE DRAWN.  IT MUST BE OF THE FORM
C                    FUNCTION F(X,Y).
C           CUBE    -LOGICAL VARIABLE SET TO .TRUE. IF THE THREE
C                    DIRECTIONS ARE TO BE SCALED INDEPENDENTLY OF EACH
C                    OTHER, AND SET TO .FALSE. IF THE THREE DIRECTIONS
C                    ARE TO BE SCALED DEPENDENTLY
C           ALPHA   -ANGLE OF ROTATION ABOUT THE X AXIS GIVEN IN DEGREES
C           BETA    -ANGLE OF ROTATION ABOUT THE Y AXIS GIVEN IN DEGREES
C           GAMMA   -ANGLE OF ROTATION ABOUT THE Z AXIS GIVEN IN DEGREES
C           DRAWME  -LOGICAL VARIABLE SET TO .TRUE. IF THE ROTATED
C                   VIEW IS TO BE DRAWN, AND SET TO .FALSE. IF THIS
C                   VIEW IS NOT TO BE DRAWN.
C           LAB     -ARRAY OF BCD CHARACTERS USED TO LABEL THE DRAWING.
C                   IF LABEL IS DESIRED, SUPPLY 15 OR LESS FOUR-CHARACTER
C                   WORDS IN COMMON/LABEL/ LAB(15)
C
C         REMARKS
C                   A CALL TO PLOT3D DEFINES AND SCALES THE SURFACE.
C                   SUBSEQUENT CALLS TO ROTATE ROTATE IT TO NEW FRAME OF
C           REFERENCE.  THE SURFACE IS NOT RETURNED TO THE ORIGINAL
C           POSITION AFTER A ROTATION.  THE SURFACE IS ROTATED ABOUT
C           THE STATIONARY XYZ AXES IN THIS ORDER+ FIRST ABOUT THE 'X'
C           AXIS THEN ABOUT 'Y' AND FINALLY ABOUT 'Z'.
```

```
C
C         SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           SCALE
C           DRAW
C           AXIS
C           TRNMAT
C           PHI
C           WRITE
C                   A FUNCTION SUBPROGRAM SUPPLIED BY THE USER TO
C                   DESCRIBE THE SURFACE TO BE DRAWN.  THE NAME OF THIS
C                   FUNCTION SUBPROGRAM MUST APPEAR IN AN EXTERNAL
C                   STATEMENT OF THE USER'S CALLING PROGRAM.
C           IN ADDITION, PLOT3D CALLS ROUTINES DEVELOPED HERE AT
C           LEWIS FOR MICROFILM PLOTTING. THEY ARE#
C           LRCURV  THE BASIC ROUTINE, DRAWS AN ARRAY OF (X,Y) POINTS
C                   AS A POINT PLOT, VECTOR PLOT, OR SPECIAL
C                   CHARACTER PLOT. PLOT3D USES VECTOR PLOTS.
C           LRGRID  SETS UP VERTICAL AND HORIZONTAL GRID LINES
C                   ON THE PLOTTING SURFACE. A CALL TO THIS ROUTINE
C                   IS PRESENTLY REQUIRED, BUT PLOT3D USES NO GRID.
C           LRANGE  DEFINES THE RANGES OF X AND Y IN THE USER'S
C                   DATA UNITS.
C           LRMRGN  DESCRIBES WHAT FRACTIONS OF THE PLOTTING
C                   SURFACE ARE TO BE USED AS MARGINS, LEFT,
C                   RIGHT, BOTTOM, AND TOP.
C           LRLEGN  PLOTS EBCDIC ARRAYS STARTING AT (X,Y) IN
C                   PLOTTING UNITS.
C           LRCNVT  CONVERTS NUMERICAL FIELDS TO EBCDIC ARRAYS
C                   FOR USE BY LRLEGN.
C           LRCHSZ  DEFINES THE SIZE OF CHARACTERS TO BE DRAWN.
C           LRNON,LRNOFF  CONTROL THE INTENSITY OF PLOTS.
C
C     ****************************************************************
      SUBROUTINE PLOT3D(XMIN,XMAX,YMIN,YMAX,S,NXPTS,NYPLS,
     * NXPLS,NYPTS,F,CUBE)
C
      LOGICAL CUBE
      DIMENSION S(1), AX(1), BY(1)
      DATA AX,BY /0.0,0.0/
      FNX1 = NXPTS-1
      FNY1 = NYPLS-1
      FNX2 = NXPLS-1
      FNY2 = NYPTS-1
      L1 = NXPTS*NYPLS
      L2 = NXPLS*NYPTS
      N1 = L1
      N2 = N1+L1
      N3 = N2+L1
      N4 = N3+L2
      N5 = N4+L2
      DX = XMAX-XMIN
      DY = YMAX-YMIN
      CALL LRGRID(1,1,0.0,0.0)
      CALL LRANGE(-3.55,5.45,-3.55,5.45)
      CALL LRMRGN(1.0,0.0,3.0,0.0)
      CALL LRCHSZ(2)
C
C
      DO 2 I=1,NXPTS
      DO 2 J=1,NYPLS
      II = NXPTS*(J-1) + I
      I1 = II
      I2 = N1+II
      I3 = N2+II
      S(I1) = XMIN+DX*FLOAT(I-1)/FNX1
```

```
      S(I2) = YMIN+DY*FLOAT(J-1)/FNY1
    2 S(I3) = F(S(I1),S(I2))
C
      DO 3 I=1,NXPLS
      DO 3 J=1,NYPTS
      II = NYPTS*(I-1) + J
      I1 = N3+II
      I2 = N4+II
      I3 = N5+II
      S(I1) = XMIN+DX*FLOAT(I-1)/FNX2
      S(I2) = YMIN+DY*FLOAT(J-1)/FNY2
    3 S(I3) = F(S(I1),S(I2))
C
C
      N1 = N1+1
      N2 = N2+1
      N3 = N3+1
      N4 = N4+1
      N5 = N5+1
      CALL SCALE(XMIN,XMAX,YMIN,YMAX,S(1),S(N1),S(N2),NXPTS,NYPLS,
     * S(N3),S(N4),S(N5),NXPLS,NYPTS,CUBE)
C
C
      CALL AXIS(0,.FALSE.)
C
C
      ASUM = 0.0
      BSUM = 0.0
      CSUM = 0.0
      CALL WRITE(ASUM,BSUM,CSUM)
      RETURN
C
C
      ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
      LOGICAL DRAWME
      CALL TRNMAT(ALPHA,BETA,GAMMA)
      CALL PHI  (S(1),S(N1),S(N2),NXPTS,NYPLS)
      CALL PHI (S(N3),S(N4),S(N5),NYPTS,NXPLS)
      CALL AXIS(1,DRAWME)
      ASUM = ASUM+ALPHA
      BSUM = BSUM+BETA
      CSUM = CSUM+GAMMA
      IF(.NOT.DRAWME) RETURN
      CALL DRAW(S(N1),S(N2),NXPTS,NYPLS)
      CALL DRAW(S(N4),S(N5),NYPTS,NXPLS)
C
C
      CALL WRITES(ASUM,BSUM,CSUM)
      CALL LRCURV(AX,BY,1,1,1HX,1.0)
      RETURN
      END
```

```
      BLOCK DATA
      COMMON /LABEL/ LAB(15)
      DATA LAB / 15*4H     /
      END
```

```
      SUBROUTINE DRAW(Y,Z,NPTS,NPLS)
      DIMENSION Y(NPTS,NPLS),Z(NPTS,NPLS)
      DO 1 I=1,NPLS
      CALL LRCURV(Y(1,I) , Z(1,I) , NPTS,2,1HX,0.0)
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WRITE(A1,B1,C1)
      DIMENSION AXI(3),AXA(3),AYI(3),AYA(3),AZI(3),AZA(3),ALPH(3),BETA(3
     * ),GAM(3),ALINE(14),BLINE(14),CLINE(14)
      DIMENSION BXI(2),BXA(2),BYI(2),BYA(2),BZI(2),BZA(2)
      COMMON /MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
      COMMON /LABEL/ LAB(15)
      EQUIVALENCE (BA,ALINE(1)),(ALPH(1),ALINE(2)),(BXI,ALINE(5)),
     *  (AXI(1),ALINE(7)),(BXA,ALINE(10)),(AXA(1),ALINE(12)),(BB,
     *  BLINE(1)),(BETA(1),BLINE(2)),(BYI,BLINE(5)),(AYI(1),BLINE(7)),
     *  (BYA,BLINE(10)),(AYA(1),BLINE(12)),(BC,CLINE(1)),(GAM(1),
     *  CLINE(2)),(BZI,CLINE(5)),(AZI(1),CLINE(7)),(BZA,CLINE(10)),
     *  (AZA(1),CLINE(12))
      EQUIVALENCE (FAKER,FAKIR)
      DATA BA,BB,BC,BXI,BXA,BYI,BYA,BZI,BZA / 4H A =, 4H B =, 4H C =,
     *  4H  XM, 4HIN =, 4H  XM, 4HAX =, 4H  YM, 4HIN =, 4H  YM, 4HAX =,
     *  4H  ZM, 4HIN =, 4H  ZM, 4HAX =    /
      SICK(XXX) = XXX - FLOAT(IFIX(XXX/360.0))*360.0
      FAKER = FAKIR
      CALL LRCNVT(XMIN,3,AXI,4,12,5)
      CALL LRCNVT(XMAX,3,AXA,4,12,5)
      CALL LRCNVT(YMIN,3,AYI,4,12,5)
      CALL LRCNVT(YMAX,3,AYA,4,12,5)
      CALL LRCNVT(ZMIN,3,AZI,4,12,5)
      CALL LRCNVT(ZMAX,3,AZA,4,12,5)
      RETURN
      ENTRY WRITES(A1,B1,C1)
      CALL LRNON
      A = SICK(A1)
      B = SICK(B1)
      C = SICK(C1)
      CALL LRCNVT(A,3,ALPH,3,12,5)
      CALL LRCNVT(B,3,BETA,3,12,5)
      CALL LRCNVT(C,3,GAM ,3,12,5)
      CALL LRLEGN(LAB,60,0,2.0,2.66,0.)
      CALL LRLEGN(ALINE,56,0,2.0,2.0,0.)
      CALL LRLEGN(BLINE,56,0,2.0,1.33,0.)
      CALL LRLEGN(CLINE,56,0,2.0,0.66,0.)
      CALL LRNOFF
      RETURN
      END
```

```
      SUBROUTINE SCALE(XMIN,XMAX,YMIN,YMAX,X1,Y1,Z1,NX1,NY1,X2,Y2,Z2,NX2
     * ,NY2,CUBE)
      DIMENSION X1(NX1,NY1),Y1(NX1,NY1),Z1(NX1,NY1),
     *          X2(NY2,NX2),Y2(NY2,NX2),Z2(NY2,NX2)
      COMMON /MAXES/ XMIN1,XMAX1,YMIN1,YMAX1,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      XMIN1 = XMIN
      XMAX1 = XMAX
      YMIN1 = YMIN
      YMAX1 = YMAX
      MAXX = (XMAX-XMIN)/3.5
      MAXY = (YMAX-YMIN)/3.5
      ZMAX = Z1(1,1)
      ZMIN = Z1(1,1)
      DO 1 I=1,NX1
      DO 1 J=1,NY1
      ZMAX = AMAX1(Z1(I,J),ZMAX)
    1 ZMIN = AMIN1(Z1(I,J),ZMIN)
      DO 2 I=1,NY2
      DO 2 J=1,NX2
      ZMAX = AMAX1(Z2(I,J),ZMAX)
    2 ZMIN = AMIN1(Z2(I,J),ZMIN)
      MAXZ = (ZMAX-ZMIN)/3.5
      IF(CUBE)  GO TO 21
      MAXX = AMAX1(MAXX,MAXY,MAXZ)
      MAXY = MAXX
      MAXZ = MAXX
   21 DO 3 I=1,NX1
      DO 3 J=1,NY1
      X1(I,J) = (X1(I,J)-(XMAX+XMIN)/2.)/MAXX
      Y1(I,J) = (Y1(I,J)-(YMAX+YMIN)/2.)/MAXY
    3 Z1(I,J) = (Z1(I,J)-(ZMAX+ZMIN)/2.)/MAXZ
      DO 4 I=1,NY2
      DO 4 J=1,NX2
      X2(I,J) = (X2(I,J)-(XMAX+XMIN)/2.)/MAXX
      Y2(I,J) = (Y2(I,J)-(YMAX+YMIN)/2.)/MAXY
    4 Z2(I,J) = (Z2(I,J)-(ZMAX+ZMIN)/2.)/MAXZ
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      LOGICAL DRAWME
      DIMENSION X(4,1),Y(4,1),Z(4,1)
      DIMENSION XA(2),XB(2),XC(2),YA(2),YB(2),YC(2)
      IF(I .NE. 0) GO TO 1
      X(1,1) = 0.
      X(2,1) = .4
      X(3,1) = 0.
      X(4,1) = 0.
      Y(1,1) = 0.
      Y(2,1) = 0.
```

```
      Y(3,1) = .4
      Y(4,1) = 0.
      Z(1,1) = 0.
      Z(2,1) = 0.
      Z(3,1) = 0.
      Z(4,1) = .4
      RETURN
    1 CALL PHI(X,Y,Z,4,1)
      IF(DRAWME) GO TO 2
      RETURN
    2 XA(1) = Y(1,1) + 4.5
      XB(1) = XA(1)
      XC(1) = XA(1)
      XA(2) = Y(2,1) + 4.5
      XB(2) = Y(3,1) + 4.5
      XC(2) = Y(4,1) + 4.5
      YA(1) = Z(1,1) + 3.0
      YB(1) = YA(1)
      YC(1) = YA(1)
      YA(2) = Z(2,1) + 3.0
      YB(2) = Z(3,1) + 3.0
      YC(2) = Z(4,1) + 3.0
      CALL LRCURV(XA,YA,2,2,1HX,0.)
      CALL LRLABL(1HX,1,0,XA(2) -.05,YA(2) -.05,0.)
      CALL LRCURV(XB,YB,2,2,1HX,0.)
      CALL LRLABL(1HY,1,0,XB(2) -.05,YB(2) -.05,0.)
      CALL LRCURV(XC,YC,2,2,1HX,0.)
      CALL LRLABL(1HZ,1,0,XC(2) -.05,YC(2) -.05,0.)
      RETURN
      END
```

```
      SUBROUTINE TRNMAT(ALPHA,BETA,GAMMA)
      COMMON /MATRIX/ TMAT(3,3)
      A = ALPHA/57.2957795
      B = BETA/57.2957795
      C = GAMMA/57.2957795
      SINA = SIN(A)
      SINB = SIN(B)
      SINC = SIN(C)
      COSA = COS(A)
      COSB = COS(B)
      COSC = COS(C)
      TMAT(1,1) = COSC*COSB
      TMAT(1,2) = COSC*SINB*SINA-SINC*COSA
      TMAT(1,3) = COSC*SINB*COSA+SINC*SINA
      TMAT(2,1) = SINC*COSB
      TMAT(2,2) = SINC*SINB*SINA+COSC*COSA
      TMAT(2,3) = SINC*SINB*COSA-COSC*SINA
      TMAT(3,1) = -SINB
      TMAT(3,2) = COSB*SINA
      TMAT(3,3) = COSB*COSA
      RETURN
      END
```

```
      SUBROUTINE PHI(X,Y,Z,NX,NY)
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON /MATRIX/ TMAT(3,3)
C
C
      DO 1 I=1,NX
      DO 1 J=1,NY
C
C
      XP = TMAT(1,1)*X(I,J) + TMAT(1,2)*Y(I,J) + TMAT(1,3)*Z(I,J)
      YP = TMAT(2,1)*X(I,J) + TMAT(2,2)*Y(I,J) + TMAT(2,3)*Z(I,J)
      ZP = TMAT(3,1)*X(I,J) + TMAT(3,2)*Y(I,J) + TMAT(3,3)*Z(I,J)
C
C
      X(I,J) = XP
      Y(I,J) = YP
    1 Z(I,J) = ZP
C
C
      RETURN
      END
```

REFERENCES

1. Roberts, L. G.: Machine Perception of Three-Dimensional Solids. Tech. Rep. 315, Lincoln Lab., Massachusetts Inst. Tech., May 22, 1963. (Available from DDC as AD-407965.)

2. Appel, Arthur: The Notion of Quantitative Invisibility and the Machine Rendering of Solids. Proceedings of the 22nd ACM National Conference. Thompson Books Co., 1967, pp. 387-393.

3. Weiss, Ruth A.: BE VISION, A Package of IBM 7090 FORTRAN Programs to Draw Orthographic Views of Combinations of Plane and Quadric Surfaces. J. Assoc. Comp. Mach., vol. 13, no. 2, Apr. 1966, pp. 194-204.

4. Adamowicz, Michael: GRAPH PAK 1 - A Three-Dimensional Manipulation Program. Tech. Note 400-24 (NASA CR-69665), New York University, Sept. 10, 1965.

5. McCue, G. A ; Dworetsky, M. M.; and DuPrie, H. J.: FORTRAN IV Stereographic Function Representation and Contouring Program. Rep. No. SID-65-1182, North American Aviation, Inc., Sept. 1, 1965.

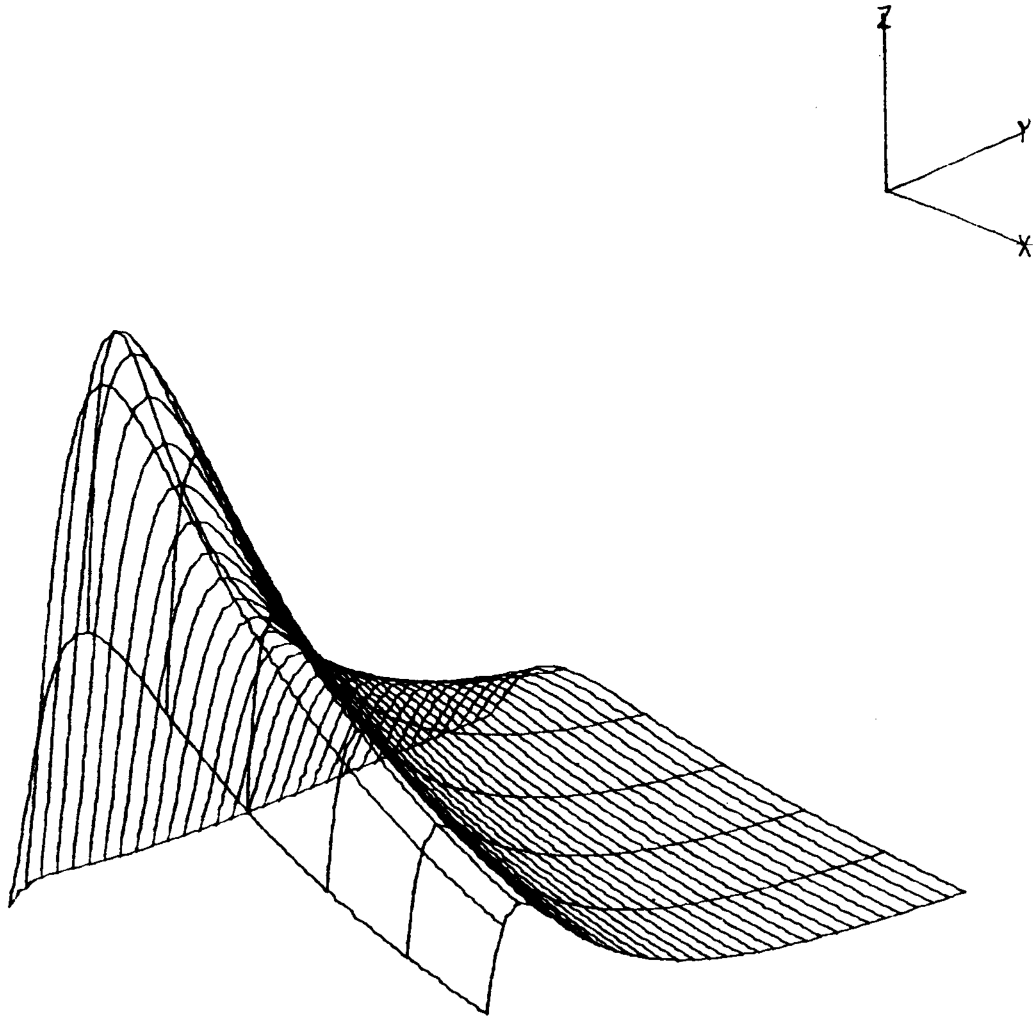
6. Branstetter, E. C.: CONTOUR, A FORTRAN Subroutine for producing Contour Maps. Rep. No. ORNL-4031, Oak Ridge National Lab., Nov. 1966.

7. Nelson, David L.: Perspective Plotting of Two Dimensional Arrays - PLOT 3D, A Computer Program for a Digital Plotter. Tech. Rep. 553 (NASA CR-85798), Univ. of Maryland, March 1966.

8. Swigert, Paul; and Goldstein, Charles M.: Monte Carlo Code for Solution of Planar Electron-Diode Problems Including Electron-Neutral Elastic Collisions. NASA TN D-4043, 1967, App. B.

9. Anon.: Plotter Programming for CalComp Digital Incremental Plotters. California Computer Products, Inc., 1963.

10. Margenau, Henry; and Murphy, George M.: The Mathematics of Physics and Chemistry. Second ed., D. Van Nostrand Co., Inc., 1956, pp. 565-570.



A MAXWELLIAN DISTRIBUTION .

ALPHA= 0.00000	XMIN= .15000	XMAX= 3.00000
BETA= 25.00000	YMIN= .15000	YMAX= 3.00000
GAMMA= 45.00000	ZMIN= .00356	ZMAX= .47511

Figure 1. - Function from statistical mechanics, $z = e^{-(1/x+1/2y)} * (xy)^{-3/2}$.

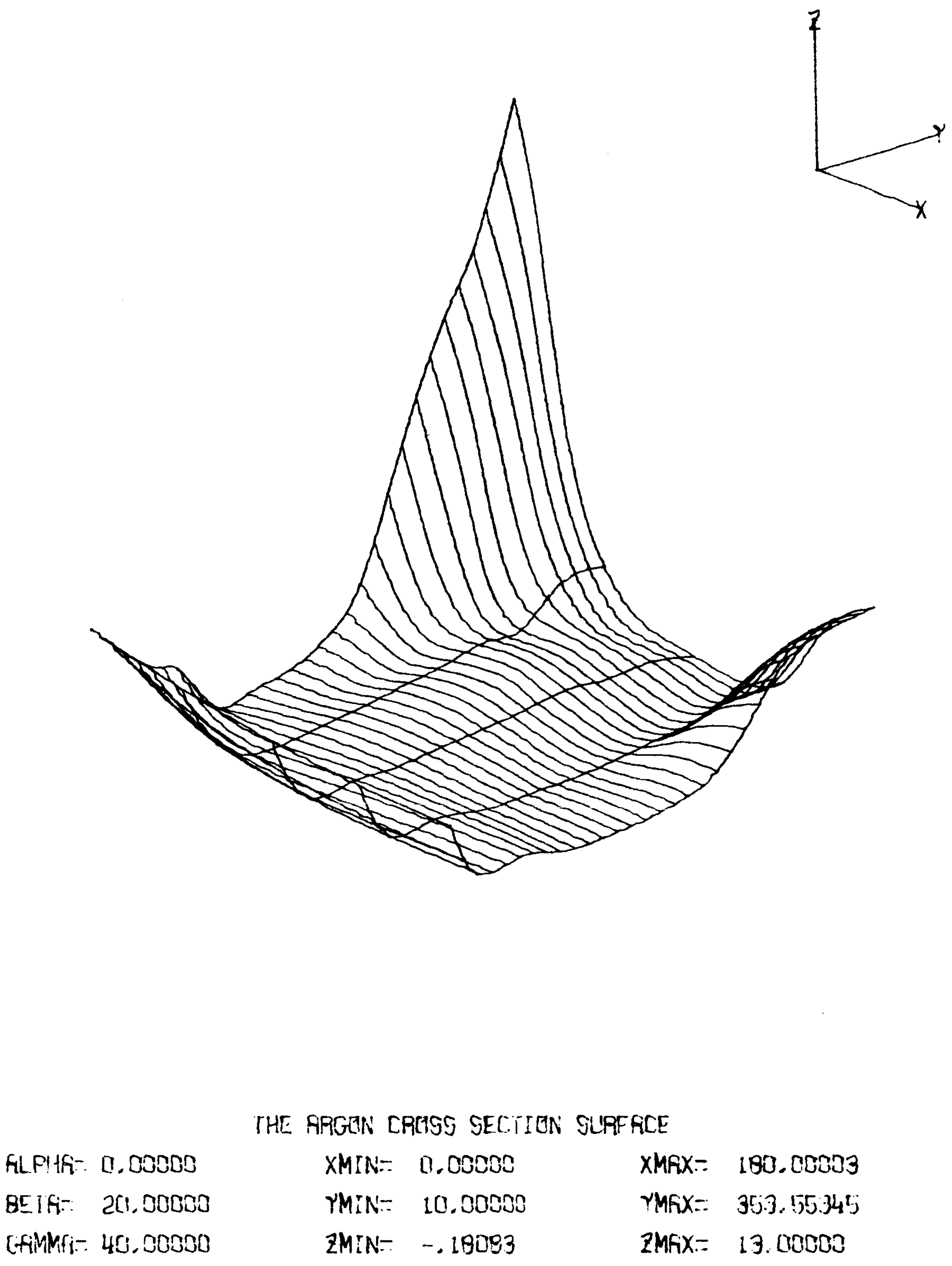
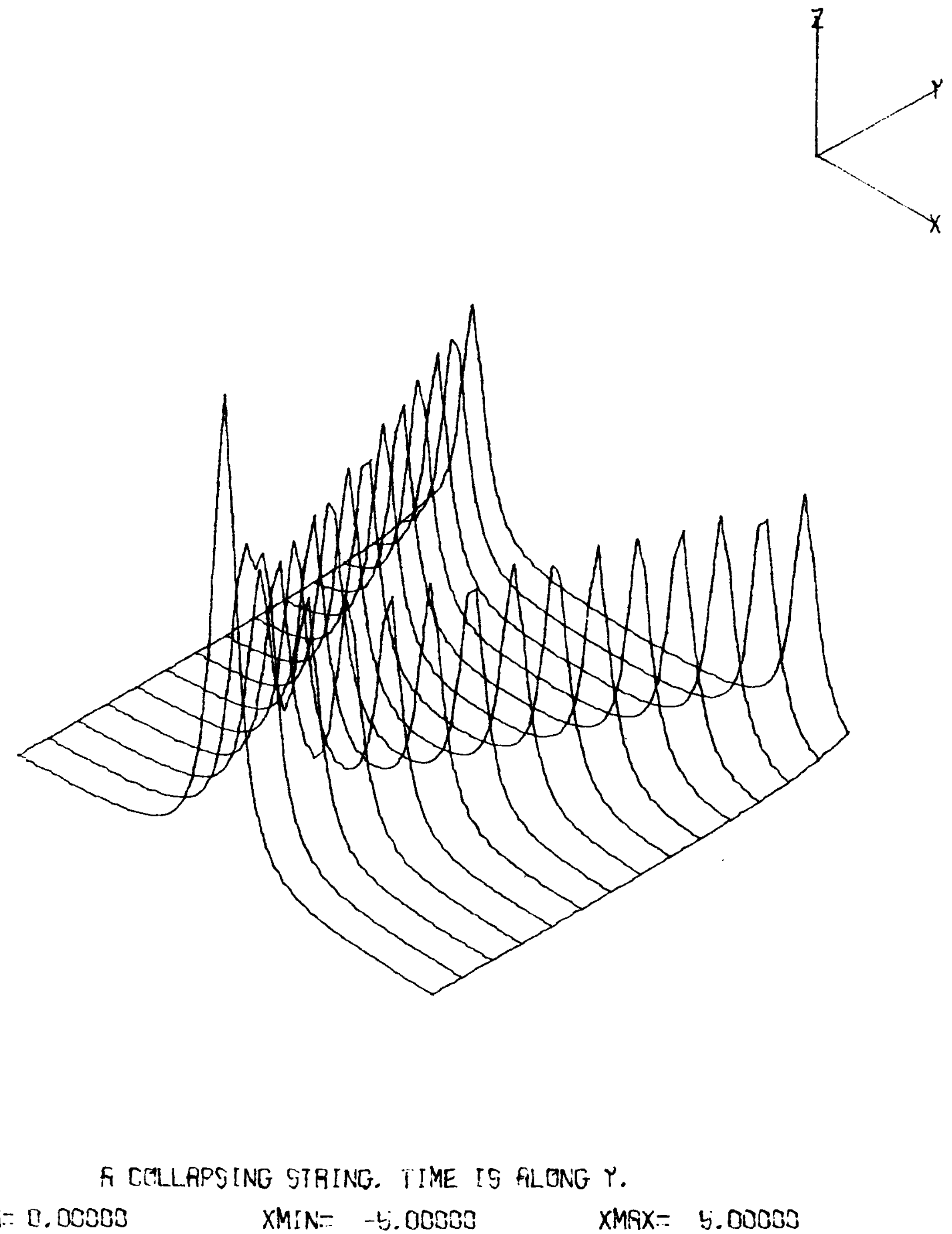


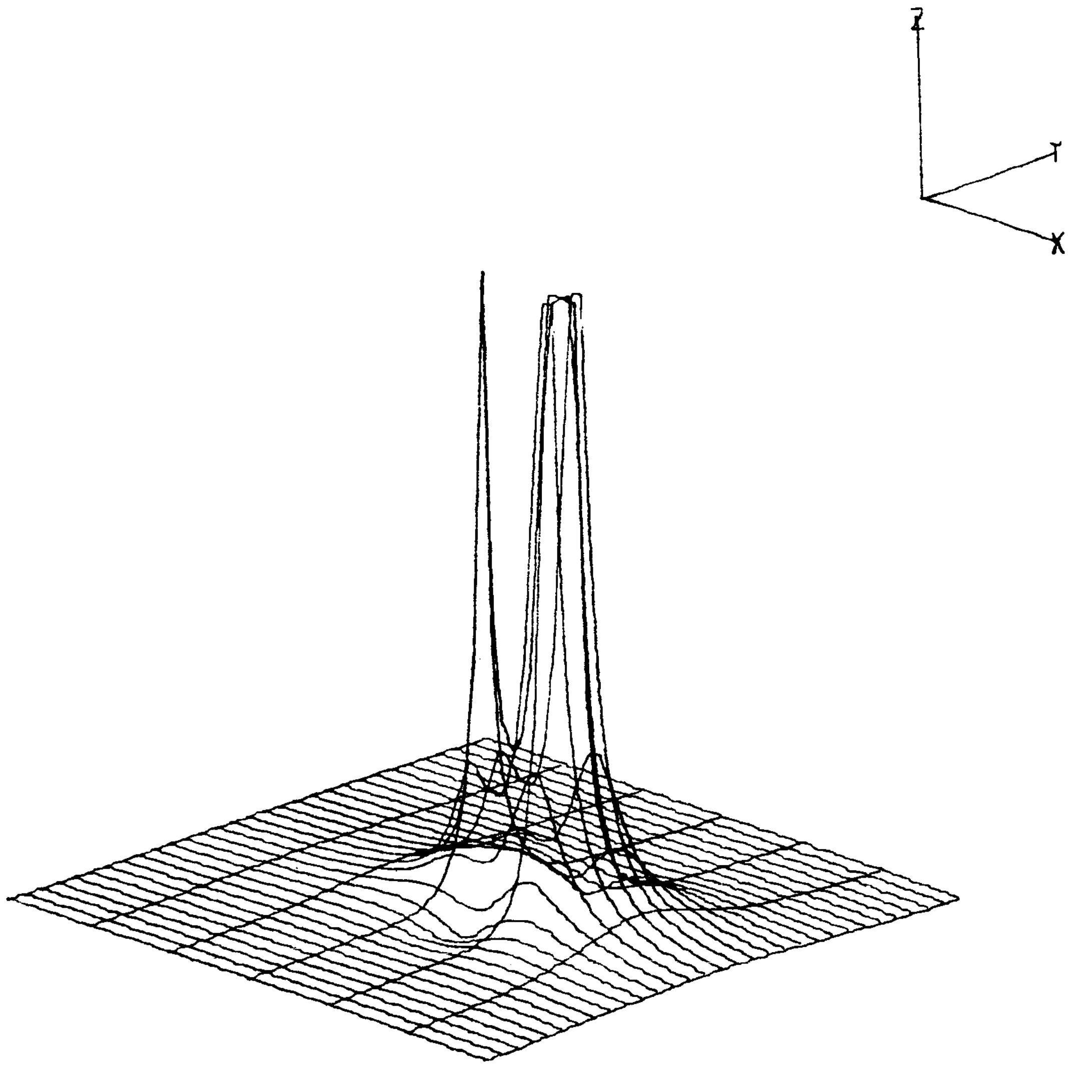
Figure 2. - Surface fit to points derived from experiment and analysis combined.



A COLLAPSING STRING. TIME IS ALONG Y.

ALPHA= 0.00000	XMIN= -5.00000	XMAX= 5.00000
BETA= 35.00000	YMIN= 0.00000	YMAX= 4.00000
GAMMA= 45.00000	ZMIN= .00497	ZMAX= 1.00000

Figure 3. - Particular solution to the two-dimensional wave equation, $z = \frac{1}{2}\left[\frac{1}{1 + 8(x - y)^2} + \frac{1}{1 + 8(x + y)^2}\right]$.

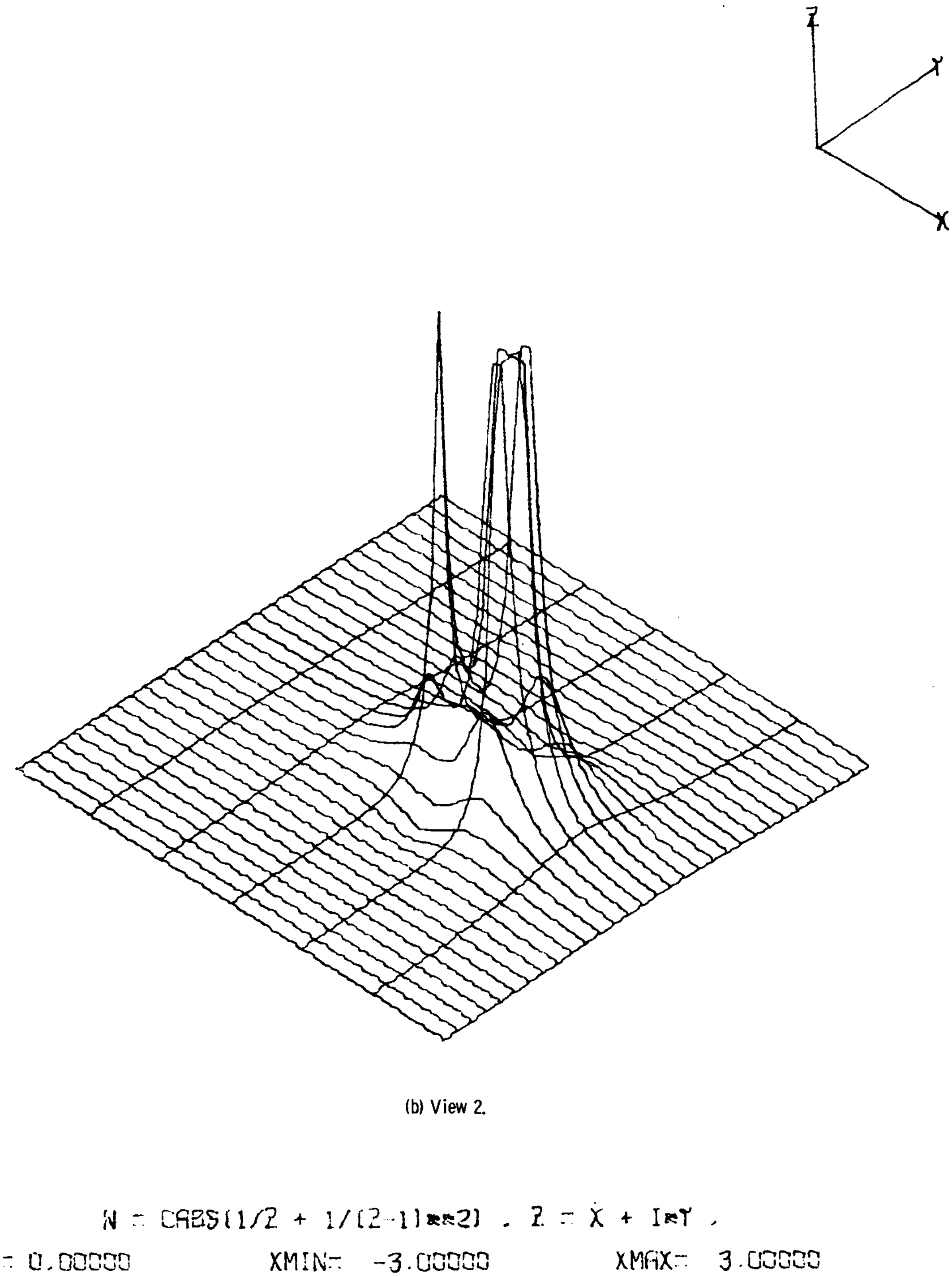


(a) View 1.

W = CABS(1/Z + 1/(Z-1)**2)   Z = X + I*Y

ALPHA=	0.00000	XMIN=	-3.00000	XMAX=	3.00000
BETA=	20.00000	YMIN=	-3.00000	YMAX=	3.00000
GAMMA=	45.00000	ZMIN=	13533	ZMAX=	20.00000

Figure 4. - Function with two poles of different order.

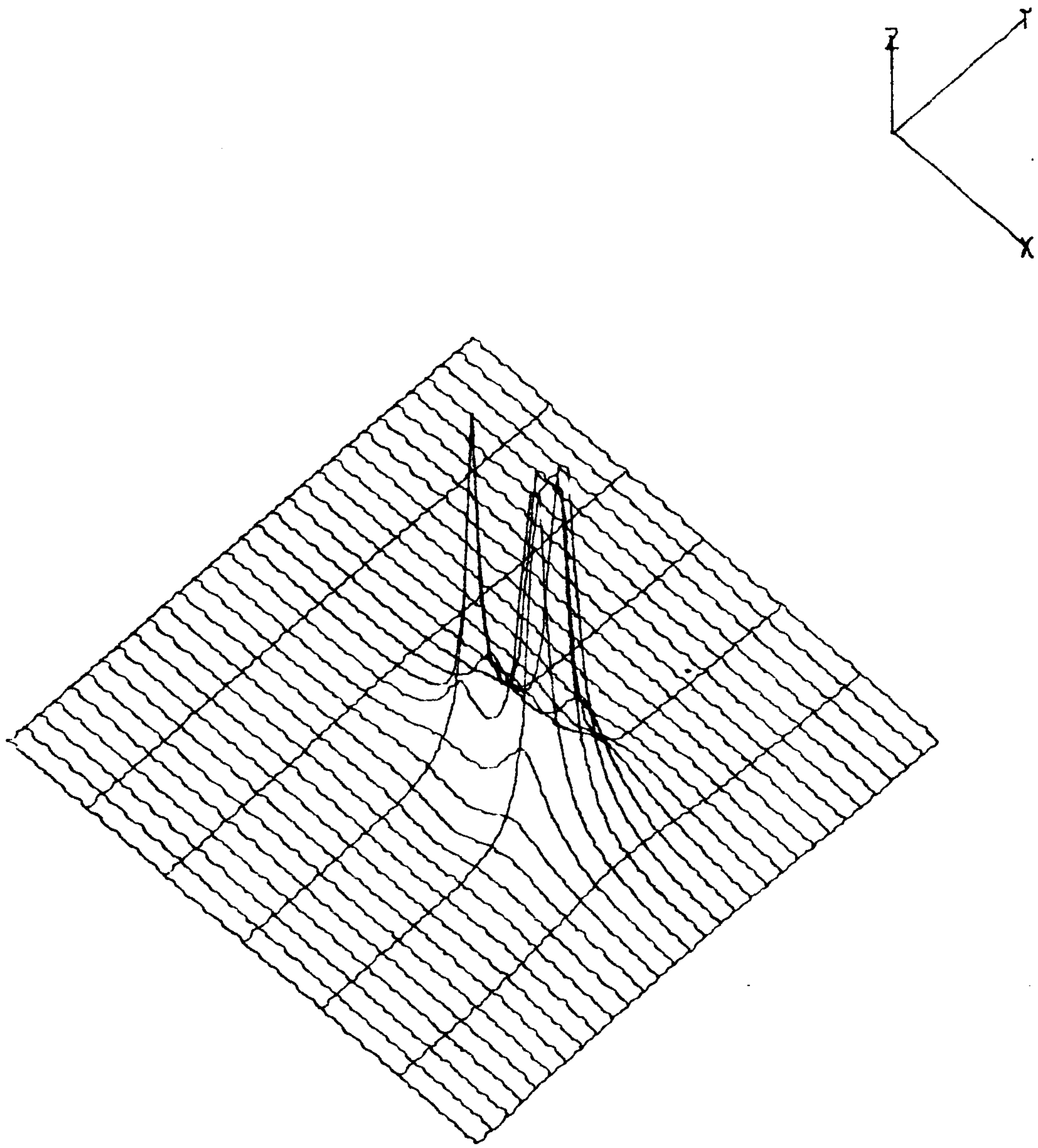


(b) View 2.

W = CABS(1/Z + 1/(Z-1)**2) , Z = X + I*Y ,

ALPHA= 0.00000	XMIN= -3.00000	XMAX= 3.00000
BETA= 40.00000	YMIN= -3.00000	YMAX= 3.00000
GAMMA= 45.00000	ZMIN= .13533	ZMAX= 20.00000

Figure 4. - Continued.

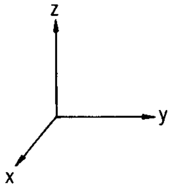


(c) View 3.

W = CABS(1/Z + 1/(Z-1)**2) , Z = X + I*Y

ALPHA= 0.00000	XMIN= -3.00000	XMAX= 3.00000
BETA= 60.00001	YMIN= -3.00000	YMAX= 3.00000
GAMMA= 45.00000	ZMIN= .13533	ZMAX= 20.00000

Figure 4. - Concluded.



Fixed axes

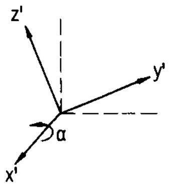


Figure axes

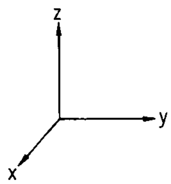
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

or

$$\vec{r}\,' = A \cdot \vec{r}$$

(A is rotation matrix for α, see ref. 10)

(a) Rotation about x-axis.



Fixed axes

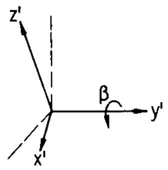


Figure axes

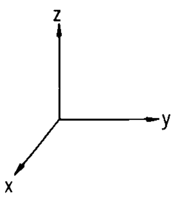
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

or

$$\vec{r}\,' = B \cdot \vec{r}$$

(B is rotation matrix for β)

(b) Rotation about y-axis.



Fixed axes

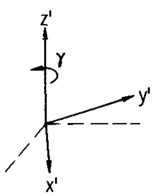


Figure axes

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

or

$$\vec{r}\,' = C \cdot \vec{r}$$

(C is rotation matrix for γ)

(c) Rotation about z-axis.

Figure 5. - Rotations of the surface.

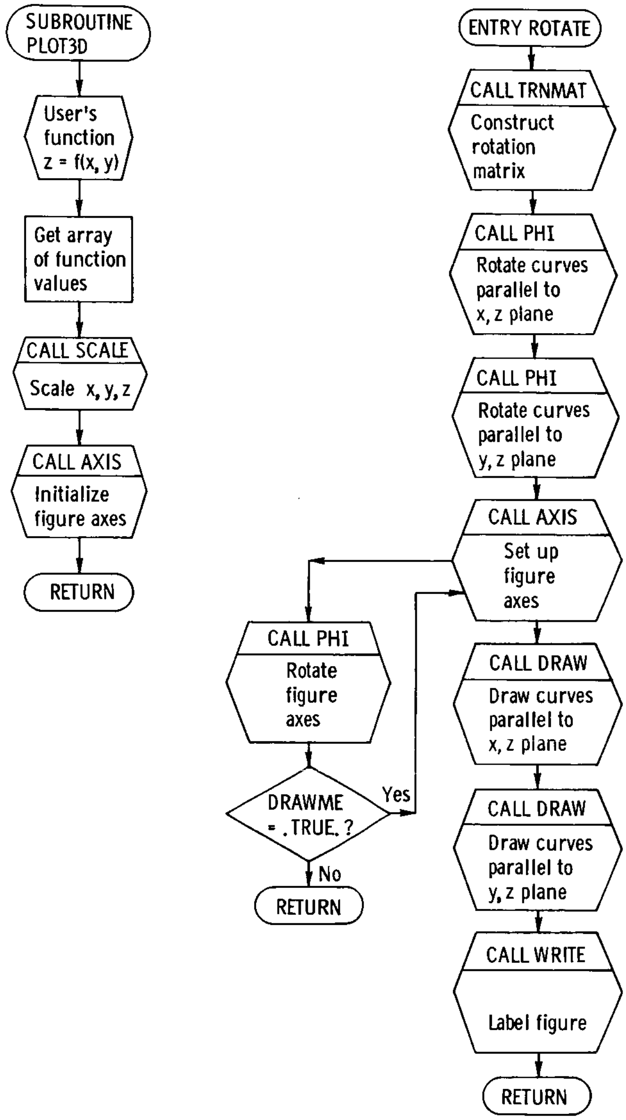


Figure 6. - General flow chart for PLOT3D.

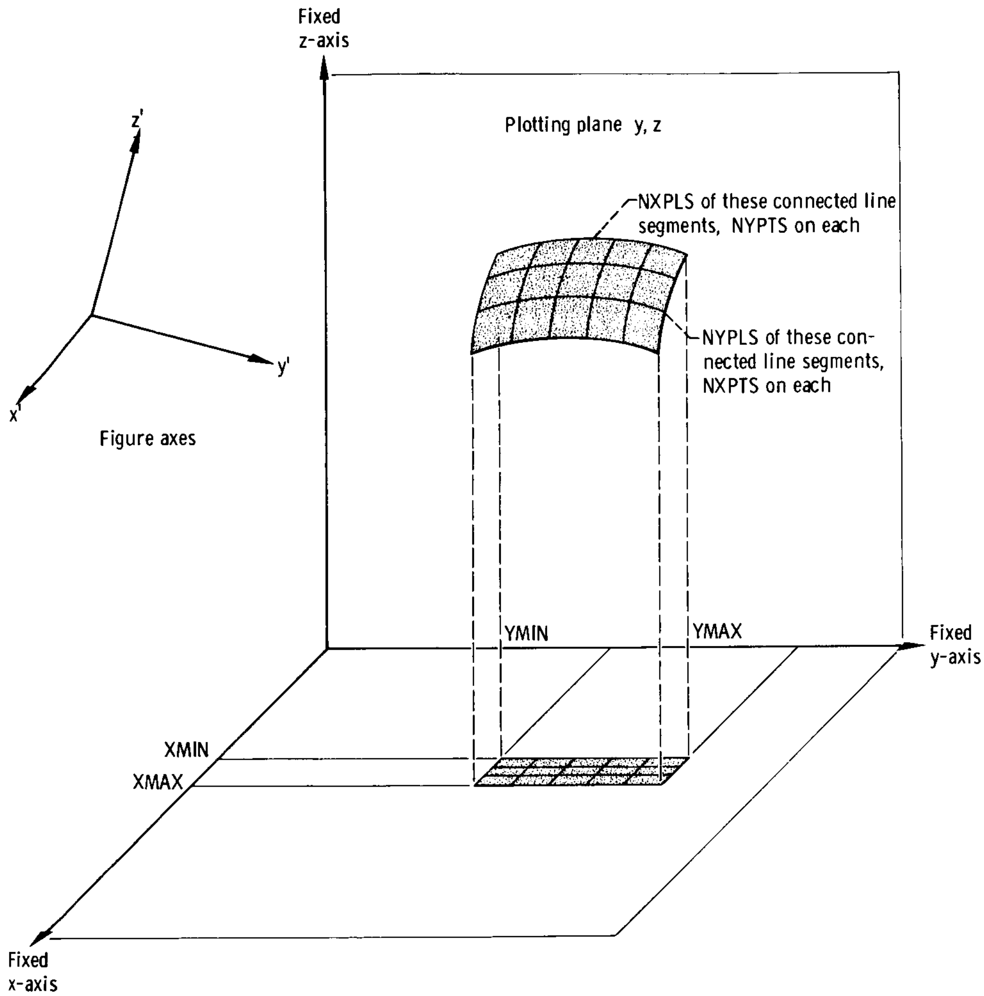


Figure 7. - Surface geometry for PLOT3D.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

— NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

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

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

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

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

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

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

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

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

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Washington, D.C. 20546