DL/CSE/TM 07

technical memorandum Daresbury Laborator

DL/CSE/TM 07

# THE DARESBURY SMOG GRAPHICS PACKAGE

by

P. BRERETON, Daresbury Laboratory

OCTOBER 1978

Science Research Council
Daresbury Laboratory
Daresbury, Warrington WA4 4AD

**IMPORTANT**

The SRC does not accept any responsibility for loss or damage arising from the use of information contained in any of its reports or in any communication about its tests or investigations.

CONTENTS PAGE

## 1. INTRODUCTION

The 'SMOG' package of FORTRAN routines is available to assist a user in the production of plotted output such as drawings and graphs. The package includes both low and high-level routines. The low-level routines, such as line drawing and character output, can be incorporated in a program to build up complex plots exactly as the user requires, alternatively, a single high-level routine can be called to produce complete graphs, histograms or contour plots entirely automatically.

The plotted output is produced either in 'hard-copy' form on a Calcomp or Versatec plotter, or it can be viewed on the screen of a Tektronix 4012 terminal.

A description of the available output devices, the necessary Job Control Statements, the SMOG routines themselves, and some worked examples is given below.

## 2. OUTPUT DEVICES

The Calcomp plotter is a single pen machine which produces output on a roll of paper 279.4mm wide. The Versatec, however, has 1024 pens across a width of 260.10mm which create output as a continuous forward sheet. The Versatec is used in the same way as the Calcomp plotter, so, for the purposes of control, it can be considered to have a single pen drawing action. Users are advised to use the Versatec plotter wherever possible since it is considerably faster than the Calcomp plotter although the resolution is somewhat poorer.

The Tektronix 4012 display is a storage cathode ray tube with an alphanumeric keyboard. It can be used to inspect plotter output, under TSO, before sending output to a hard copy device.

A comparison of the Calcomp and Versatec plotters as well as further information on all the graphical output devices can be found in Chapter 7 of the Daresbury Laboratory Computer User's Guide.

## 3. JOB CONTROL STATEMENTS

The library containing the SMOG routines is called SMOG.LOAD and is accessed by using either the control statements

```
//L.SMOG DD DSN=SMOG.LOAD,DISP=SHR

//L.SYSIN DD *

    INCLUDE SMOG(routine names)
```

```
/*
```

or by concatenating SMOG.LOAD with other libraries being referenced. If, for example, a FORTRAN catalogued procedure is used, SMOG.LOAD is concatenated by including the following statements.

```
//L.SYSLIB DD

//         DD  DSN=SMOG.LOAD,DISP=SHR
```

The particular plotter required is selected using one of the following statements.

```
//G.CALCOMP DD SYSOUT=(G,GPLOT)

//G.VERSATEC DD SYSOUT=(V,VPLOT)
```

In order to inspect output on the Tektronix, no changes are necessary to the program but certain JCL modifications are required. These are described in Chapter 7 of the Computer User's Guide.

If the SMOG routines are called from a PL1 program, additional Job Control statements are necessary for the following purposes.

1) The FORTRAN library must be included at link edit time to satisfy any calls to FORTRAN subprograms made by the graphics library routines.

2) A DD statement specifying FT06F001 is required to accomodate any output and error messages generated by the package.

## 4. MAPPING TO THE PLOTTER

The Calcomp and Versatec each have an associated grid or mesh of addressable points which can be indexed by column and row to give device X and Y coordinates, or 'raster' coordinates. It is necessary therefore, to map from the problem coordinate system (or space) onto the raster coordinates. This is achieved using one of the LIMIT routines (or the default limits) to perform the necessary scaling, translation and mapping (see sections 5.1.3 to 5.1.5).

By convention the device mesh is called Region 0 and the space in which the problem is defined is called Region 1. The user may change the frame of reference using the routine REGION, so that routine parameters may refer to the problem or the raster coordinates.

## 5. SMOG ROUTINES

The SMOG routines described below are divided into the four categories: initialization and termination, low level pen movements and drawing, high level graph, contour and histogram plots and 3-D projection.

In addition to the set of 'real' routines provided in SMOG a smaller set of equivalent 'integer' routines is included to facilitate the direct addressing of raster coordinates as integer values. Since a considerable amount of time, when using SMOG, is devoted to the mapping process a saving can be made by selecting Region 0 (see 5.1.2) and using these 'integer' routines. Each routine name begins with the letter 'J' and is described in the same section as its 'real' counterpart.

### 5.1 Initialization and Termination

In this section the routines for initializing and terminating the plotting device are described. Also included are the routines used to advance the plotter and to define non-default conditions for the plotting area, coordinate system and character size and orientation.

5.1.1 CALL VERSTK

This routine initializes the plotter device specified in the JCL, and must be called before any other SMOG routine. It sets up the default plotting frame size to a 25cm square (psuedo 25cm for the Versatec), sets the default region to 1.0 (the problem coordinate system), and sets the character height to 0.25cms.

5.1.2 CALL REGION(x)

This routine defines the users frame of reference. The selected region prevails until a further call to REGION is made.

a) x=1.0. This is the default region. The system interprets all coordinate numbers specified by the user as being with respect to his own 'problem space'.

b) x=0.0. This causes the system to interpret coordinates as raster coordinates on the device mesh. Conversion to type 'integer' takes place and where possible actual mesh points are addressed.

5.1.3 CALL LIMIT(xmin,ymin,xmax,ymax)

CALL JLIMIT(ixmin,iymin,ixmax,iymax)

These routines define the coordinate system to be used over the whole plotting area. The point (xmin,ymin) and similarly (ixmin,iymin) identifies the lower left-handed corner of the plotting area, and (xmax,ymax) or (ixmax, iymax) identifies the upper right-hand corner. In region 0, that is, in raster coordinates, the minimum values are normally 1 and, for the Versatec and the Calcomp the maximum in both dimensions is 1024 and 2500 respectively.

5.1.4 CALL LIMITS(xmin,ymin,xmax,ymax,scalex,scaley)

CALL JLIMS(ixmin,iymin,ixmax,iymax,iscalex,iscaley)

These routines perform the same task as LIMIT and JLIMIT but, in addition, the scaling factors used in the x and y direction are stored in 'scalex' ('iscalex') and 'scaley' (or 'iscaley') respectively.

5.1.5 CALL LIMITV(xmin,ymin,xmax,ymax)

CALL JLIMV(ixmin,iymin,ixmax,iymax)

These routines perform the same function as LIMIT and JLIMIT on the Calcomp or Versatec plotter. They define the coordinate limits for the visible region rather than the whole plotting area, but, for the Calcomp and Versatec, these regions are the same. For other output devices, not available at Daresbury, the visible region is only a part of the whole plotting area.

5.1.6 CALL ABUT

This routine is used to extend the plotting area by increasing its length by 25cms (pseudo 25cms for the Versatec).

5.1.7 CALL ADVFLM

This routine advances the plotter to a new frame so that following output appears in a new plotting area.

5.1.8 CALL CHHT(height)

This routine sets the character height to 'height' centimetres. The default set by VERSTK is 0.25cms.

5.1.9 CALL CHOCT(angle)

This routine sets the angle (in radians measured from the horizontal) at which characters or text is to be drawn. The default is angle=0.0.

5.1.10 CALL CHSIZE(height)

This routine has the same effect as CHHT(see section 5.1.8).

5.1.11 CALL ENDSPR

This terminates the use of the plotter, and must be called after all plotting is complete.

## 5.2 Low-level Routines

This section includes single pen movements, the drawing of lines, and character and text output. The available routines are described within these categories.

### 5.2.1 Pen-movement

The following routines are used to reset the current pen position. No drawing takes place. If we consider the current pen position to be the point (X,Y) for 'real' routines, or (IX,IY) for the integer routines, then the effect of each routine is as shown on the next page.

| Real Routine | Integer Equivalent | New Position |
|---|---|---|
| SETX(x) | JSETX(ix) | (x,Y)or(ix,IY) |
| SETXY(x,y) | JSETXY(ix,iy) | (x,y)or(ix,iy) |
| SETY(y) | JSETY(iy) | (X,y)or(IX,iy) |
| UPDX(dx) | JUPDX(idx) | (X+dx,Y)or<br>(IX+idx,IY) |
| UPDXY(dx,dy) | JUPDXY(idx,idy) | (X+dx,Y+dy)or<br>(IX+idx,IY+idy) |
| UPDY(dy) | JUPDY(idy) | (X,Y+dy)or(IX,IY+idy) |

### 5.2.2 Line drawing

These routines cause a line to be drawn from either the current (old) position to a new position, or between two specified points.

a) From the current position (X,Y) or (IX,IY)

| Real Routine | Integer equivalent | New position |
|---|---|---|
| TOX(x) | JTOX(ix) | (x,Y)or(ix,IY) |
| TOXY(x,y) | JTOXY(ix,iy) | (x,y)or(ix,iy) |
| TOY(y) | JTOY(iy) | (X,y)or(IX,iy) |
| TODX(dx) | JTODX(idx) | (X+dx,Y)or(IX+idx,IY) |
| TODXY(dx,dy) | JTODXY(idx,idy) | (X+dx,Y+dy)or<br>(IX+idx,IY+idy) |
| TODY(dy) | JTODY(idy) | (X,Y+dy)or(IX,IY+idy) |

b) Between specified points.

| Routine | Line drawn |
|---|---|
| VEC(x1,y1,x2,y2) | (x1,y1)to(x2,y2) |
| JVEC(ix1,iy1,ix2,iy2) | (ix1,iy1)to(ix2,iy2) |

### 5.2.3 Character and text output

Routines to specify character height (CHHT,CHSIZE) and angle (CHOCT) are described in sections 5.1.8 to 5.1.10. Table 1 shows the default character set and codes. The use of a different larger character set is described in Chapter 7 of the Daresbury Laboratory Computer User's Guide. In addition to text string output it is often useful to represent numerical information in the form of characters distributed over the visible area. Routine for plotting a single character are described below followed by those for text string output.

#### 5.2.3.1 Single character output.

Three routines are available for single character output. For these, the concept of 'current plotting position' is still relevant as each routine will print one character so that its lower left-hand corner, or that of its defining or exscribed rectangle is the current pen position (XP,YP). The routines are:

CALL HPLOT(i)

where 'i' is the code number of the character to be plotted with its lower left-hand corner at (XP,YP). The only exception is the plotting point which is centred on (XP,YP).

CALL HPLOTS('s')

which plots the one-element string ''s'' located in the same way as HPLOT.

```
CALL PLOTPT
```

This routine cause the plotting point (a small circular dot) to be printed at the current pen position.

#### 5.2.3.2 Output of text strings

Four routines (HTEXT,HTYPE,HTYPES and TYPNMB) are available for character string output. They cause the current pen position (XP,YP) to be moved to the end of the last character plotted.

```
CALL HTEXT(num,text)
```

This routine writes 'num' characters from the text string 'text' starting at the current pen position (XP,YP). 'text' can be in the form of a hollerith string, e.g. 10HTEXTOUTPUT; enclosed in quotes, e.g. 'textoutput'; or a string identifier (characters are packed 4 to an INTEGER*4 or REAL*4, and 8 to a REAL*8 variable).

```
CALL HTYPE(i)
```

The SMOG character number 'i' is printed at (XP,YP) and (XP,YP) is incremented by one character space.

```
CALL HTYPES('s')
```

The single element string ''s'' is printed at (XP,YP) which is reset as for HTYPE.

```
CALL TYPNMB (x,a,b)
```

The number 'x' is printed as a character string starting from (XP,YP). The form in which the number is to be represented is indicated by the values 'a' and 'b' where

```
 a = 0.0  gives E - format, field width b+7
 b = 0.0  gives I - format, field width a+1
a,b not = 0.0  gives F - format, field width a+b+2
          with 'a' places and a sign before
          and 'b' places after the decimal
          point.
```

## 5.3 High-Level Routines

This section includes the drawing of curves, contours, axes, graphs, histograms and shapes and a routine for area shading.

### 5.3.1 Curve Drawing

The routines available for curve drawing are defined within the groups: arcs, circles, plotting of functions and ellipses.

#### 5.3.1.1 Arc drawing

Four routines are available for drawing arcs of a circle. These are:

```
CALL ARCA(x,y,rad,d,theta,phi,p)

CALL ARCB(x,y,rad,q,theta,phi,p)

CALL ARCR(rad,d,theta,phi,p)

CALL ARCS(rad,q,theta,phi,p)
```

where:

'rad' is the radius of the circle (in centimetres),
'd' is the maximum line segment length (in centimetres),
'q' specifies the number of line segments.

If p=0.0 all the lines are drawn, otherwise only every other line is drawn, giving a dashed arc.

ARCA and ARCB take the circle centre to be (x,y) and draw an arc from 'theta' to 'phi' where 'theta' and 'phi' are measured in radians from the positive x-axis (see figure 1).

ARCR and ARCS start at the current pen position (XP,YP), and, using (XP-rad * COS(theta),YP-rad * SIN(theta)) as the circle centre, draw an arc through 'phi' radians.

#### 5.3.1.2 Circle drawing

Four circle drawing routines are available. These are:

```
CALL CIRCLA(xx,yy,rad,d,p)

CALL CIRCLB(xx,yy,rad,q,p)

CALL CIRCLR(rad,d,p)

CALL CIRCLS(rad,q,p)
```

where:

'rad' is the circle radius (in centimetres),
'd' is the maximum line segment length (in centimetres),
'q' specifies the number of line segments
of equal lengths.

If p=0.0 all lines are drawn, otherwise only every other line is visible, giving a dashed circle.

CIRCLA and CIRCLB take (xx,yy) as the circle centre, whereas CIRCLR and CIRCLS use the currrent pen position as the centre. On exit the circle centre will be the current pen position.

The following errors are possible:

| | |
|---|---|
| rad < 0.0 | ABS(rad) is taken,or 1.0 if rad=0.0 |
| d < 0.001*rad | d = 0.1*rad, |
| q < 1.0 | ABS(q) is taken, or 8.0 if q = 0.0 |

### 5.3.1.3 Single-valued functions

Four routines are available for plotting a single-valued function y=f(x).

These are:

CALL CVFNCA (f,xmn,xmx,dx,dy)

CALL CVFNCB (f,xmn,xmx,q)

CALL CVFNCR (f,xmn,xmx,dx,dy)

CALL CVFNCS (f,xmn,xmx,q)

where:

'f' is the name of a function subroutine which
defines the function to be plotted.
'xmn' is the minimum value of the variable x.
'xmx' is the maximum value of the variable x.
'dx','dy' are the maximum values of the
x-increment and y-increment
respectively for each line segment.
'q' specifies the number of line segments
equally spaced in the x-direction,
which are to be used.

These routines draw the curve y=f(x) between the x values 'xmn' and 'xmx'. The function f(x) must be provided as a Fortran Function Segment with the name specified in 'f' and an argument x. The function subroutine must be declared as EXTERNAL in the calling routine.

CVFNCA and CVFNCB use the scaling factors set up in a call to one of the LIMIT routines to evaluate the absolute position of each function point. CVFNCR and CVFNCS start at the current pen-position and use relative values to determine the positions of each point.

The following errors can occur:

| | |
|---|---|
| xmx < xmn | curve drawn from xmx to xmn |
| xmx-xmn > 100*dx | dx = 0.1*(xmx-xmn) |
| dy < = 0 | dy = 1.0 |
| q < 0 | MOD(q) taken |
| q = 0 | q = 1.0 |

### 5.3.1.4 Ellipse drawing

A complete ellipse with axes parallel to the X and Y axes can be drawn using one of the following routines. These are:

CALL ELLPSA (xx,yy,a,b,dx,dy,p)

CALL ELLSPB (xx,yy,a,b,q,p)

CALL ELLSPR (a,b,dx,dy,p)

CALL ELLSPS (a,b,q,p).

where:

'xx','yy' is the centre
'a','b' are the major and minor axes lengths (cms.)
'dx','dy' are the maximum line segment
increments in x,y directions.

If p=0.0 all lines are drawn, otherwise only every other line is visible.

ELLPSA and ELLPSB select (xx,yy) as the centre, whereas ELLPSR and ELLPSS use the current plotting position. The centre will be the current plotting position on exit.

The possible errors are:

| | |
|---|---|
| a,b=0.0 | absolute value taken, or 1 if zero. |
| dx<0.001*a | dx = 0.1*a |
| dy<0.001*b | dy = 0.1*b |
| q< =0.0 | absolute value taken, or 8 if zero. |

### 5.3.2 Contours

Two sets of contouring routines are available in SMOG. The first contains two routines, CNTOUR and CNTOR1, which are simple to use but require data points defined on a square grid with a maximum of 50 points in each direction. The other set includes two contouring routines, CNTR2A and CNTR2B as well as an associated routine, CNTKEY for plotting a key to the contour heights. These routines allow the user to specify more options, such as contours of different thickness, an irregular grid and a choice of two contouring techniques.

### 5.3.2.1 CNTOUR and CNTOR1

The data points used by these routines must be defined on a square grid. The x and y coordinates of the grid are taken to range from (1,1) in the bottom left-hand corner to (n,n) at the top right-hand corner, where n is the number of grid points in each direction. That is:

```
          (n,1)   (n,2).....         (n,n)
            .       .                  .
            .       .                  .
            .       .                  .
          (2,1)   (2,2).....         (2,n)
  row 1:  (1,1)   (1,2).....         (1,n)
```

The routines are used in the following way:

```
CALL CNTOUR(xgrid,xn,ht,xn0,xmk)

CALL CNTOR1(xgrid,xn,ht,xn0,xmk,m)
```

where:

'xgrid' is a 1-D array containing the function values at each grid point, stored by rows.

'xn' is the number of grid points in each direction (<=50)

'xn0' is the number of contours required

'ht' is a 1-D array to contain the required contour values.

If 'xn0' > 0 contour values must be specified in 'ht'.

If 'xn0' < 0 equally spaced contours are calculated and these values written to 'ht'.

'xmk'= 1.0 indicates that dots are to be placed for reference at each grid point, otherwise 'xmk' should be set to zero.

CNTOUR draws contours to cover the whole grid, while CNTOR1 draws contours within the boundary specified by the integer array M as follows:

```
M(1)          = count of boundary points
M(2)          = not used,
M(I),M(I+1)   = x,y coordinates of boundary
                points in order.
```

NOTE: CNTOUR and CNTOR1 make calls to LIMIT therefore it is not necessary for the user to set up the coordinate system.

### 5.3.2.2 CNTR2A, CNTR2B, and CNTKEY

CNTR2A is the routine for producing a contour map of a surface for which the heights are given on a regular grid (the points are assumed to be one unit apart in the x and y directions starting at (0.0,0.0)). CNTR2B is the routine for producing a contour map of a surface for which the heights are given on an irregular grid (this is specified by means of the arrays XPTS and YPTS). It is assumed that if an irregular grid is being used then the points are closest together where most detail is needed and farthest apart where least detail is needed. Hence, for both CNTR2A and CNTR2B, each mesh rectangle is treated the same by the internal contouring routines and the x and y coordinates are only used to calculate the points to be plotted.

The routines are used in the following way:

```
CALL CNTR2A(rmesh,unused,id,m,n,height,nconts)

CALL CNTR2B(rmesh,unused,xpts,ypts,id,m,n,height,nconts)
```

The arguments of the routines are defined as follows:

'rmesh' is a two-dimensional REAL array of size ('id','n') which contains the heights at the grid points. These are arranged so that the first dimension increases as y increases and the second dimension increases as x increases.

'unused' is a two-dimensional LOGICAL array of the same size as 'rmesh' which is used by the routine as workspace.

'xpts','ypts' (CNTR2B only) are one-dimensional REAL arrays of size 'n' and 'm' respectively and contain the distances of successive points along each axis from the bottom left-hand corner of the grid ie 'xpts'(1)='ypts'(1)=0.0. If 'xpts'(1) or 'ypts'(1) is not zero then it will be treated as an additional offset from the origin but any skewing will then be about the point (OX,OY) (see below) and not the corner of the grid (normally these points are coincident).

'id' is the size of the first dimension of 'rmesh'. This is necessary in case 'm' and 'n' are read in. (As arrays must be declared explicitly in FORTRAN, this is needed for the subroutine to use the correct elements of 'rmesh' if only part of the array is used. Usually 'id' and 'm' will be the same.)

'm','n' are the number of points in the y and x directions respectively.

'height' is a one-dimensional REAL array of size 'nconts' and contains the contour heights if they are specified or is used to store them if they are calculated (this is the default option).

'nconts' is the number of contour heights.

CNTKEY is the routine for plotting a key to the contour heights (the contours are only labelled with the contour number) and is normally called after the call to CNTR2A or CNTR2B. The key is plotted within a rectangle (not drawn) of size 22 x character width by (n2-n1+8) x line space where line space is 5/3 x character height. The character height and width are set by CHSIZE in CNTR2A and CNTR2B (see RCH later) but may be altered before the call to CNTKEY if desired. It is used in the following way:

```
CALL CNTKEY(x,y,height,n1,n2)
```

The arguments of the routine are defined as follows:

'x','y' are the x,y coordinates of the top left-hand corner of the exscribed rectangle described above.

'height' is a one-dimensional REAL array containing the contour heights and should be the same array specified in the call to the contouring routine.

'n1','n2' are the numbers of the first and last contours to be included in the key. (Several keys may be plotted to include all the contours.) No check is made by the routine to see if the key will fit within the current visible area.

Various options concerning the production of the contour map are available. These are specified by assigning values to the thirteen variables in a named COMMON block (CNTOPT). Default values are

assigned to these variables by the routine VERSTK so that values only need be assigned to those variables corresponding to an option for which a value other than the default is required. Note that it is necessary to set non-default values after the call to VERSTK.

The common block CNTOPT takes the form:

```
COMMON/CNTOPT/ICON,ICH,OX,OY,RCOS,IBOX,IGR,ILAB,
                                   RFR,IT,IDF,RLV
```

The options and corresponding variables are outlined on the next page.

| VARIABLE | OPTIONS AVAILABLE | DEFAULT VALUE |
|---|---|---|
| ICON | Internal contouring routine CNTRH1 (ICON=1) or CNTRH2 (ICON=2) is used see section 5.3..2.3 | 2 |
| ICH | Contour heights are specified in array HEIGHT (ICH=1) or calculated by the routine (ICH=0). | 0 |
| OX,OY | Offset of the origin of the grid from the origin of the user's coordinates. | 0.0,0.0 |
| RCOS | Cosine of the angle (between 0 and 180 degrees between the two axes of the grid (ie the skew of the grid). | 0.0 |
| IBOX | Frame drawn round map (IBOX=1) or not (IBOX=0). | 1 |
| IGR | Grid points plotted (IGR=1) or not (IGR=0). | 0 |
| ILAB | Contours are labelled : none (ILAB=0), every one (ILAB=1) every Ith one (ILAB=I). | 1 |
| RFR | Frequency of labelling on contours (labels appear every ICON*RFR*MAX(M,N) steps, with a minimum of one label per contour unless ILAB=0. One label per contour is achieved with RFR set to a very large value or zero). | 1.0 |
| RCH | Height of characters used for labelling height is RCH*M/80 (CNTR2A) or RCH*(YPTS(M)-YPTS(1))/80 (CNTR2B). | 1.0 |
| IT | Contours drawn thicker (the number of hits is trebled so this has its best effect for a low number of hits originally): values as for ILAB. | 0 |
| IDF | Differentiate between contours above and below a certain level (IDF=1) or not (IDF=0). Contours below level drawn as broken lines. (This is most effective on large grids) | 0 |
| RLV | Is the level associated with IDF. | 0.0 |

The values of each of these parameters are output by the contouring routines on Channel 12. it is therefore necessary to allocate this channel by including a Job Control Statement of the form:

```
//G.FT12F001 DD SYSOUT=A
```

### 5.3.2.3 CNTRH1 and CNTRH2

These routines are not callable by the user but are briefly described here to enable the user to choose between them (see ICON in 5.3.2.2).

In both routines the basic method is to take each contour height in turn and trace any contours of this height through the mesh. In CNTRH1 inverse linear interpolation is used to find the points where the contour crosses the mesh lines and these points are joined by straight lines. Any ambiguity as to which way a contour turns is solved by assuming 'high ground on the right'. In CNTRH2 the surface height at the centre of each mesh square is approximated by the average of the heights at the four corners. In addition to the points where the contour crosses the mesh lines (as in CNTRH1), points where the contours cross the mesh diagonals are found using inverse linear interpolation. These points are again joined by straight lines. This technique, which effectively produces a triangular mesh, avoids any ambiguity as to which way a contour turns.

### 5.3.3 Axis and Graph drawing

It is possible to draw either a set of axes or a complete graph. Two routines AXESA and DRAXES are available for axes drawing. They are used in the following way:

```
CALL AXESA(x,y)
```

This draws axes through (x,y) reaching to the edges of the current region. The current pen position on exit will be (x,y).

```
CALL DRAXES(xmin,ymin,xmax,ymax,xl,yl,title,char)
```

where:

'xmin','ymin' are the minimum and maximum
'xmax','ymax' values of the axes span.
'xl','yl','title' specify the address of the first character of text arrays which are to label the x,y axes, and to used as the title.
'char' is a single character used to delimit the text arrays.

A set of axes is drawn in the specified position with labels and a title.

The routines available for graph drawing are DRPLOT and DREBAR. DRPLOT is used to draw a complete graph with labelled axes and a title. DREBAR has the addition facility to include error bars drawn for each data point. They are used in the following way:

```
CALL DRPLOT(x,y,n,xl,yl,title,char)

CALL DREBAR(x,y,erbar,n,xl,yl,title,char)
```

where:

'x','y'  are arrays containing the data points to be plotted.
'erbar'  is an array containing the size of the error bars to be drawn at each point.
'n'  is the number of points to be plotted.
'xl','yl','title','char' are as for DRAXES above.

### 5.3.4 Histograms

Four histogramming routines are available; DRSIST, for simple histograms, HSTGMA for shaded histograms, CMHSTA for shaded composite histograms, and DRHIST for labelled histograms. These are used in the following way:

```
CALL DRSIST(ylow,delta,y,n)
```

where:

'ylow'  is the lowest point at which horizontal bars are drawn.
'delta'  is the length of the bar.
'y'  is the array of heights to be plotted.
'n'  is the number of points in y.

A simple histogram is drawn with no vertical bars, and using a base line for y=0.0.

```
CALL HSTGMA (xmn,ymn,deltax,dyar
                      ymax,dx,dy,type,border)
```

where:

'xmn','ymn'  is the origin,
'deltax'  is interval in the x direction,
'dyar'  is an array containing the values to be plotted,

'ymax'  is the number of values to be plotted (note 'real' type),
'dx','dy' and 'type' describe shading (see TXTURA).

If 'border' =0.0 the border around the histogram does not include the base line, and if border = 1.0 the base line is included.

A simple shaded histogram is drawn.

```
CALL CMHSTA (xmn,ymn,deltax,dyar,ymax,cmax,dx,dy,
                                     type,border)
```

where:

'xmn', 'ymn'  is the origin
'deltax'  is the interval in the x-direction
'dyar'  is a 1-D array containing all the y values.
'ymax'  is the number of elements in each histogram level.
'cmax'  is the number of histograms to be plotted.
'dx','dy' and 'type' are arrays of length 'cmax' containing the values as defined in TXTURA for each level.
'border'  is as described for HSTGMA.

A composite shaded histogram is drawn.

```
CALL DRHIST(y,n,xl,yl,title,char)
```

where:

'y','n' are as described for DRSIST
and all other parameters are as described for DRAXES.

A labelled and titled histogram, of 'n' elements of the array 'y' is drawn.

### 5.3.5 Shapes

The simple shapes that can be drawn automatically are arrowheads and boxes. The routines available are:

```
CALL AROWHR(dx,dy)

CALL AROWVR(dx,dy)
```

Both routines draw an arrow head (V) at the current plotting position, where:

'dx','dy' are the sizes in the x and y direction respectively.

AROWHR draws a horizontal arrow (pointing left if 'dx'<0, and right is 'dx'>0). AROWVR draws a vertical arrow (up if 'dy'>0, and down if 'dy'<0).

```
CALL BOXA(x1,y1,x2,y2)

CALL BOXR(dx,dy)
```

BOXA constructs a rectangle with the bottom left-hand corner at (x1,y1) and the top right-hand corner at (x2,y2). The current plotting position, on exit, is (x1,y1). BOXR starts from the current plotting position and draws a rectangle with sides 'dx' and 'dy'.

### 5.3.6 Shading

One routine, TXTURA is available for shading a rectangular region. It is used in the following way:

```
CALL TXTURA(xmin,ymin,xmax,ymax,
                  dx,dy,type,border)
```

where:

'xmin'..'ymax' describe the rectangle limits.
'dx','dy' are increments, whose use depends on the value of type.

The pattern drawn is designated by type where:

type = 0.0 No pattern
= 1.0 Horizontal lines with interval 'dy',
= 2.0 Vertical lines with interval 'dx',
= 3.0 Both type = 1.0 and type = 2.0,
= 4.0 Diagonal lines angled to the horizontal at arctan('dy'/'dx') and 'dy' apart.
= 5.0 As for type = 4.0 but angled at PI - arctan('dy'/'dx').
= 6.0 Both type = 4.0 and type = 5.0.
= 7.0 Horizontal dashed lines, 'dx' in length 'dy' apart.
= 8.0 As type = 7.0 but with every alternate line displaced by 'dx'.
= 9.0 Vertical dashed lines, length 'dy' 'dx' apart.
=10.0 As type = 9.0 but with every alternate line displaced by 'dy'.

If border = 0.0 no edge or border is drawn.
If border = 1.0 a border around the area is drawn.

## 5.4 3-D Routines

The following routines provide a projection of a 3-D drawing or surface on to the 2-D plane of the output medium. The projection is in perspective from a 'view-point' (xv,yv,zv) situated 'd' units along a normal to the plane of projection (see figure 2).

### 5.4.1 CALL ZVIEW(xv,yv,zv,d)

This routine defines the location of the view-point (xv,yv,zv). The plane of projection is taken to be at a distance 'd' measured from the view point in the direction of the origin.

### 5.4.2 CALL SETXYZ(x,y,z)

CALL UPDXYZ(dx,dy,dz)

These routines cause pen movements (no drawing) which are obvious extensions of the equivalent 2-D routines (see 5.2.1).

### 5.4.3 CALL TODXYZ(dx,dy,dz)

CALL TOXYZ(x,y,z)

CALL VECZ(x1,y1,z1,x2,y2,z2)

These routines cause lines to be drawn and are equivalent to the 2-D routines TODXY, TOXY and VEC(see 5.2.2).

### 5.4.4 CALL EXPANZ(x,y,ex,ey,ez)

CALL ROTAXZ(x1,y1,z1,x2,y2,z2)

CALL ROTANZ(theta)

CALL ERDRZ(flag)

The routine EXPANZ expands the picture about the point (x,y,z) by the factors 'ex', 'ey' and 'ez' in the X,Y and Z-directions respectively. This has the same effect as a zoom lens.

ROTAXZ defines the axis of rotation to be the line joining the points (x1,y1,z1) and (x2,y2,z2).

ROTANZ defines the angle of rotation to be 'theta'.

ERDRZ is used if both rotation and expansion are required. If 'flag'=1.0 rotation is performed first, and if 'flag'=2.0 expansion is performed first.

### 5.4.5 CALL CUBER(dx,dy,dz)

This routine projects a cube drawn from the current pen position with sides parallel to each axis and lengths 'dx', 'dy' and 'dz'.

## 6. EXAMPLES

Three examples of the use of SMOG high level routines are given below. They illustrate the use of DRPLOT to draw a simple graph, CNTR2B to plot a contour map, and CMHSTA to plot a complex histogram.

### 6.1 DRPLOT-graph drawing routine.

The code shown in table 2 is a simple program to draw a graph of the function y=x**2 with 0<=x<=9. The axes labels and graph title are provided by the user. Figure 3 shows the plot produced by this program.

### 6.2 CNTR2B-contour plotting.

This example produces a contour plot of the function

```
   z=8*exp(-(x-2.3)**2+(y-2.3)**2)
            +8*exp(-(x-4.2)**2+(y-4.6)**2)
```

An irregular grid is used; the values taken by x and y are defined in data statements. The non-default values used for variables in the common block CNTOPT are as follows:

```
IGR=1   Grid points to be plotted.
ICH=1   Contours heights provided.
RCH=3   Character height is changed.
RCOS=1.0/2.0   Grid is skewed.
IT=2    Thickened contours.
IDF=1   Contours below RLV drawn as broken lines.
RLV=4.0   Value associated with IDF.
```

The contour key is drawn using the routine CNTKEY. Figure 4 shows the plot produced by the program given in table 3.

### 6.3 CMHSTA-complex histogram.

The program given in table 4 produces a composite histogram of the functions y1=x**2 and y2=x. The second histogram is drawn 'above' the first. Figure 5 shows the plotted histogram.

DARESBURY LABORATORY

SCIENCE RESEARCH COUNCIL

## ERRATUM TO DL/CSE/TM7

Please add as page 22:

## ACKNOWLEDGEMENTS

The SMOG system available on the Daresbury Laboratory IBM 370/165 is adapted from the original SMOG system(1) which was designed and implemented by members of the Atlas Computer Laboratory. New contouring routines(2) were added by D. C. Sutcliffe of the Rutherford Laboratory.

## REFERENCES

1. P. A. Dewar, 'The SMOG SYSTEM Manual', Atlas Computing Division, Rutherford Laboratory, 1976.

2. D. C. Sutcliffe, 'Graphics User Note 1 - Contouring', Atlas Computing Division, Rutherford Laboratory, 1976.

TABLE 1

Character set and code numbers

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | □ | 16 | \| | 32 | } | 48 | Σ | 64 | | 80 | & | 96 | - | 112 | 0 |
| 1 | ○ | 17 | | 33 | { | 49 | ÷ | 65 | A | 81 | J | 97 | / | 113 | 1 |
| 2 | △ | 18 | ∧ | 34 | μ | 50 | ≤ | 66 | B | 82 | K | 98 | S | 114 | 2 |
| 3 | + | 19 | ≡ | 35 | π | 51 | ≥ | 67 | C | 83 | L | 99 | T | 115 | 3 |
| 4 | × | 20 | → | 36 | φ | 52 | △ | 68 | D | 84 | M | 100 | U | 116 | 4 |
| 5 | ◇ | 21 | | 37 | θ | 53 | [ | 69 | E | 85 | N | 101 | V | 117 | 5 |
| 6 | ↑ | 22 | ≠ | 38 | ψ | 54 | ] | 70 | F | 86 | O | 102 | W | 118 | 6 |
| 7 | ⊼ | 23 | ± | 39 | χ | 55 | \ | 71 | G | 87 | P | 103 | X | 119 | 7 |
| 8 | Z | 24 | — | 40 | ω | 56 | ↑ | 72 | H | 88 | Q | 104 | Y | 120 | 8 |
| 9 | Y | 25 | _ | 41 | λ | 57 | √ | 73 | I | 89 | R | 105 | Z | 121 | 9 |
| 10 | ⌧ | 26 | ‾ | 42 | α | 58 | † | 74 | ¢ | 90 | ! | 106 | ∞ | 122 | : |
| 11 | ✳ | 27 | ∫ | 43 | δ | 59 | ‡ | 75 | . | 91 | $ | 107 | , | 123 | # |
| 12 | ⧗ | 28 | ⊃ | 44 | ∈ | 60 | ← | 76 | < | 92 | * | 108 | % | 124 | @ |
| 13 | \| | 29 | ∨ | 45 | η | 61 | × | 77 | ( | 93 | ) | 109 | — | 125 | ' |
| 14 | ☆ | 30 | ∼ | 46 | | 62 | ↑ | 78 | + | 94 | ; | 110 | > | 126 | = |
| 15 | – | 31 | ≈ | 47 | | 63 | ↓ | 79 | \| | 95 | ¬ | 111 | ? | 127 | " |

```
//PBSMOGDR JOB (0030,PB),PEARL,MSGLEVEL=(1,1),NOTIFY=PB,MSGCLASS=H
// EXEC FORTRAN,TIME.G=(,59),REGION.G=180K
//C.SYSIN DD *
      DIMENSION XARRAY(15),YARRAY(15)
      XARRAY(1)=0.0
      YARRAY(1)=0.0
      DO 10 I=2,10
      XARRAY(I)=XARRAY(I-1)+1.0
   10 YARRAY(I)=XARRAY(I)**2
      CALL VERSTK
      CALL LIMIT(0.0,0.0,12.0,110.0)
      CALL DRPLOT(XARRAY,YARRAY,10,
     1'X-LIMITS&','Y-LIMITS&','TEST DRPLOT&','&')
      CALL ENDSPR
      STOP
      END
//L.SYSLIB DD
//          DD DSN=SMOG.LOAD,DISP=SHR
//G.VERSATEC  DD SYSOUT=(V,VPLOT)
//
```

```
//PBSMOGC  JOB  (0030,PB),BRERETN,MSGLEVEL=(1,1),MSGCLASS=A,
//     NOTIFY=PB
//RUN  EXEC FORTRAN,TIME.G=(0,59),REGION.G=180K
//C.SYSIN  DD  *
      COMMON /CNTOPT/ICON,ICH,OX,OY,RCOS,IBOX,IGR,ILAB,RFR,
     1RCH,IT,IDF,RLV
      DIMENSION D(52,50),H(10),XPS(50),YPS(52)
      LOGICAL UNUSED(52,50)
      DATA H/0.5,1.5,2.5,3.5,4.5,5.5,6.5,7.5,0.0,0.0/
      DATA XPS/0.0,0.4,0.7,0.9,1.1,1.3,1.5,1.6,1.7,1.8,1.9,2.0,2.1,2.2,
     1         2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0.3.1,3.2,3.3,3.4,3.5,3.6,
     2         3.7,3.8,3.9,4.0,4.1,4.2,4.3,4.4,4.5,4.6,4.7,4.8,4.9,5.0,
     3         5.2,5.4,5.6,5.8,6.0,6.2,6.5,6.9/
      DATA YPS/0.0,0.4,0.7,0.9,1.1,1.3,1.5,1.6,1.7,1.8,1.9,2.0,2.1,2.2,
     1         2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0,3.1,3.2,3.3,3.4,3.5,3.6,
     2         3.7,3.8,3.9,4.0.4.1,4.2,4.3,4.4,4.5,4.6,4.7,4.8,4.9,5.0,
     3         5.1,5.2,5.3,5.4,5.6,5.8,6.0,6.2,6.5,6.9/
      CALL VERSTK
      IGR=1
      ICH=1
      RCH=3.0
      RCOS=1.0/2.0
      IT=2
      IDF=1
      RLV=4.0
      CALL LIMIT(0.0,0.0,10.0,6.9)
      DO 10 J=I,52
      Y=YPS(J)
      DO 10 I=1,50
      X=XPS(I)
      Z1=8.0*EXP(-((X-2.3)*(X-2.3)+(Y-2.3)*(Y-2.3)))
      Z2=8.0*EXP(-((X-4.2)*(X-4.2)+(Y-4.6)*(Y-4.6)))
   10 D(J,I)=Z1+Z2
      CALL CNTR2B(D,UNUSED,XPS,YPS,52,52,50,H,8)
      CALL CNTKEY(0.0,7.0,H,1,8)
      CALL ENDSPR
      STOP
      END
/*
//L.SYSLIB  DD
//          DD  DSN=SMOG.LOAD,DISP=SHR
//G.FT12F001 DD SYSOUT=A
//G.VERSATEC  DD  SYSOUT=(V,VPLOT)
//
```

## TABLE 4

```
//PBSMOGCM JOB (0030,PB),PEARL,MSGLEVEL=(1,1),NOTIFY=PB,MSGCLASS=H
// EXEC FORTRAN,TIME.G=(,59),REGION.G=180K
//C.SYSIN DD *
      DIMENSION XARRAY(15),YARRAY(20),DX(2),DY(2),TYPE(2)
      X=0.0
      Y=0.0
      DX(1)=1.0
      DX(2)=0.0
      DY(1)=4.0
      DY(2)=0.0
      TYPE(1)=6.0
      TYPE(2)=0.0
      XARRAY(1)=1.0
      YARRAY(1)=1.0
      DO 10 I=2,10
      XARRAY(I)=XARRAY(I-1)+1.0
   10 YARRAY(I)=XARRAY(I)**2
      DO 20 I=11,20
   20 YARRAY(I)=I-10
      CALL VERSTK
      CALL LIMIT(0.0,0.0,12.0,120.0)
      CALL DRAXES(0.0,0.0,10.0,120.0,
     1'X-LIMITS&','Y-LIMITS&','TEST CMHSTA&','&')
      CALL CMHSTA(X,Y,1.0,YARRAY,10.0,2.0,
     1DX,DY,TYPE,1.0)
      CALL ENDSPR
      STOP
      END
//L.SYSLIB DD
//         DD DSN=SMOG.LOAD,DISP=SHR
//G.VERSATEC  DD SYSOUT=(V,VPLOT)
//
```

Fig.1 The arc drawing routines take the circle centre at (x,y) and draw an arc from 'theta' to 'phi' radius measured from the positive x-axis.

Fig.2 A projection of a 3-D object onto a 2-D plane output medium.

Fig.3 Plot produced by the program shown in table 2.

Fig.4 Plot produced by the program shown in table 3.

Fig.5 Plut produced by the program shown in table 4.

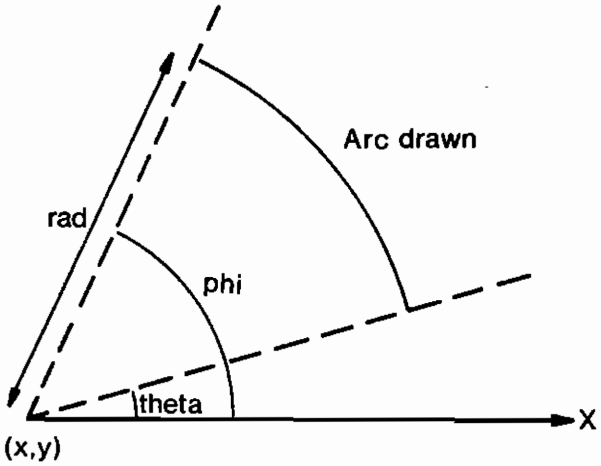
Arc drawn
rad
phi
theta
X
(x,y)

Fig.1

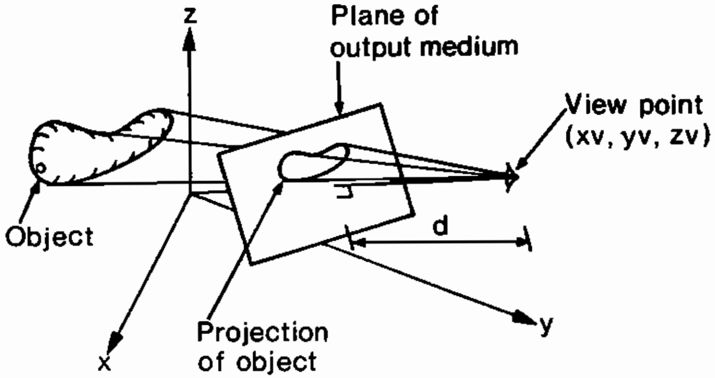
z
Plane of
output medium
View point
(xv, yv, zv)
Object
d
Projection
of object
x
y

Fig. 2

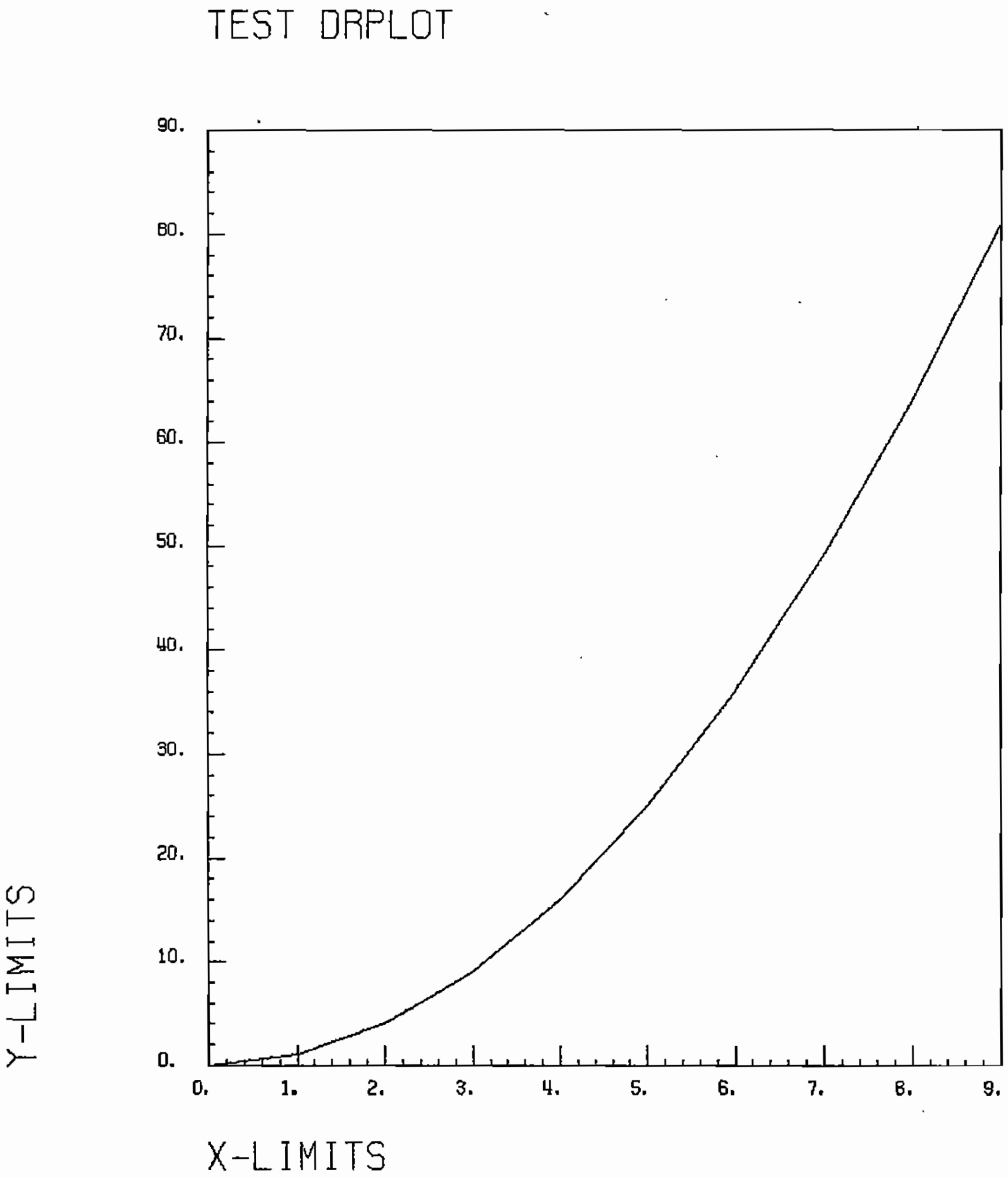
TEST DRPLOT
90.
80.
70.
60.
50.
40.
30.
20.
10.
0.
0.
1.
2.
3.
4.
5.
6.
7.
8.
9.
Y-LIMITS
X-LIMITS

Fig. 3

CONTOUR KEY

| C.NO | CONTOUR HEIGHT |
|---|---|
| 1 | 0.5000000E-00 |
| 2 | 0.1500000E+01 |
| 3 | 0.2500000E+01 |
| 4 | 0.3500000E+01 |
| 5 | 0.4500000E+01 |
| 6 | 0.5500000E+01 |
| 7 | 0.6500000E+01 |
| 8 | 0.7500000E+01 |

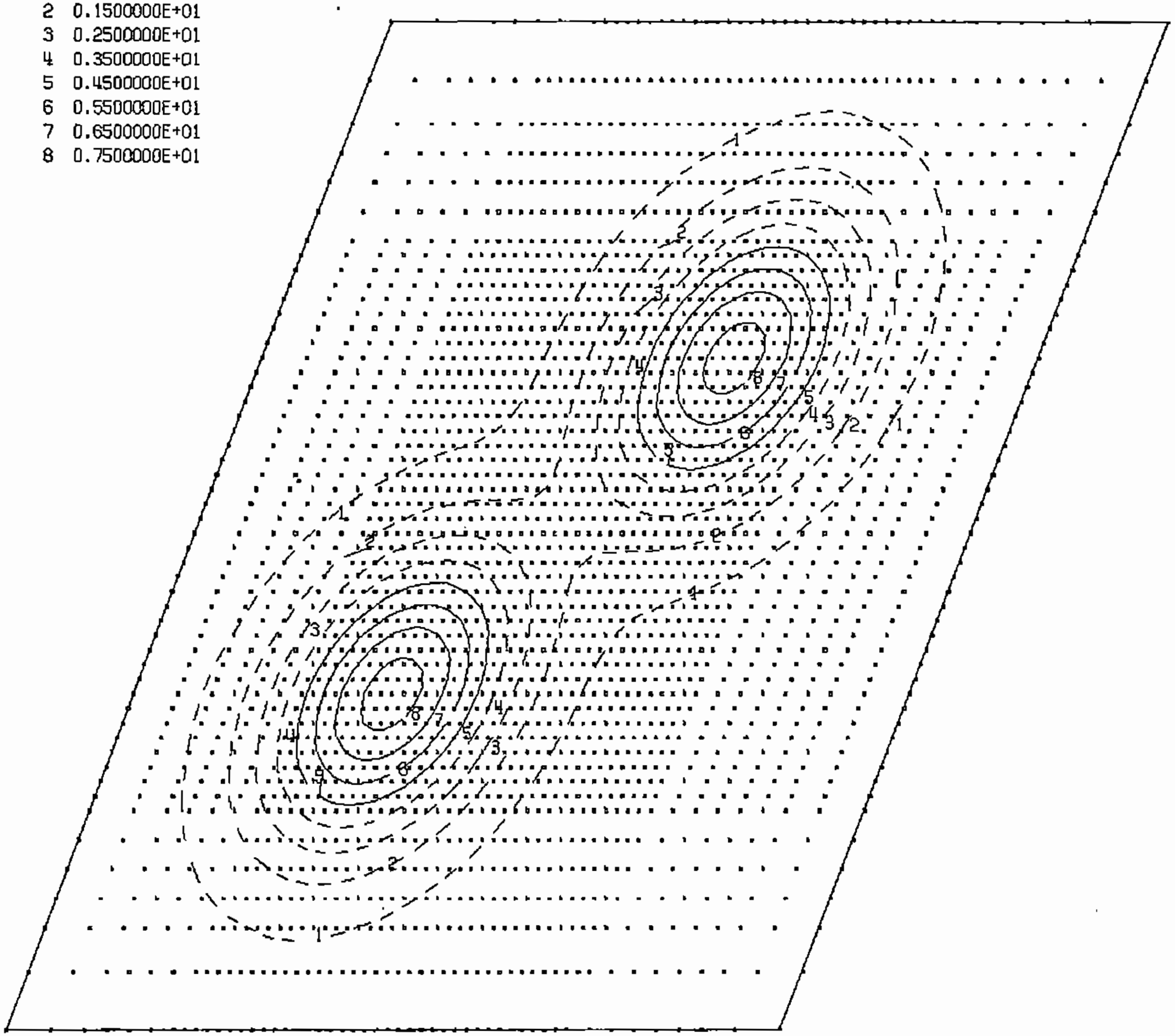

Fig. 4

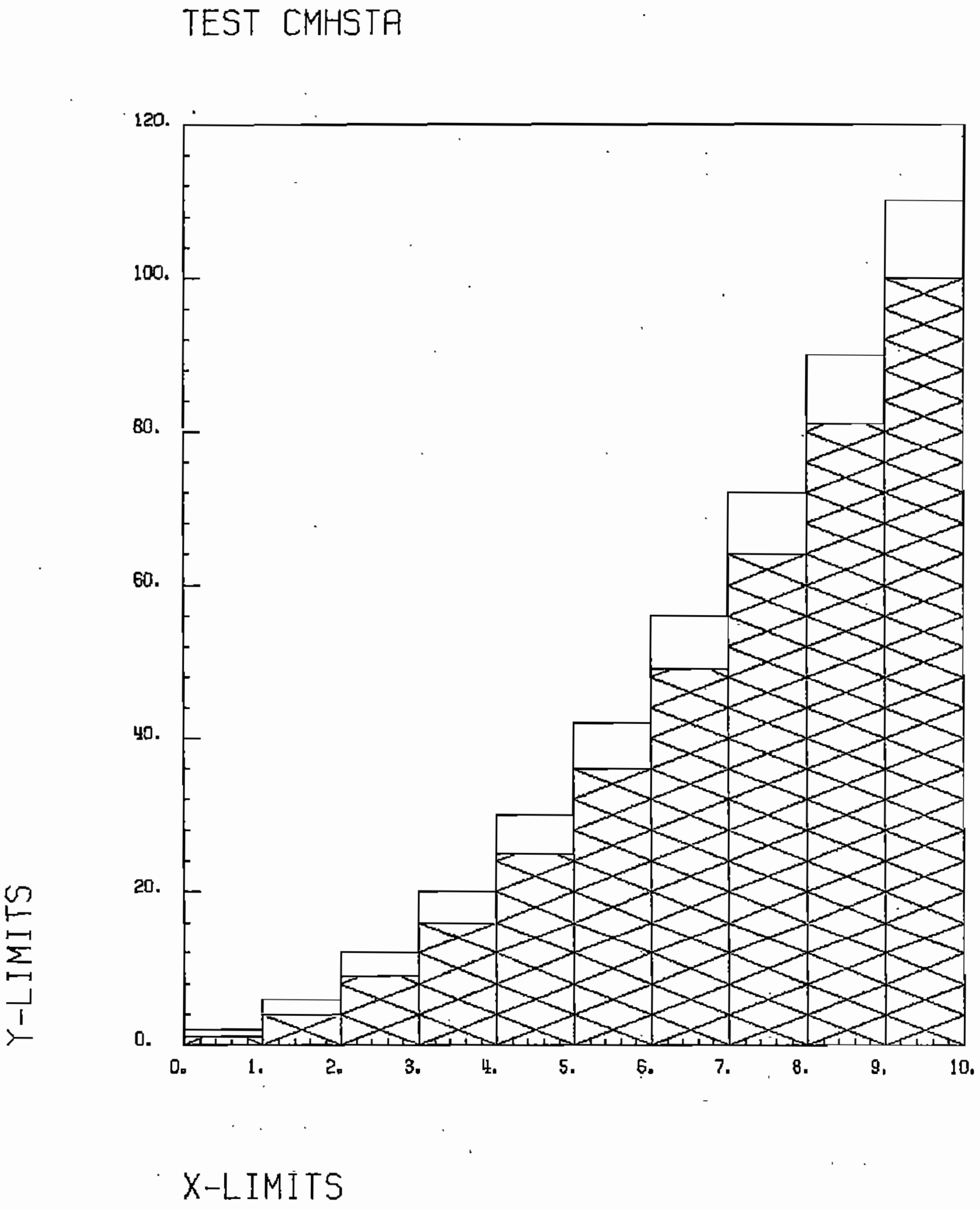
TEST CMHSTA
120.
100.
80.
60.
40.
20.
0.
0.
1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
Y-LIMITS
X-LIMITS

Fig. 5

INDEX PAGE

INDEX PAGE