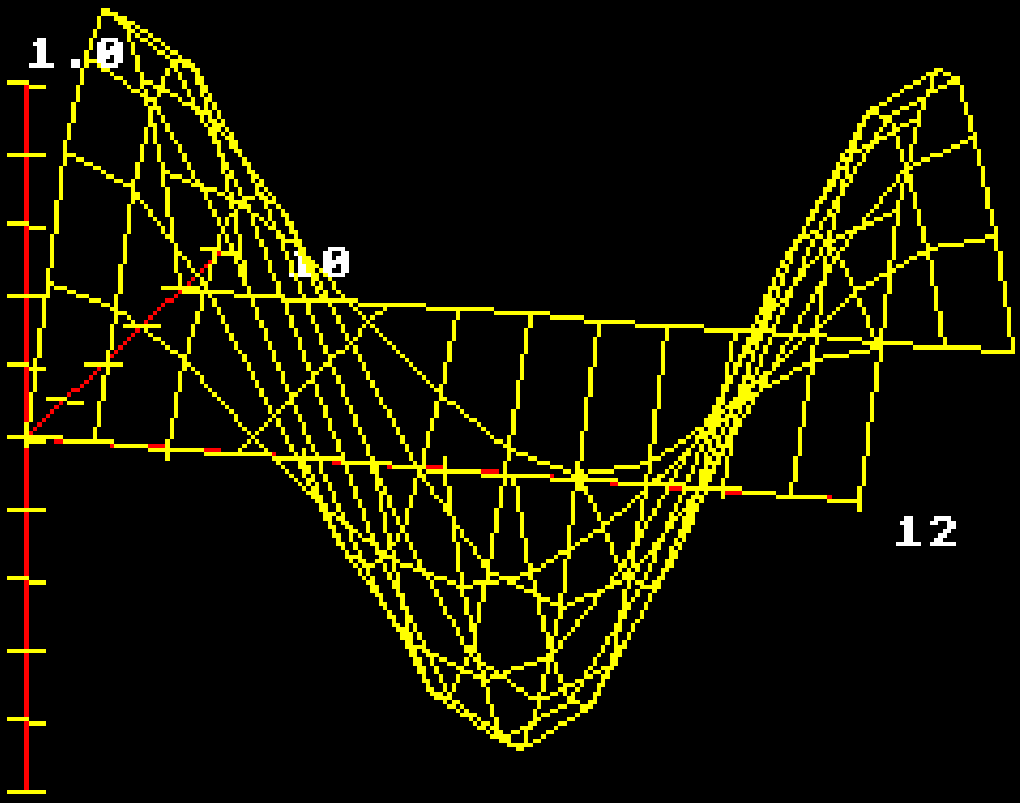
Graphs and Charts

on the BBC Microcomputer

ROBERT D. HARDING



Graphs and Charts

on the BBC Microcomputer

ROBERT D.HARDING

ACORNSOFT

Acknowledgement

This book and the Graphs and Charts Pack software are based on the graphics package written by the author for the project 'Computer Aided Teaching of Applied Mathematics' (CATAM) at the Department of Applied Mathematics, Cambridge University, England.

FIRST EDITION

ISBN 0 907876 04 8

Published by:

Acornsoft Limited
4a Market Hill
Cambridge
CB2 3NJ
England

A cassette of the software described in this book is available from Acornsoft.

DIGITALLY REMASTERED ON RISC OS COMPUTERS, NOVEMBER 2011.

Contents

	Index of Programs and Procedures	
	Summary of Loading Procedures	
1	Introduction	3
2	Level Three Programs	5
3	Level Two Procedures	11
4	Level One Procedures	33
5	Complete Listings	57

Summary of Loading Procedures

L3-BAR		LI-BAR	(Ll-2D)
L3-GRA		LI-SEC	(Ll-2D)
L3-CUR		L2-HIS	(Ll-2DY
L3-PIE		L2-PIE	(Ll-2D)
L3-CV3D		L2-XY	(Ll-2D)
L3-CO2D		L2-XYZ	(Ll-3D)
L3-CO3D		L2-C2	(Ll-2D)
L3-SURF		L2-C3	(Ll-3D)
Ll-2D		L2-SURF	(Ll-3D)
Ll-3D		L2-STER	(Ll-3D)
Ll-CNTR	(Ll-2D or -3D)	LF-STSU	(Ll-3D)
Ll-BOX	(Ll-2D or -3D)		

Level 3 programs (prefixed L3-) can be loaded and run in the usual way. For example, to load and run the program L3-BAR, enter

```
CHAIN "L3-BAR"
```

Press RETURN, and then press PLAY on the cassette recorder.

Programs in Levels 1 and 2 are designed to be incorporated into your own programs, and so are in EXEC format, All Level 2 programs and some Level 1 programs require either Ll-2D or Ll-3D, the fundamental plotting programs, to be loaded also - the appropriate program name appears in parentheses next to these program names in the listing above.

To load L2-HIS, for example, you would enter

```
*EXEC "L2-HIS"
```

and press RETURN. Press the PLAY button on the cassette recorder. The program is listed on the screen during loading. Once loaded, enter

```
*EXEC "Ll-2D"
```

and press RETURN. Press PLAY. on the cassette recorder and wait for this program to be loaded in also.

At this point you should refer to the appropriate chapter for details of running the program.

Index of Programs and Procedures

Name	Function	Page Description	Page Listing
L3-BAR	Bar Chart	6	57
L3-GRA	X-Y Graph	6	61
L3-CUR	Function Plot	7	66
L3-PIE	Pie Chart	7	70
L3-CV3D	3-D Function Plot	7	74
L3-CO2D	2-D Contour Map	8	79
L3-CO3D	3-D Contour Map	8	83
L3-SURF	Surface Plot	9	87
L2-HIS	Histogram	13	93
L2-PIE	Pie Chart	16	93
L2-XY	2-D Graph	18	94
L2-XYZ	3-D Graph	21	94
L2-C2	2-D Contour Map	24	94
L2-C3	3-D Contour Map	26	95
L2-SURF	Surface Plot	28	96
L2-STER	Stereo Graph	30	96
L2-STSU	Stereo Surface	32	96
Ll-2D	Fundamental 2-D Plot	44	98
Ll-3D	Fundamental 3-D Plot	46	100
LI-CNTR	Contour Map	48	103
LI-BOX	Box Outline	51	104
LI-BAR	Histogram Bar	52	104
LI-SEC	Pie Chart Sector	54	104

Introduction

1.1 The Graphs and Charts Pack

This book describes a system of program modules that can be used to create a wide range of graphs and charts in a professional way for presenting data supplied by the user. The program modules will be referred to as the Graphs and Charts Pack, and have been designed to run on the BBC Microcomputer Model B. A listing of the component modules is given at the end of this book. A cassette is also available.

The Graphs and Charts Pack makes full use of the powerful and versatile graphics facilities of the BBC Microcomputer. It is nevertheless constructed in such a way that relative newcomers to computers and programming will be able to use it with ease.

You can get a general impression of what you can do with the Graphs and Charts Pack by glancing through the book to look at the examples. Notice that the programs are divided into three main sections: Level Three, Level Two, and Level One.

It is the provision of levels that makes the Graphs and Charts Pack easy to use, yet allows the BBC Microcomputer's graphics to be used to the full.

Level Two contains procedures and functions which you can use as part of your own program. They are designed so that apart from setting data, users make only one FROG call in their progam in order to obtain a picture. The data can be in any units that the user pleases , there is no need to use the screen coordinates understood by the BASIC statements PLOT, MOVE or DRAW. At this level the package is therefore very easy to use.

Level One also consists of procedures and functions that you can use in your own program, but they give you much more freedom to achieve special effects. As in Level Two, users can work in any units that they please.

Using the Level One and Level Two routines it is a simple matter to build up programs for the interactive creation from the keyboard of graphs, charts, or plots of mathematical functions. The Level Three section of the book describes what such a set of programs might be, and these are included in the Graphs and Charts Pack.

Levels One and Two are fully compatible with the screen control facilities provided by the BBC Microcomputer - the user is free to continue to use these alongside the routines.

The examples presented in the manual can be used directly on the BBC Microcomputer Model B, after loading the appropriate Level One and Level Two routines.

1.2 Loading the Programs from Cassette

Level Three programs, as explained above, are complete and ready to use, so they are provided on the cassette as BASIC programs in normal 'SAVE/LOAD' form. For example, to run the bar chart program, enter:

```
CHAIN "L3-BAR"
```

in the usual way and start the cassette recorder playing.

This simple method cannot be used for Levels Two and One, however, because LOAD and CHAIN delete any program in the computer's store before bringing in the new one. As the purpose of Levels Two and One is to form part of your program, another way must be used. This is provided by the '*EXEC' command: this causes programs on the tape to be transmitted to the computer rather as if you were typing them in yourself quickly. Each line of input starting with a line number is then accepted by the interpreter as a new line to be added to your program. As a consequence, the program code from the tape will appear on the screen as it is read in. For example:

```
> *EXEC "Ll-2D"
```

You will also need to know which line numbers the Graphs and Charts Pack routines will use. They are all in the range 10000 to 20000. More precise information is given later.

1.3 Typing the Programs from Listings

To avoid any possible confusion between the lower case 'l' and the numeral '1', no lower case 'l's have been used in variable or procedure names, except in 'lo' and 'l%'.

2 Level Three Programs

2.1 Introduction

These utility programs are supplied on cassette complete and ready to run. To use them, you do not need to read any more than this section of the manual, although they should serve as a good introduction to the sort of thing you can do using the lower-level programs. To load the programs, use the CHAIN or LOAD command in the usual way.

The best way to find out how to use these programs is to run one, but the following brief summary may help.

On starting up, two or three pages of control data are displayed a page at a time. The first page is to do with which mode is to be used for the eventual display, colours to be used, etc. At the foot of each page a prompt line gives you the chance to make changes to the data on that page (type C) or you can go on (press SPACE). If you ask to make changes, the page is displayed again, pausing after each line to allow you to set new values. If the old value is acceptable, type only a RETURN, otherwise type the new value and end it with RETURN. You can go on doing this until you are satisfied, then type RETURN by itself.

The last page says that the program is ready to start drawing the graphs (or whatever is appropriate) and gives a list of keys that you will be able to use to control the program. After you next press the SPACE bar, the screen is cleared, axes are usually drawn, and the program waits for you to hit one of the control keys. To help you remember which they are, the ? key will give a list of them (without explanations) which lasts for 5 seconds. Each control key puts up a prompt on the bottom line, which you must answer with one or more data items.

For example, in the bar chart utility L3-BAR, the SPACE bar prompts for data to draw the next bar as follows :

```
X,Y ?
```

to which you could reply, for example, '5,8', to put the next bar at horizontal position 5 with height 8.

Certain control keys apply to all the utilities. C allows you to select a logical colour (see the User Guide), L allows the logical colours to be redefined in terms of physical colours, and T asks for a title and a position on the screen to put it. This position

is always expected in the units used to draw the display.

When your graph is complete or you want to abandon the program, escape will clear the screen and ask you if you want to repeat or stop.

2.2 List of Level Three Utilities

L3-BAR

Draws bar charts (histograms). The user defines each bar's position individually. It is possible to change the width, colour and offset of each bar, and to put a title on the chart.

Keys:

C: colour
L: change logical colours
B: bar base
O: offset
W: width
T: title
?: list of prompt keys

L3-GRA

Draws graphs (X-Y plots). The user may specify each point individually, or define a function which will be evaluated for a specified range of the X-axis and plotted, or any combination of these. The graph may be titled, colours changed, and if a function is used this may be written on the screen at any desired point. The function must be typed in as a valid BASIC expression in X.

Keys :

M: move to X,Y
D: draw to X,Y
C: colour
L: change logical colours
F: input function(X)
P: range to plot, no. of points
U: display function at X,Y
T: title
?: list of prompt keys

L3-CUR

Draws graphs like L3-GRA, but for function plotting allows X and Y to be defined as separate functions of a parameter T. Note that by setting the function X(T) to be just T, the program can be made to behave just like L3-GRA. Functions must be typed as valid BASIC expressions in terms of the variable T only.

Keys:

M: move to X,Y
D: draw to X,Y
C: colour
L: change logical colours
X: input function X(T)
Y: input function Y(T)
U: display X(T) at X,Y
V: display Y(T) at X,Y
P: T-range to plot, no. of points
T: title

L3-PIE

Draws pie charts. The user specifies the size of each sector in %. The starting angle of each sector is chosen automatically, but this can be over-ridden if required. Colours may be varied, and each sector labelled with its size in %.

Keys:

S: starting angle for next sector
D: draw next sector, size in %
C: colour
L: change logical colours
T: title
W: write sector size in colour 0 of the last sector drawn.
?: display list of prompt keys

L3-CV3D

Draws perspective graphs, otherwise like L3-CUR. Three functions X(T),Y(T), and Z(T) must be set, and/or individual points may be given.

Keys:

A: change viewing angles
M: move to X,Y,Z
D: draw to X,Y,Z
C: colour
L: change logical colours
X: input function X(T)
Y: input function Y(T)
Z: input function Z(T)
P: T-range to plot, no.
T: title
U: print X(T) at X,Y,Z
V: print Y (T)
W: print Z (T)
?: list of prompts

L3-CO2D

Draws contour maps on 'flat' axes. A function F(X,Y) needs T be set, and it must be a valid expression in BASIC involving X and Y only. The key P (for Plot) will ask for the range of contour heights that are to be drawn.

Keys :

C: colour
L: change logical colours
F: input function(X,Y)
P: plot for range of heights
T: title at X,Y
U: print fn at X,Y
?: print all recognised prompts

L3-CO3D

Draws contour maps in perspective. This program behaves in most respects like L3-CO2D except that perspective plots are produced. Because of the size of the program only modes 4 or 5 may be used.

Keys:

A: change view angles
M: move to X,Y,Z
D: draw to X,Y,Z
C: colour
L: change logical colours
F: input function F(X,Y)
P: plot contours
T: title

U: print F(X,Y) at X,Y,Z
?: list of prompts

L3-SURF

Draws a wireframe view of a function of two variables. This is an alternative method to L3-CO 3D of investigating such functions.

Keys:

A: change view angles
M: move to X,Y,Z
D: draw to X,Y,Z
C: colour
L: change logical colours
F: input function F(X,Y)
P: plot
T: title
U: print F(X,Y) at X,Y,Z
?: list of prompts

2.3 Using Level Three Utilities

The programs are large and use display mode 5 by default as this allows more memory for program than modes 1 or 2. However many of the programs can be successfully run in other modes.

Also concerning memory space, the contouring functions use an array, fW(I,J] to store function values. This is dimensioned as fW(10,10) by default, but the user may change this by modifying line 20 of the program to set IM%, JM% to other values.

If ESC is pressed, followed by a request to repeat program, any functions that were previously set preserved, but other values are not.

Note that by using the C key to draw in the background colour (0 by default) it is possible to selectively erase part of a graph or chart by drawing or plotting again over the part to be deleted.

3 Level Two Procedures

3.1 Introduction

These routines are intended for incorporation into the user's own BASIC program. They are 'minimum fuss' routines: in your part of the program you will have perhaps to dimension certain arrays and set values in them, and then call just one procedure to get graphical displays. In exchange for ease of use, you have to accept certain choices that will be made automatically by the routines. If these need to be changed, you will need to use Level One procedures.

3.2 Implementation Restrictions

All Level Two routines are accessed by FROG calls as detailed later. All global variables used by the procedures are prefixed with f, a convention which should prevent any clash of names with the user's own program. In fact the only such global variables are the graphics parameters which will be listed later, the variables f0, f1, f2, and certain arrays which the user is sometimes required to set in order to pass data. Obviously you should not use these variables in your part of the program except as specified.

Procedure and function names are also prefixed with E, with the exceptions of FNmin and FNmax. All line numbers used are in the range 10000 to 17999, but of course they may be renumbered with the RENUMBER command.

The routines assume that the full screen is available for graphics. Only the graphics cursor is moved, and only the foreground graphics colour is changed. VDU and other cursor control commands may be freely used in the user's program.

3.3 Loading the Routines

As explained in the introduction of the manual, the routines are supplied on cassette in a form that can be read using the *EXEC command. This is done so that the routines can be added to-an existing program. In addition to the Level Two tile that you are going to use, one of the fundamental Level One files L1-2D or L1-3D must also be loaded (using *EXEC). The detailed specifications tell you which.

3.4 Errors

If, during a run, the BASIC system were to report errors within the routines provided, this would usually be due to a fail are by the routines to calculate a suitable scale factor for the data you have provided. For example, in L2-XY, suppose you set all the values fY(I) to the same value, then the routines will give up because the range of Y-values is zero.

3.5 List of Level Two Procedures

L2-HIS Complete Bar Chart Plotter

PROCfHIS(M,N,W) - Requires Ll-2D

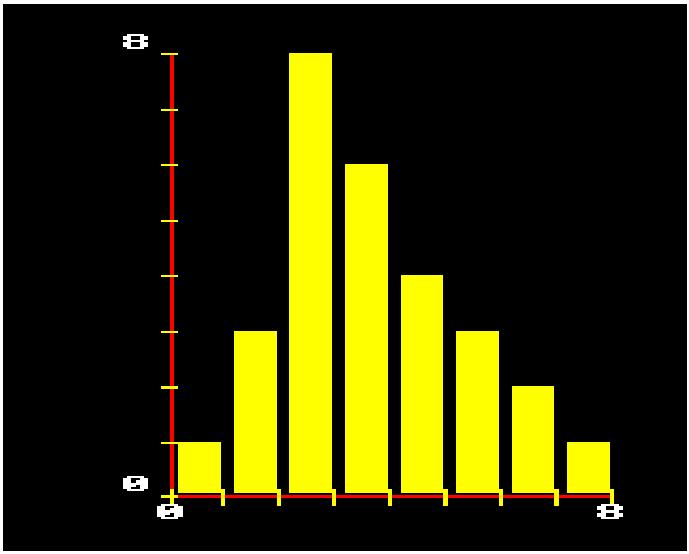
If you have a program which has calculated some data, or read it in from a data file, or whatever, such as monthly sales figures, then you can use this procedure to create a bar chart (also known as a histogram).

Your program must first store the data in two arrays fX(I) and fY(I), which you must previously dimension according to what you need. fY(I) will be the height of bar number I; the height may be in any units you like, and I must be in the range O to N. Note that this means that N+l bars will be drawn. fX(I) will specify the horizontal position of the bar, for example month in the range I to l2. You can store the data in any order, it doesn't have to be in numerical order, but each fX(I) and fY(I) must correspond.

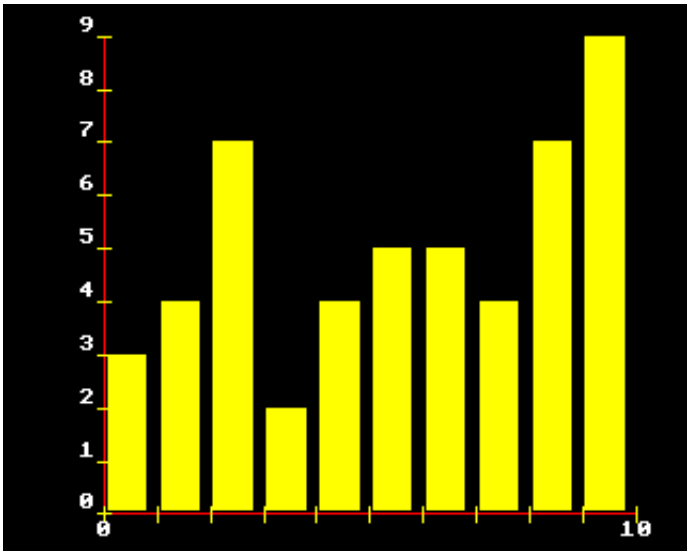
To draw the chart, call PROCfHIS (M,N,W), where M is the mode you want to use, N is the maximum for the index I (= one less than number of bars), and W is the width of each bar in your own units.

You may call the routine again with M set greater than l0 and other values set as above to a new set of data. The routine will repeat its actions except that it will assume axes are already set up.

Example Programs



```
 10 REM L2-HIX1
 30 REM
 40 REM  dimension arrays for data
 50 DIM fX(10),fY(10)
 60 REM  now ask for data
 70 INPUT "How many bars",N
 75 REM N-1 will be max subscript
 80 FOR I=0 TO N-1
 90 PRINT "Bar  ";I+1;
 95 INPUT " height",H
100 fX(I)=I+.1:fY(I)=H
110 NEXT
120 REM
130 REM ----- now call PROCfHIS ----
140 REM with bar width .7
150 MODE 5: PROCfHIS(5,N,.7)
160 END
```




```
400 REM L2-HIX2
420 REM
430 REM  generate 30 random no.s
440 REM  and count frequency.
450 DIM £X(10),£Y(10)
460 FOR I%= 1 TO 50
470 J%=10*RND(1): £Y(J%)=£Y(J%)+1:NEXT
480 REM generate vector of X-values
490 FOR I%=0 TO 9:£X(I%)=I%:NEXT
500 REM
510 REM ----- now call PROC£HIS ----
520 MODE1:PROC£HIS(1,9,.7)
530 END
540 REM ----------------------------
```

L2-PIE Ccmplete Pie Chart Plotter

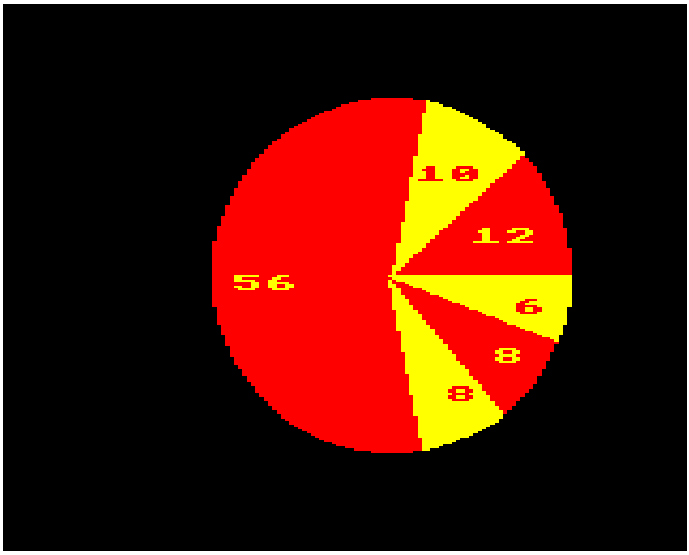
PROCfPIE (M,N) - Requires Ll-2D

Pie charts are useful if you want to show an analysis of a total figure, for example to show the contribution of various departments to the earnings of the parent organisation as a whole. This procedure would be useful if you had a program that could calculate these figures and you then wanted to display them.

Your program must first store the data in the array fY(I), which you must dimension previously according to what you need. fY(I) will be the size of sector number I as a percentage of a full circle. I must be in the range 0 to N. Note that this means that N+l sectors will be drawn. The data must be stored in the order in which the sectors are to be drawn. The first sector will start from the 3 o'clock position and drawing is anti-clockwise from there. There is no need for the fY(I) values to add up to l00%; if that is what you want you must ensure that your part of the program makes them do so.

To draw the chart, call PROCfPIE(M,N) where M is the mode you want to use, N is the maximum for the index I (= one less than number of sectors j.

Example Program




```
 10 REM L2-PIX1
 30 REM
 40 REM  dimension array for data
 50 DIM fY(20)
 60 REM  now ask for data
 70 INPUT "How many sectors ",N
 80 REM N-1 will be max subscript
 90 PRINT "Input sector sizes in %"
100 FOR I=0 TO N-1
110 PRINT "Sector ";I+1;
120 INPUT " size ",fY(I)
130 NEXT
140 REM
150 REM ----- now call PROCfPIE ---
160 MODE5:PROCfPIE(5,N-1)
170 END
```

L2-XY Complete Graph/Chart Plotter

PROCfXY(M,N) - Requires L1-2D

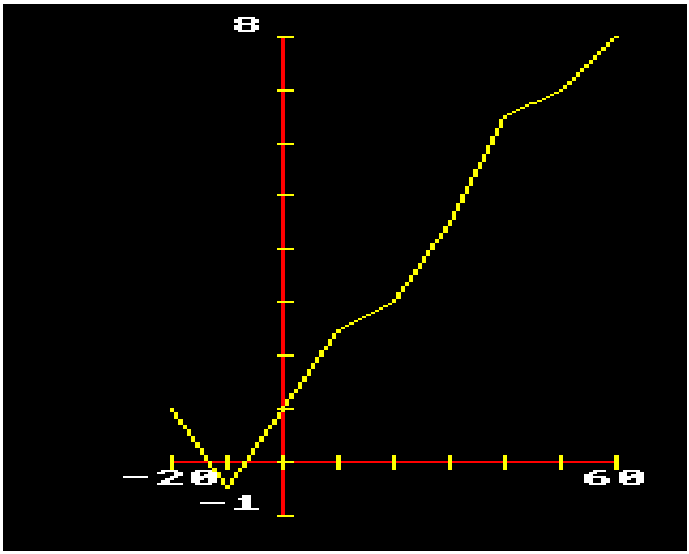
This procedure produces the most familiar form of graph, as used for example to show stockmarket indices, money supply, daily temperatures, etc. In most of these examples, time (e.g. days) runs left to right across the page, and usually a line (an 'axis') is drawn with intervals marked off along it. The quantity being graphed (e.g. temperature) is measured in an up and down direction and usually an axis is drawn for this too, marked off, for example, in centigrade. These axes are of ten call ed the horizontal, or X-axis, and vertical, or Y-axis. Any position on the graph may now be specified by giving two coordinates (X,Y). Successive points are joined to produce the graph. Graphs are also useful for representing mathematical functions, for example y= sin(x). By working out y for sufficiently closely spaced values of x, a smooth curve will appear to be drawn.

Assuming that your program can calculate the coordinates (X,Y) for each point to be joined up on the graph, all that the routine requires is that you store these values successively in the arrays fX(I), fY(I) where I take values from 0 to N inclusive. Thus there will be N+l points on the graph and N lines between them. The arrays must have been dimensioned in your part of the program.

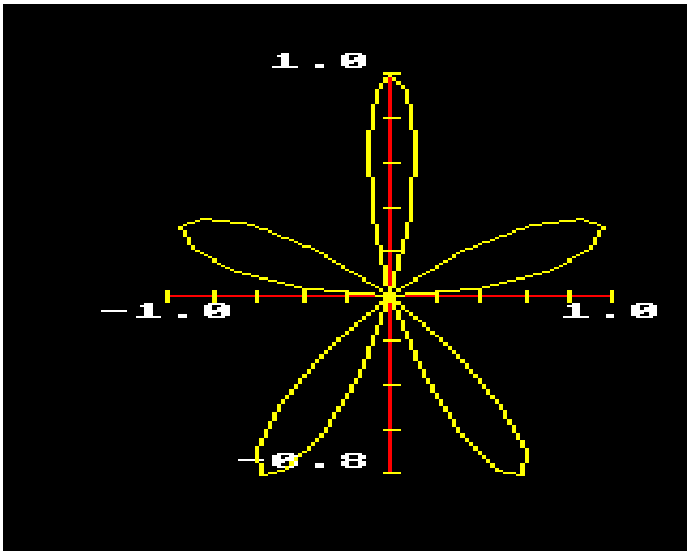
To draw the graph, call PR0CfXY(M,N), where M is the mode you want to use, N is the maximum for the index I (= one less than number of points, = number of lines joining them).

If having called PROCfXY as above you want to draw more graphs on the same axes, you can do this by storing the new data in fX(I) and fY(I) and then calling PROCfXY(M,N) again but with M greater than l0.

Example Programs




```
 10 REM L2-XYX1
 30 REM
 40 REM Input coordinates of some
 50 REM points and join them up.
 60 REM
 70 INPUT "How many points to join ",N
 80 REM Dimension arrays for coords.
 90 DIM fX(10),fY(10)
100 FOR I=0 TO N-1 : REM N points
110 PRINT "Point ";I+1;
120 INPUT "  X,Y ",fX(I),fY(I)
130 NEXT
140 REM
150 REM ---- now call L2-XY to draw--
160 MODE5:PROCfXY(5,N-1)
170 REM ---- that's all ----
180 END
```




```
500 REM L2-XYX2
520 REM
530 REM --- simple x,y plot ---
540 REM first DIM fX,fY & set values
550 DIM fX(40),fY(40)
560 N%=40
570 FOR I%=0 TO N%: th=I%*PI/N%
580 r = SIN(5*th)
590 fX(I%)= r *COS(th)
600 fY(I%)= r *SIN(th)
610 NEXT
620 REM --- now call PROCfXY ---
630 MODE5:PROCfXY(5,N%)
640 END
650 REM --- that's all ---
```

L2-XYZ Complete 3-Dimensional Graph Plotter

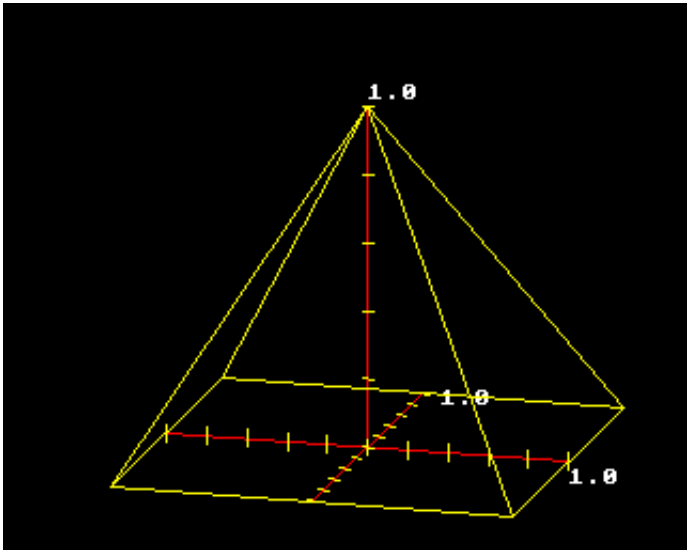
PR0Cf XYZ(M,N) - Requires Ll-3D

The complete 2-dimensional graph plotter L2-XY used values that the user had stored in fX(I), fY(I) to craw a 'flat' graph. This procedure extends the idea to 3 dimensions. As the computer screen is only 2-dimensional, the best that can be done is to produce a perspective view of the graph. J

Your program should dimension the arrays fX(I), fY(I) and fZ(I), and store in them the coordinates of the points to be joined up into the graph. As for 'z-dimensional graphs, mathematical curves can be made to look quite smooth by taking the points close enough together. Then call PROCfXYZ(M,N), where M is the mode to be used, N is the maximum subscript of the arrays.

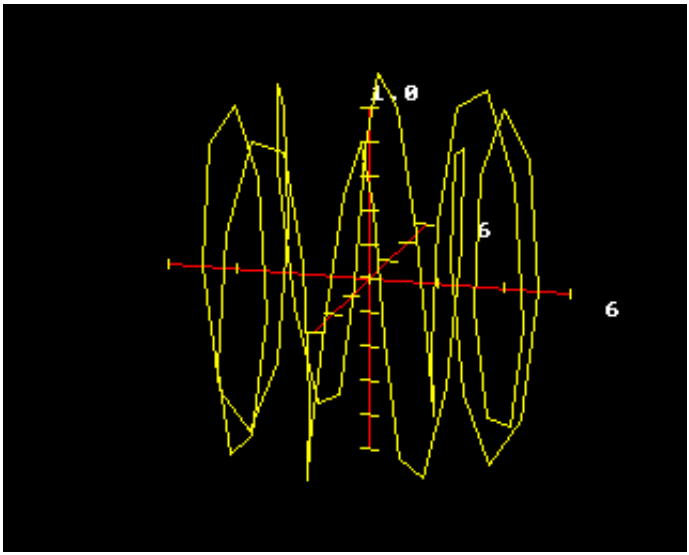
If having called PROCfXYZ as above you want to draw more graphs on the same axes, you can do this by storing the new data in fX(I), fY(I) and fZ(I), and then calling PROCF-XY (M, N) again but with M greater than l0.

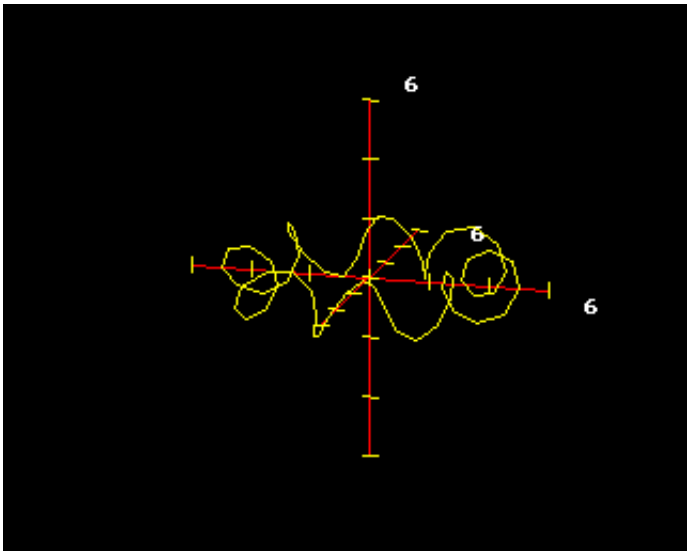
Example Programs




```
10 REM L2-XYZ1
40 REM
50 REM  dimension arrays
```

```
 60 DIM fX(50),fY(50),fZ(50)
 70 REM
 80 REM --- ask for data ---
 90 INPUT "No. of points ",N
100 FOR I=0 TO N-1
110 PRINT"Point ";I;
120 INPUT "  X,Y,Z= ",fX(I),fY(I),fZ(I)
130 NEXT
140 REM
150 REM --- now call L2-XYZ ---
160 MODE1:PROCfXYZ(1,N-1)
170 END
```






```
200 REM L2-XYZ2
240 REM
250 REM  dimension arrays
260 DIM £X(60),£Y(60),£Z(60)
270 REM
280 REM calculate data
290 N=60: dt=2*PI/N
300 FOR I=0 TO N
310 t=I*dt: r=4+COS(8*t)
320 £X(I)=r*COS(t)
330 £Y(I)=r*SIN(t)
340 £Z(I)=SIN(8*t)
350 NEXT
360 REM
370 REM --- now call L2-XYZ ---
380 MODE1:PROC£XYZ(1,N)
390 t=INKEY(300): MODE1
400  REM repeat, forcing axes ranges
410 £ZL=-5:£ZH=5:PROC£AX3(0)
420 GCOL0,2:PROC£XYZ(11,N)
430 END
```

L2-C2 Complete Contour Map Plotter

PROC£CN2D(M,IM,JM,N) - Requires Ll-2D

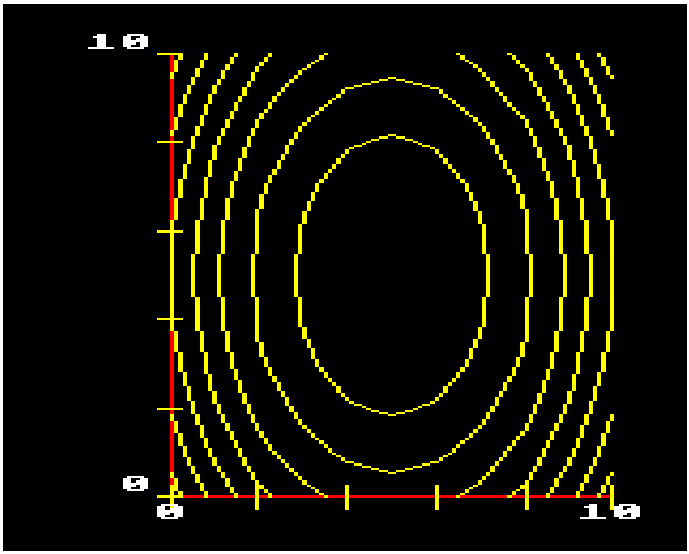
Contour maps are a useful way of representing functions of two variables. On a graph with X and Y-axes for example one might have a function of X and Y which represented the height of a hill; that is, for each point (X,Y) the height could be calculated from some function f(X,Y), Contours are the curves along which the function value (height) is constant, just as on Ordnance Survey maps.

To draw a contour map, you must first work out the function at a number of regularly spaced points. Suppose for example that you wanted to map the region X between 0 and 100, Y between -10 and 10. You must decide how to space out the points; one possibility is to take all the points with X-coordinates 0, 10, 20, 30, ... 100, and Y--coordinates -10, -8, -6, ... 0, 2, 4, ... 10. Every X-coordinate could pair up with any Y-coordinate, so you get 11 x 11 = 121 points altogether. These are called 'mesh points' or 'grid points'. The values of your function must be stored in a 2-dimensional array £W(I,J) where I counts the points along the X-axis, J counts along the Y-axis, numbering from O upwards. Let IM be the maximum value taken by I, and JM the maximum value for J. In our example IM=JM=10, but they might be different in general. Your program must dimension £W , of course.

To draw the map, call PROC£CN2D(M,IM,JM,N), where M is the mode you want to use, IM and JM are as defined above, and N is the number of contours that you want drawn. These will be at evenly spaced heights throughout the range found in the values in £W(I,J).

You may call the routine again with M set greater than 10 and other values set as above to a new set of data. The routine will repeat its actions except that it will assume axes are already set up.

Example Program




```
 10 REM L2-C2X1
 30 DIM £W(10,10)
 40 FOR I%=0 TO 10:FOR J%=0 TO 10
 50 X=(I%-5):Y=(J%-5)
 60 £W(I%,J%)=X^2+Y^2/2
 70 NEXT:NEXT
 80 REM now call PLOT£CN2D
 90 MODE5:PROC£CN2D(5,10,10,10)
100 END
```

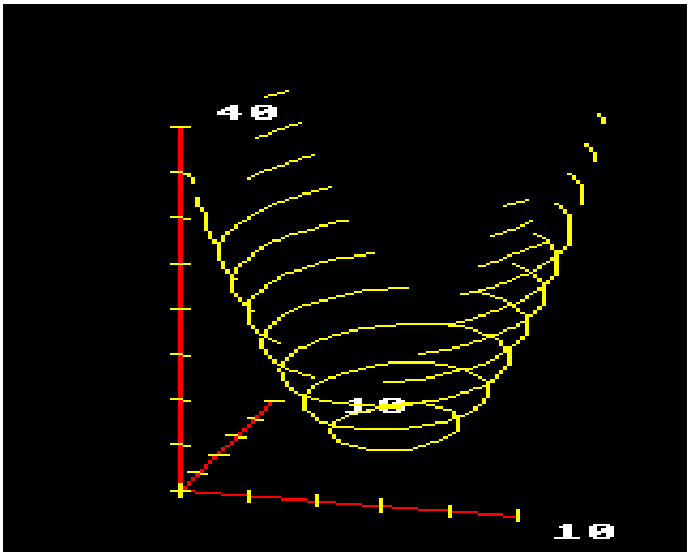
L2-C3 Complete Perspective Contour Map Plotter

PROCfCN3D(M,IM,JM,N) - Requires Ll-3D

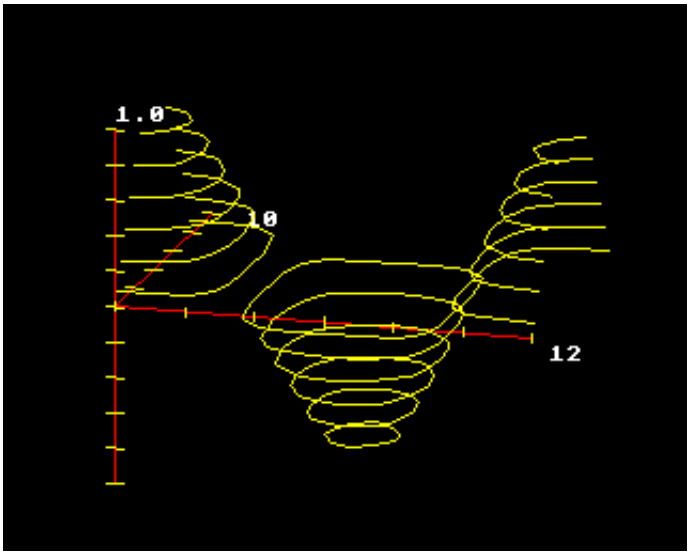
This procedure works in the same way as L2-CN2D except that the contours are drawn in perspective view according to height.

The array fW(I,J) must be dimensioned and set exactly as for L2-CN2D, then the call PR0CfCN3D(M,IM,JM,N) will draw the contours.

Example Programs




```
 10 REM L2-C3Xl
 30 DIM fW(10,10)
 40 FOR I%=0 TO 10:FOR J%=0 TO 10
 50 X=(I%-5):Y=(J%-5)
 60 fW(I%,J%)=X^2+Y^2/2
 70 NEXT:NEXT
 80 REM now call PROCfCN3D
 90 MODE5:PROCfCN3D(5,10,10,10)
100 END
```




```
200 REM L2-C3X2
230 DIM £W(12,8)
240 REM
250 REM set w = cos(x).sin(y)
260 FOR J=0 TO 8: B=SIN(J*PI/8)
270 FOR I=0 TO 12
280 £W(I,J)=B*COS(I*PI/6)
290 NEXT
300 NEXT
310 REM
320 REM now call PLOT£CN3D (12 contrs)
330 MODE1:PROC£CN3D(1,12,8,12)
340 END
```

L2-SURF Ccmplete Surface Plotter

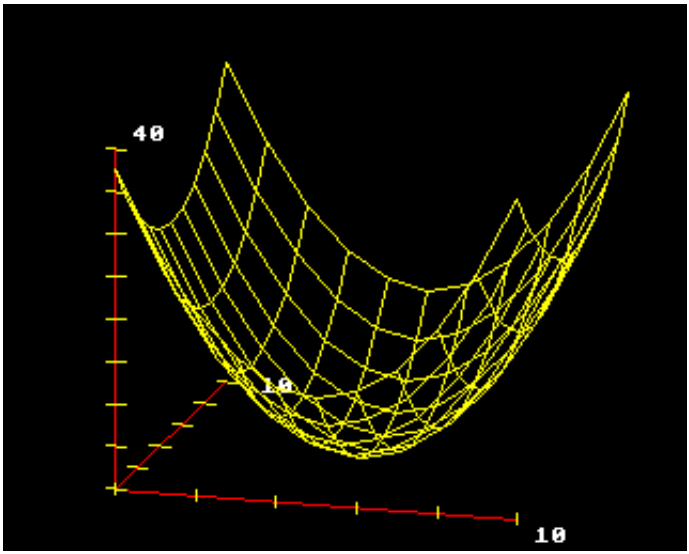
PROCfSURF(M,IM,JM) - Requires Ll-3D

The procedure L2-CN2D provides one way of representing functions of two variables by drawing 'flat' contours, and L2-CN3D enables those contours to be drawn in perspective. This procedure gives an additional method: imagine that all the points on a hill at the 'grid points' (see L2-CN2D) were joined to their neighbours by wires. If the hill were then taken away, you would be able to look at the wire frame left behind. L2-SURF draws this wire frame in perspective, which gives quite a good idea of the underlying surface.

To use the routine, set values in fW(I,J) just as for L2-CN2D, and then call PROCE-SURF (M,IM,JM). M is the mode to be used, IM, JM are the maximum values of I,J.

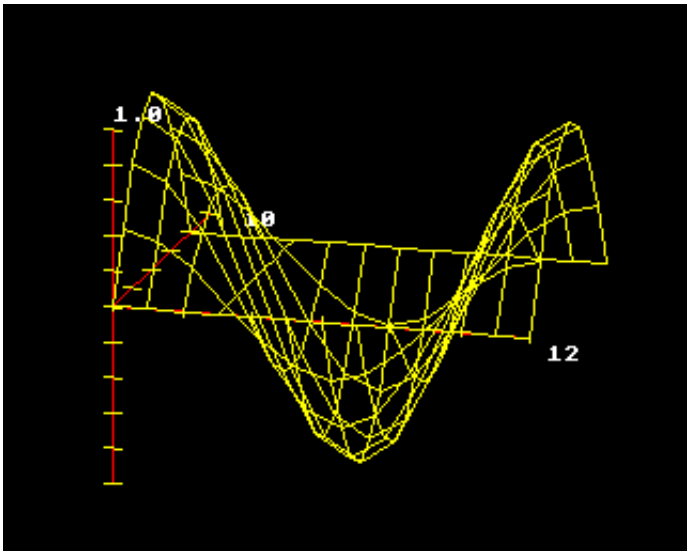
You may call the routine again with M set greater than l0 and other values set as above to a new set of data. The routine will repeat its actions except that it will assume axes are already set up.

Example Programs




```
10 REM L2-SUXl
40 REM
50 REM first dimension fW(I,J)
```

```
 60 DIM £W(10,10)
 70 REM
 80 REM now calculate values
 90 FOR I%=0 TO 10:FOR J%=0 TO 10
100 X=(I%-5):Y=(J%-5)
110 £W(I%,J%)=X^2+Y^2/2
120 NEXT:NEXT
130 REM
140 REM now call PROC£SURF
150 MODE1:PROC£SURF(1,10,10)
160 END
```




```
200 REM L2-SUX2
230 REM
240 REM first dimension £W(I,J)
250 DIM £W(12,8)
260 REM
270 REM set w = cos(x).sin(y)
280 FOR J=0 TO 8: B=SIN(J*PI/8)
290 FOR I=0 TO 12
300 £W(I,J)=B*COS(I*PI/6)
310 NEXT
320 NEXT
330 REM
340 REM now call PROC£SURF
350 MODE1:PROC£SURF(1,12,8)
360 END
```

L2-STER Stereographic Pair Plotter

PROC£STEREO(M,N) - Requires L1-3D

This procedure is like L2-XYZ but instead of a single graph it gives a pair in different colours viewed from slightly different angles. Because good resolution is needed, MODE 1 is recommended.

To use, set £X(I), £Y(I) and £Z(I) as for L2-XYZ, and then call PRCC£STEREO(M,N) where M is the nude to use and N is the maximum value of 1.

If called with M greater than 10, the axis ranges and other relevant parameters are assumed to be already set up, either by a previous call to PROC£STEREO by the technique described on page 46.

Example Program

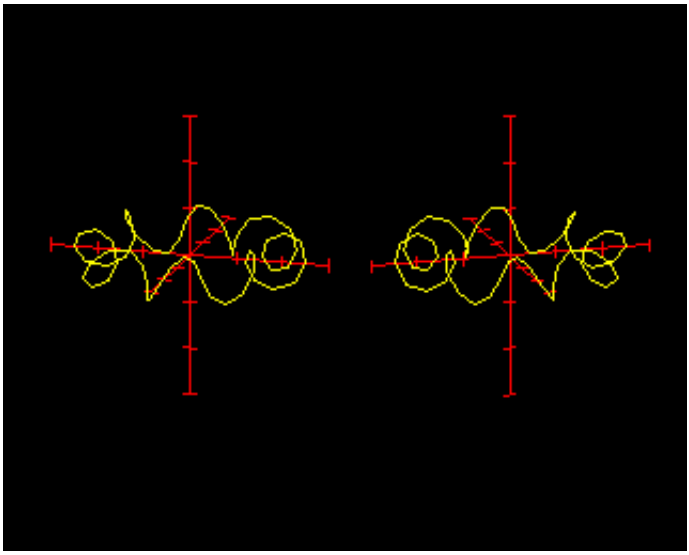
```
200 REM L2-STX1
240 REM
250 REM  dimension arrays
260 DIM £X(60),£Y(60),£Z(60)
270 REM
280 REM calculate data
290 N=60: dt=2*PI/N
300 FOR I=0 TO N
310 t=I*dt: r=4+COS(8*t)
320 £X(I)=r*COS(t)
330 £Y(I)=r*SIN(t)
340 £Z(I)=SIN(8*t)
350 NEXT
360 REM
370 REM --- now call L2-STEREO ---
380 MODE1:PROC£STEREO(1,N)
390 t=INKEY(300): MODE1
400  REM repeat, forcing axes ranges
410 £ZL=-5:£ZH=5
420 PROC£STEREO(11,N)
430 END
```

An alternative technique to the two-colour pair is to use two views side-by-side, one a mirror image of the other. Shown below are suitable modifications to the example and to L2-STER itself. To view, hold a mirror between the two views so the reflective face is vertical, is at right angles to the screen, and faces right. Position your head so that the mirror is mid-way between your eyes. Look at the left view; your right eye should see the reflection of the right-hand view on top of the left-hand view. Adjust your position until the images coincide.

Modifications

```
  380 MODE 1: PROCfINIT(1)
  390 fXL=-6: fXH=6:fYL=-6:fYH=6
  400 fZL=-5: fZH=5
  410 PROCfSTEREO(11,N)
  420 END

17290 DEFPROCfSTEREO(I%,N%):LOCAL S
17420 fXB%=0:fYB%=200: fXS%=700:fYS%=700
17430 fHM%=0: fVM%=0: fXC%=0: fPH=-1.3
17440 S=1:GOSUB 17470
17450 fXB%=600: fPH=-1.29:S=-1
17460 GOSUB 17470:ENDPROC
17470 GCOL 0,1:PROCfAX3(1):fP11=S*fP11
17475 fP12=S*fP12:PROCfAX3(2):GCOL 0,2
```



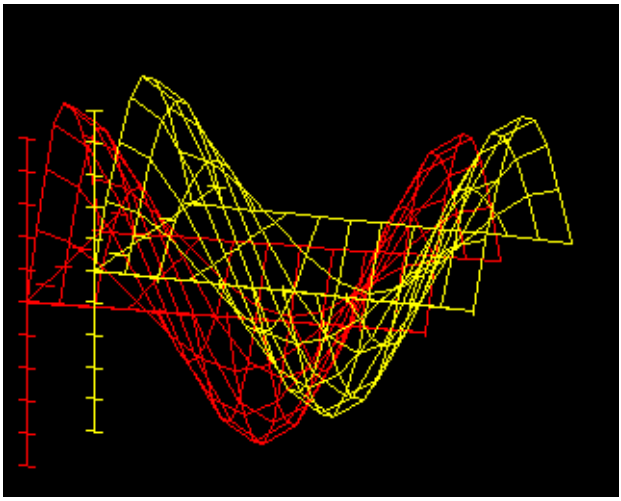
L2-STSU Stereoscopic Surface Plotter

PROCfSTSURF(M,IM,JM) - Requires Ll-3D

This procedure is like L2-SURF but instead of a single graph it gives a pair in different colours viewed from slightly different angles. Mode l is recommended, and if M is greater than l0 the same remarks apply as for L2-STER (page 30]. To use, set fW(I,J) as in L2-S0RF, and then call PRCEESTSURF (M,IM,JM) where M is mode to use, IM and JM are maximum values I and J.

Side-by-side views can obtained with modifications similar to those suggested for L2-SFER (see gages 30-3l).

Example Program



```
200 REM L2-SSXl
230 REM
240 REM first dimension fW(I,J)
250 DIM fW(l2,8)
260 REM
270 REM set w = cos(x).sin(y)
280 FOR J=0 TO 8: B=SIN(J*PI/8)
290   FOR I=0 TO l2
300   fW(I,J)=B*COS(I*PI/6)
310   NEXT
320 NEXT
330 REM
340 REM now call PROCfSTSURF
350 MODEl:PROCfSTSURF(l,l2,8)
360 END
```

4 Level One Procedures

4.1 Introduction

Level One routines are intended for incorporation into the user's own BASIC program. They offer the same kind of facilities as the Level Two routines but. with greater flexibility. In fact the Level One routines are the building blocks from which the Level Two routines are constructed.

Greater flexibility for the user than available from Level Two is provided because the user can vary the order in which routines are obeyed and it is possible to position a graph in any sector of the screen, and to vary the number of scale marks on an axis.

4.2 Implementation Restrictions

All Level One routines are accessed by PROC or FN calls as detailed later. All global variables used by the procedures are prefixed by E, a convention which should prevent any clash of names with the user's own program. In fact the only such global variables are the graphics parameters which are listed later, the variables f0, f1, f2, and certain arrays which the user is sometimes required to set in order to pass data.

Procedure and function names are also prefixed with f, with the exceptions FNmin and FNmax. All line numbers used are in the range 10000 to 15999, but of course they may be renumbered with the RENUMBER command.

The default graphics parameters (see below) assume that the full screen is available for graphics, but the user may change this. The only interaction between Level One routines and the screen control facilities of the BBC microcomputer (using the VDU statement) is that the graphics foreground colour is changed and the graphics cursor is moved. All lettering is done by linking the text and graphics cursors but this link is always switched off before returning. The user is therefore free to use PLOT, VDU, etc. commands in the usual way to put titles etc on the display.

4.3 Loading the Routines

As explained in the introduction of this manual, the routines are supplied on cassette in a form that can be read using the *EXEC command. This is done so that the routines can be added to an existing program. There are two fundamental files of Level One routines, L1-2D and L1-3D. Normally you will require one or other of these files, but memory space permitting they can be loaded together if you wish. The other Level one routines are supplied individually on files, and so when they are wanted they should be loaded in addition to the fundamental files.

4.4 Using the Routines

The BBC microcomputer has extensive and powerful graphics facilities built into it, but they require the user to address the screen in units which will not necessarily bear any relation to the units in which the user would perhaps like to work for a particular program. For example, if a bar chart of monthly sales was wanted, it might be nice to specify the bar heights in units of £s (or £1,000!) and the position horizontally of each bar in months. Level One routines are designed to enable the user to do just that.

For normal two dimensional plotting, the routines convert x-coordinates onto horizontal displacements on the screen, and y-coordinates onto vertical screen displacements. As the user may not always want to use the whole screen for a particular graph, a means is provided of specifying which region of the screen is to be used and what range of user coordinates is to be fitted onto the screen region. But even this information is not sufficient to enable axes to be drawn: the point in user coordinate space where the axes are to cross must also be specified. Usually this will be (0,0), but not always, as for example, one might wish to plot bar charts for the years 1945 to 1982 which does not include the year 0 AD.

The actual mapping used in two-dimensional plotting is therefore of the form:

```
X screen = £XCR + £XFA*(X-£SXO)
Y screen = £YCR + £YFA*(Y-£SYO]
```

where:

(X,Y) is the point in user coordinates to be plotted

(£SXO,£SYO) is the axes crossing point in user

coordinates

(£XCR,£YCR) are the screen coordinates of the crossing point

(X screen, Y screen) are the screen coordinates of the point being plotted

£XFA, £YEA are scale factors

The screen coordinates are exactly as defined in the User Guide.

The information needed to set up this transformation is passed to the graphics routines by means of graphics parameters (see below). Similar considerations apply to 3-dimensional plotting: fuller details are given in the description of the axes procedures below.

4.5 Errors

If, during a run, the BASIC system were to report errors within the routines provided, this would usually be due to incorrect settings of the primary parameters (see next section). The user should check that PROCEINIT has been called to set these, or that the user's program has set all the parameters. Next, the actual values set by the user should be checked: note that the axis range parameters £XL, £XH, etc., must be set with £XL < £XH, and equality is not allowed (a range of zero width cannot be mapped onto a finite screen width) . Also £XI%, etc. , must be strictly positive, and zero values of XN% or £YN% could lead to division by zero.

4.6 Graphics Parameters

Graphics parameters are just specially earmarked variables which are set to give the routines the data they need to work out the transformation from user or problem units to screen units. They include such things as the range of values for each axis (1945 to 1982 for example), the colour of each axis, how many intervals there are to be along each axis, etc.

There are actually two kinds of parameters: primary and secondary . Broadly speaking , the primary parameters are targets, that is values that the user would like to use, whereas the secondary parameters are the actual parameters used internally by the routines to define the scaling and transformations.

The need for this distinction will become clearer as the structure of the routine is understood, but one example has already been mentioned, that is, the occasional need to shift the crossing point of the axes. The exact algorithm will be described later. One rule always holds: the secondary parameters are calculated from the primary parameters. However, it is not necessary to understand the secondary parameters in detail to use Level One routines. Another rule is that primary parameter values are never altered except by direct request of the user.

Each parameter has a default value, invoked by the function PROCfINIT (described below). Some parameters are not always required (e.g. all those referring to the Z-axis are not used in 2-D plotting), but all parameters are initialised by PROCfINIT, even if they are not going to be used.

It is important to note that the user is responsible for setting the primary parameters to sensible values. As most of the standard values will be satisfactory for most purposes, the most efficient way of doing this is to call PROCfINIT first, and then over-ride particular parameter values as required. But if the user over-rides any default values with nonsense values, errors will occur. One obvious error is, for example, to set fXL and fXH (low and high ends of the x-axis) to the same value which obviously prevents a scale factor being found.

4.6.1 Primary Graphics Parameters

Now follows a list of all the primary parameters and the values that they acquire as a result of calling PROCfINIT.

Parameter name	fXL	fYL	fZL	fXH	fYH
Default value	-10	-10	-10	10	10
Restrictions	High values must be strictly greater than low values.				

These parameters define the range of X, Y and Z values which must be at least spanned by the axes. The actual values used (i.e. the values of the corresponding secondary parameters] may be different but will always include the primary range . One reason for the difference might be that the interval size along an axis is not an exact fraction of the primary range (see below).

Parameter name	£XI%	£YI%	£ZI%
Default value	10	10	10
Restrictions	Values must be positive integers		

These parameters are used as a guide to the number of intervals that are to be marked off along each axis. The range is calculated from the primary parameters, and this is used to obtain a target interval size, for example:

```
target X-interval size = (£XH-£XL)/£XI%
```

The actual interval size used is the nearest of values of the form 1, 2 or 5 times a power of 10. This ensures that interval sizes are always sensible numbers. The axis range parameters (£XL, £XH etc) might not give multiples of the interval size, and thus the secondary parameters for the axis ranges may not be equal to the primary parameters.

Parameter name	£XO	£YO	£ZO
Default value	0	0	0

These parameters the point of intersection in the axes as drawn. The corresponding secondary parameters will only differ if this point does not map into the current screen sector (see below).

Parameter name	£XP%	£YP%	£ZP%
Default value	484040225	484040225	484040225

Each of these parameters contains 4 pieces of information. The first 3 digits (default 484) after having 500 subtracted (giving -16 by default) define a displacement in screen units which is used when labelling the axes to avoid the figures coming on top of the axis line. The next pair of digits (04) gives the maximum number of spaces that may be used for printing values alongside the axes. The next pair (02) is the 'pip mode'; value 2 makes the pips straddle the axis, 0 puts them on one side (above for X, to the right for Y and Z), I puts them on the other side. The final pair of digits defines the length of the pips as a percentage of interval size along one of the other axes (Y for X-axis, X for the Y and Z-axes).

These parameters allow the colours used when drawing the axes to be varied. If digit 7 (counting from the right, starting at 1) is zero, then when that axis is being drawn, having pips drawn, or labelled, the routines will not change the current graphics colour. But if digit 7 is 1, then each of the other pairs of

digits defines the logical colour to be used for axes, pips, and labelling respectively. With default values, logical colour I will be used for axes, 2 for pips, and 3 for labelling. When using modes with less than 4 colours, £XC% is set to zero so that the graphics colours are never changed.

The operations of drawing axes, pips and labelling are done in the following order: X, Y, then Z--axes, X, Y, then Z pips, X, Y then Z labelling. Thus the default values give each axis the same colour scheme, because digit 7 is zero for the Y & Z axes.

Parameter name	£XN%	£YN%	£SN%
Default value	1	1	1
Restrictions	Values must be positive integers		

These parameters enable the screen to be divided into an arbitrary number of rectangular sectors and a particular graph drawn in just one. The diagram shows the effect of taking £XN% = 3, £YN% = 2, giving 3 intervals horizontally and 2 intervals vertically. Each sector is given a unique number (£SN%), numbered consecutively from left to right starting at £SN% = 1 for the lower left sector.

4 5 6

1 2 3

The screen area to be sectored is defined by further parameters - see below.

Parameter name	£TH	£PH
Default value	1.3	-1.3

These parameters are only relevant for 3-D perspective plotting. They are the usual spherical polar angles in radians, and give the direction from which the graph is to be viewed. £TH is the angle between the line of sight and the Z-axis, and £PH is the angle between the x-axis and the plane through the line of sight and the Z-axis.

Parameter name	£HM%	£VM%	£CH%
Default value	64064	64064	32032

The default values actually depend on the mode in use. £CH% defines the width and height of characters in the mode selected, combined in the form 1000*width + height in screen units. £HM% and £VM% define how much of a margin is to be allowed inside the current screen sector when calculating scale factors, etc. £HM% is in the form 1000*(left margin) + (right margin), and £VM% similarly gives the top and bottom margin.

Parameter name	£XB%	£YB%	£XS%	£YS%
Default value	96	0	1184	1023

These parameters define the window within which plotting will be done. The window is a rectangle whose lower left corner is at (£XB%, £YB%] in screen coordinates and whose sides have length £XS%, £YS%, £XN%, £YN%, and £SN% determine how this window is to

Parameter name	£XM%
Default value	none

This parameter is set to the value of the argument M used in the call to PROC£INIT(M). It therefore records the mode that the user indicated for graphics. The routines never alter the graphics mode so this is not necessarily the actual mode at the time.

4.6.2 The Axes-setting Procedures

PROC£AXES(K) for 2-D plotting

PROC£AX3(K) for 3-D plotting

These procedures perform three separate actions, selected by K.

Action	
0	combines all the actions
1	sets the secondary parameters (sets up a transformation)
1	draws axes and mark them off with pips
3	labels the axes

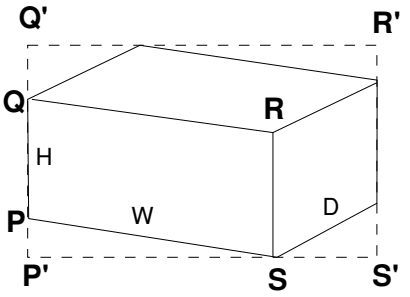
Although both routines conform to the above brief description, their actual effects are slightly different.

PROCfAXES only sets secondary parameters concerned with 2-D plotting. It first chooses a suitable interval size for marking off axes, in the manner described above (Section 4.5, see £XI%, etc.). If necessary, it then chooses a crossing point for the axes. This will differ from the primary parameter value only if that value lies outside the range to be spanned, and in that case a crossing point near the low end of the range is chosen. The actual end points of each axis are now chosen so that end points and axis crossing points all occur at integer multiples of the interval sizes (in user units). Next, the coordinates of the lower left hand corner of the chosen screen sector are calculated, together with the width and height of the sector. Scale factors are now calculated, and the horizontal and vertical factors are set to be equal if the initial calculation shows them to be within 30% of each other. The position of the axis crossing is now calculated in screen coordinates.

The axes are drawn and marked off. Finally, the axes are labelled. An attempt is made to label every axis marker, but if this would crowd the screen, only the end points are labelled. The labelling uses a printing format which may be scaled by powers of 10 if necessary. In that case the power of 10 used is displayed at the high end of the axis, using an E followed by the power of 10.

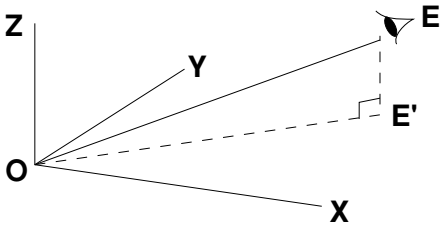
AX3 follows the same course for the X,Y,Z, axes up to the calculation of screen sector parameters. At this point axes ranges, interval size, and the screen sector have been chosen. To calculate the scaling factors a notional 3-1) box-shaped region is set up, having height equal to the sector height, width equal to the sector width, and depth equal to width.

Temporary scale factors are found which scale the user's x-range to width, y-range to depth, and z-range to height. The x and y factors are now equalised to the smaller of these two if the original values were within a ratio of 7/3 of each other. The x and z factor are next equalised if within 6/4 of each other.



The temporary factors must be further modified because under perspective projection, values of (X,Y,Z) lying within the notional box will not map into region PQRS, but into P'Q'R'S'. First the projection matrix elements are computed (formula below), then the extent of P'Q'R'S', using the temporary scale factors just found. A 'shrink factor' can now be found which reduces P'Q'R'S' to lie within the chosen screen sector. All the scale factors are then modified by this factor.

The projection is that obtained by projecting onto a plane orthogonal to the line of sight OE when the picture is viewed from infinity. THETA is the angle ZOE, and PHI the angle XOE' where OE' is the line of intersection of the plane through ZOE with the OXY plane.



If XF, YE, ZE are the scale factors, XC, YC the cm displacements from the axis crossing point on screen, and X,Y,Z the displacements from the user axis crossing:

$$\begin{matrix} XC \\ \\ YC \end{matrix} = \begin{matrix} -\sin(\phi) & \cos(\phi) & 0 \\ -\cos(\theta)\cos(\phi) & -\cos(\theta)\sin(\phi) & \sin(\phi) \end{matrix} \begin{matrix} XF*X \\ \\ YF*Y \\ ZF*Z \end{matrix}$$

The axis crossing point must now be positioned, which is done by finding the projection and user ranges the area of screen needed. The axis crossing point is then chosen to map this area centrally within the sector.

This algorithm is necessarily complicated and may not always achieve the effect desired by the user. The projection is defined entirely by secondary parameters which can of course be over-ridden by the user if desired (see below for list of these).

4.6.3 The Secondary Graphics Parameters

In order to make full use of the flexibility offered by Level One routines, the user may wish to over-ride some of the secondary parameters. For instance, the user may wish to be able to set the interval size between axis marks, or to set the scale factors explicitly.

The sequence of operations needed to achieve this will already be apparent from earlier sections. For instance, when using the 2-D routines, the user could break up the normal sequence of operations during a call to AXES as follows:

```
PROCfAXES(1) Set secondary parameters as usual
User code to over-ride any secondary parameters
PROCFfAXES(2) Draw axes
etc.
```

List of secondary parameters:

```
fSXL  fSYL  fSZL  fSXH  fSYH  fSZH

define actual span of axes

fSKI  fSYI  fSZI

interval between pips on axes (user units)


     fSXO  fSYO  fSZO        fXCR%  fYCR%

axes crossing point    axes crossing point
    (user units)          (screen units)


Scale factors
```

```
f$XB% f$YB% f$XS% f$YS%

screen coords of base of current sector,
and length of its sides

f$11  f$12  f$21  f$22  f$23
elements of transformation matrix used for
3-D plotting
```

Some of these parameters are used in the transformation algorithms which map user coordinates into screen coordinates.

For 2-D plotting, the point (X,Y) is transformed as follows:

```
X screen = f%XCR% + f$XFA * (X - f$SXO]
Y screen = f%YCR% + f$YEA * (Y - f$SYO)
```

For 3-D plotting, the point (X,Y,Z) in user units is transformed in two stages.

```
X′ = f$XFA * (X-f$SXO)
Y′ = f$YEA * (Y-f$SYO)
Z′ = f$ZFA * (Z-f$SZO)

X screen = EXCR% + f$11*X′ + f$12*Y′
Y screen = f%YCR% + f$21 *X ′ + f$22*Y′ + f$23 *Z′
```

4.7 List of Level One Procedures

L1-2D Fundamental 2-dimensional plotting procedures

These procedures allow plotting to be done in the user's own coordinates.

PROCEINIT (M) PROC£AXES(K)
PROC£MOVE (X,Y) PROC£DRAW(X,Y) PROCEPLOT (K,X,Y)
FN£POINT(X,Y) FNmax(A,B) FNmin(A,B)

This set of routines is fundamental to all other 2-dimensional plotting routines. They allow the user to work in units which are appropriate to application rather than screen units as used by MOVE, DRAW, and PLOT.

Before any other procedure can be used, PROC£INIT(M) must be called with M set to the graphics mode that will be used when plotting. This initialises the primary graphics parameters (Section 4.6.1]. If any non-default values are required, these must be set at this point, after initialisation.

The next step is to call PROCF-AXES (K) usually with K=0. This will 'set up a transformation', or in other words, will make it possible for the other routines to translate problem units into screen units. The call PROC£AXES(O) does this and also draws and labels the coordinate axes. It is possible to unbundle all these actions, indeed this is the purpose of the argument K.

K=1 sets up the transformation,
K=2 draws the axes, and
K=3 labels the axes.

Thus

```
10 PROC£AXES(0)
```

is equivalent to

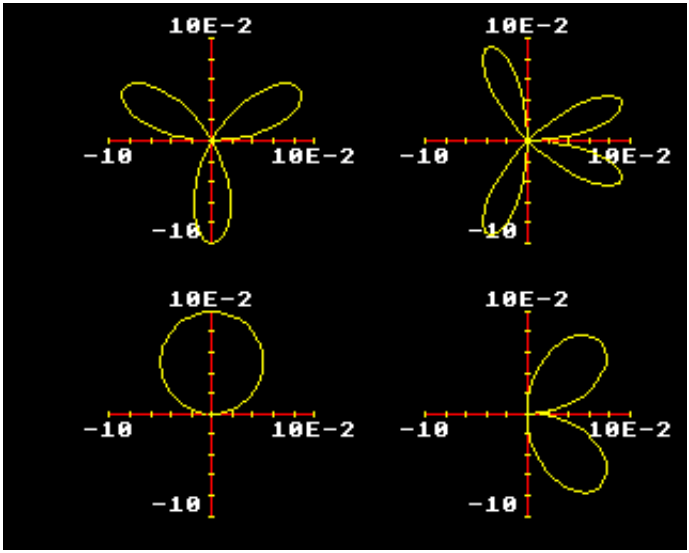
```
20 PROC£AXES(1):PROC£AXES(2):PROC£AXES(3)
```

Once a transformation has been set, the procedures PROC£MOVE, PROC£DRAW, and PROC£PLOT may be used. These behave exactly as MOVE, DRAW and PLOT (as described in the BBC Microcomputer User Guide) except that X and Y are given in the user's own problem units.

Besides the routines named, this suite of routines also includes other routines used internally by the package. Listings are given in an appendix at the end of this manual. All internal procedure names start with E m accordance with the naming convention, with the exception of FNmin and FNmax. These functions

return the minimum and maximum value respectively of the two arguments.

Example Program




```
  5 REM L1-2DX1
 10 MODE1:PROCfINIT(1)
 15 fXL=-.1:fXH=.1:fYL=-.1:fYH=.1
 20 fXN%=2:fYN%=2
 25 FOR S%= 1 TO fXN%*fYN%
 30 fSN%=S%: PROCfAXES(0)
 35 PROCfMOVE(0,0):GCOL 0,2
 40 FOR th= 0 TO PI STEP 2*PI/100
 50 r = .1*SIN(S%*th)
 60 x = r *COS(th)
 70 y = r *SIN(th)
 80 PROCfDRAW(x,y)
 90 NEXT:NEXT
100 END
200 REM -------------------------
```

Ll-3D Fundamental 3-dimensional plotting procedures

These procedures allow plotting to be done in the user's own X,Y,Z coordinates, and produce a perspective view.

PROCfINIT(M) PROCfAX3(K)
PROCfMV3(X,Y,Z) PROCfDR3(X,Y,Z) PROCfPL3 (K,X,Y,Z)
FNfPT3 (X,Y,Z)

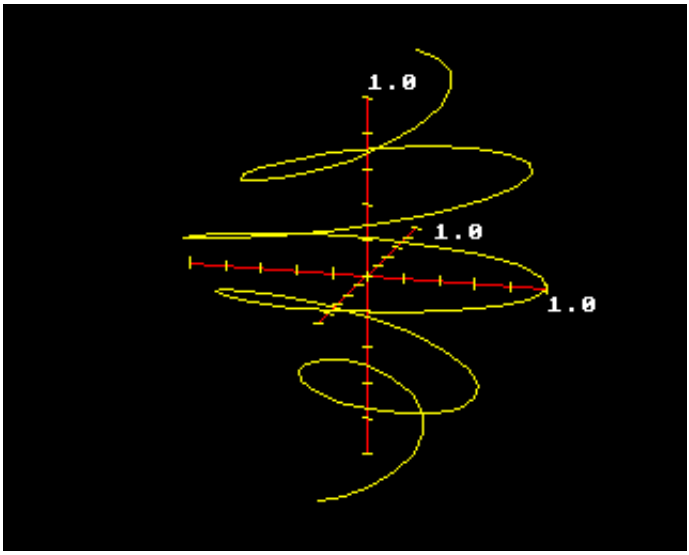
This is of routines are fundamental to all other is-dimensional plotting routines. They behave exactly like the 2-dimensional routines PROCfMOVE etc (see Ll-2D) except that the user must give three coordinates X,Y and Z.

Before any other procedure can be used, PROCfINIT(M) must be called with M set to the graphics mode that will be used when plotting. This initialises the primary graphics parameters (Section 4.6.l). If any non-default values are required, these must be set at this point, after initialisation. Note that some parameters which have no effect on 'f-dimensional plotting now become useful, for example fTH and fPH which control the viewing angle used when drawing in perspective.

The next step is to call PROCfAX3(K), usually with K= 0. PR0CfAX3(K) is the 3-dimensional equivalent of PROCfAXES; it sets up the perspective transformation. The argument K behaves just as in the 2-dimensional case.

Besides the routines named, this suite of routines also includes the same set of internal routines as for Ll-2D.

Example Program




```
100 REM L1-3DX1
110 MODE1:PROCfINIT(1)
115 fXL=-1:fYL=-1:fZL=-1
120 fXH=1:fYH=1:fZH=1
130 PROCfAX3(0)
140 GCOL0,2:FOR I=0 TO 80: Z=2*I/80-1
150 x=COS(4*PI*Z): y=.3*SIN(4*PI*Z)
160 c=COS(Z*PI/2): s=SIN(Z*PI/2)
170 X=x*c-y*s: Y=x*s+y*c
180 IF I=0 THEN PROCfMV3(X,Y,Z) ELSE PROCfDR3(X,Y,Z)
190 NEXT:END
200 REM
```

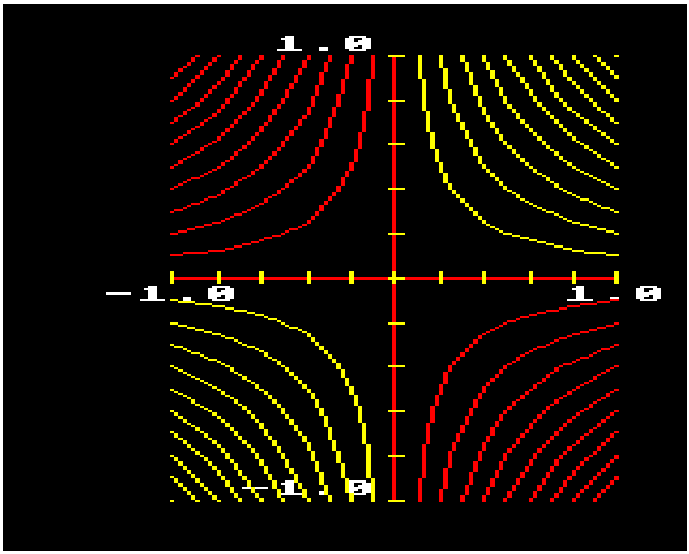
L1-CNTR Locates contours in mesh coordinates

PROCfCNTR(Z,IM%,JM%) - calls user-defined procedure PROCfPLOTCON(K,I,J)

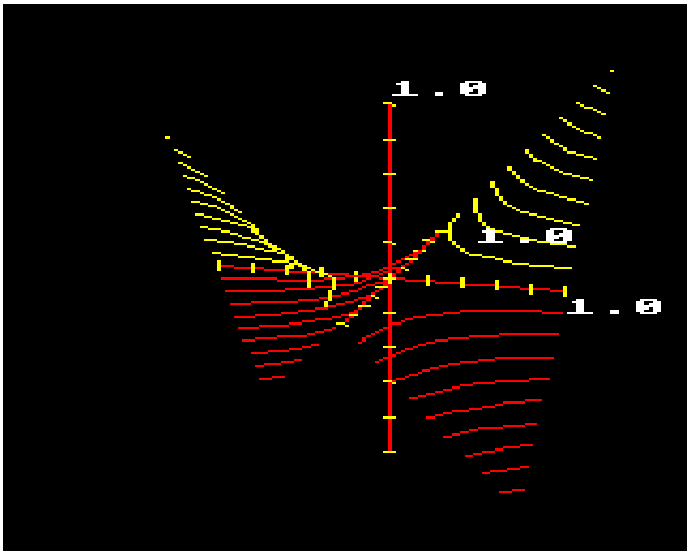
This procedure assumes that the user has previously set up values in the array fW(I,J), which of course must also have been dimensioned. The routine will expect I to run between 0 and IM%, J between 0 and JM%. The set of points (I,J) with values in this range is called a 'mesh'. Usually, fW(I,J) is thought of as representing the values of points on a hill at distances X east, Y north from some fixed point of

reference. The collection of points where the heights are known are called 'mesh points'. To draw a contour (in our example the curve along which all points of the hill are at constant height), the routine has to find all the places where the heights are the wanted height, which will have been specified in the first argument Z of the call to PROCfCNTR. These places will not all lie exactly at the mesh points; if you imagine the mesh points all joined up into a set of squares, then PR0CfCNTR actually finds the places where the contour crosses any side.

PROCfCNTR does not actually join all these points up itself. Instead, it calls another routine which the user must have defined in his own part of the program. This must be called PROCF-PLOTCON, and must expect arguments K,I,J, PROCfCNTR will call PROCF-PLOTCON every time it finds a place where a contour crosses the side of a mesh square, and it will set K to 4 if a MOVE is required, and to 5 if a DRAW is required. I and J will be set to the crossing place using mesh coordinates; it is up to the user to decide whether to convert these to his own system before calling a plotting routine. Note that I and J as set on this call need not be whole numbers, and that K follows the control variable convention as used in PLOT (in the BBC Microcomputer User Guide]. It is also important to note that care must be taken wit thin PROCF-PLOTCON if global variables are to be used; these must not have the same names as any of PROCfCNTR's local variables, because the interpreter will not restore the global values until it finds an ENDPROC statement to take it out of PROCECNTR. Local variables used by PROCE'CNTR are: D, F%, I%, IM%, J%, JM%, K%, X, Y, Z, ZO, ZI. It is of course perfectly alright to use local variables with these names. It may also be useful to note that Z, IM% and JM% take the values of the corresponding arguments throughout the routine.




```
 10 REM L1-CNX1
 30 MODE5:PROCfINIT(5)
 40 fXL=-1:fXH=1:fYL=-1:fYH=1
 50 PROCfAXES(0)
 60 N=10:NH=5:N2=NH^2:DIM fW(10,10)
 70 FOR I=0 TO N: FOR J=0 TO N
 80 fW(I,J)=(I-NH)*(J-NH)/N2:NEXT:NEXT
 90 FOR Z=-1 TO 1 STEP .09999
100 IF Z<0 THEN GCOL 0,1 ELSE GCOL 0,2
110 IF ABS(Z)>.01 THEN PROCfCNTR(Z,N,N)
120 NEXT
130 END
140 REM ------ Plotting procedure ---
150 DEF PROCfPLOTCON(K%,I,J): LOCAL X,Y
160 X=(I-NH)/NH:Y=(J-NH)/NH
170 PROCfPLOT(K%,X,Y)
180 ENDPROC
199 REM-------------------------------
```




```
 10 REM L1-CNX2
 30 MODE5:PROC£INIT(5)
 40 £XL=-1:£XH=1:£YL=-1:£YH=1
 50 £ZL=-1:£ZH=1:PROC£AX3(0)
 60 N=10:NH=5:N2=NH^2:DIM £W(10,10)
 70 FOR I=0 TO N:FOR J=0 TO N
 80 £W(I,J)=(I-NH)*(J-NH)/N2:NEXT:NEXT
 90 FOR Z=-1 TO 1 STEP .09999
100 IF Z<0 THEN GCOL 0,1 ELSE GCOL 0,2
110 IF ABS(Z)>.01 THEN PROC£CNTR(Z,N,N)
120 NEXT
130 END
140 REM ------ Plotting procedure ---
150 DEF PROC£PLOTCON(K%,I,J): LOCAL X,Y
160 X=(I-NH)/NH:Y=(J-NH)/NH
170 PROC£PL3(K%,X,Y,Z)
180 ENDPROC
199 REM ------------------------------
```

L1-BOX Draws a box around the current sector

This procedure assumes that the user has previously called PROCEINIT or otherwise set £SXB, £SYB, £SYS, and draws a box around the region so defined in the current graphics colour.

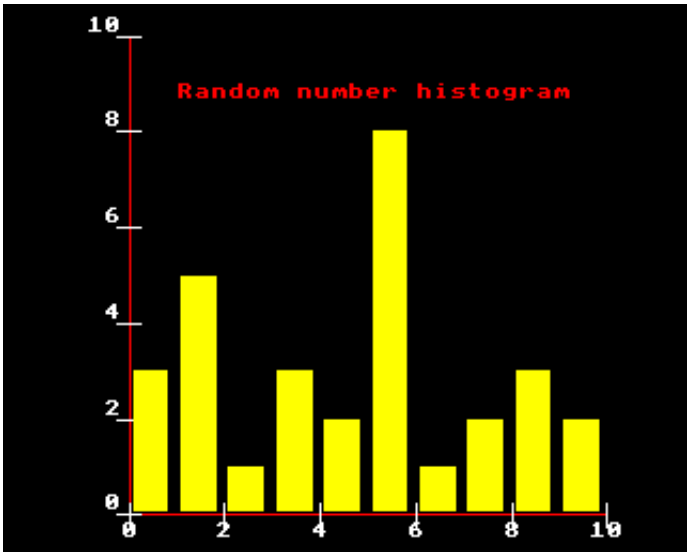
L1-BAR Draws an individual histogram bar

This procedure assumes that there has previously been a call to PROCEINIT and PROCE-AXES so that a 'z-dimensional transformation has been set up. Using the current graphics colour , it draws a bar, positioned according to K and X, base of bar at height YO, top of bar at Y, and of width W.

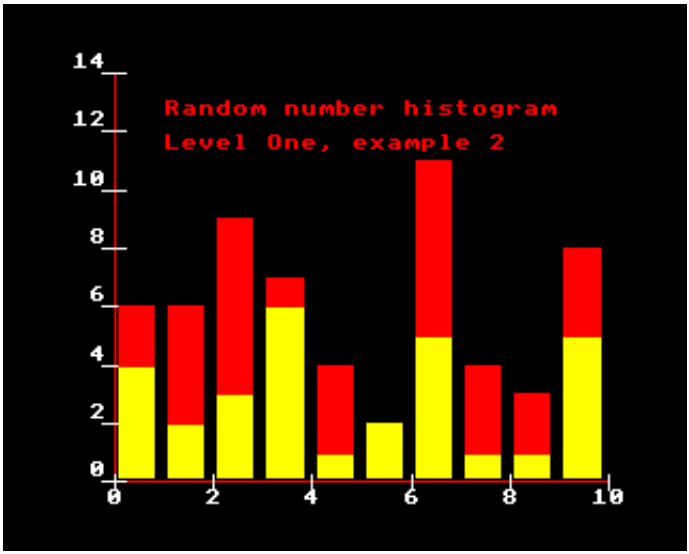
If K=0, then the left side of the bar is at X, and if K=l, then the bar is centred at X.

It is allowed for Y to be less than YO, so that 'downward' bars may be drawn.

Example Programs




```
210 REM L1-BAX1
220 MODE1:PROCfINIT(1)
230 fXL=0:fYL=0:fXH=10:fYH=10
240 fXC%=10103O3
250 PROCfAXES(0)
260 REM  generates 30 random no.s
270 REM  and count frequency.
280 DIM Y(10)
290 FOR I%=1 TO 30
300 J%=10*RND(1): Y(J%)=Y(J%)+1: NEXT
310 GCOL 0,2
320 FOR I%=0 TO 9:PROCfBAR(0,I%+.1,0,Y(I%),.7):NEXT
325 PROCfMOVE(1,9):VDU5:GCOL 0,1
328 PRINT "Random number histogram"
330 VDU4:END
```




```
  210 REM L1-BAX2
  220 MODE1:PROCfINIT(1)
  230 fXL=0:fYL=0:fXH=10:fYH=14
  240 fVM%=128128:fHM%=128128:fXC%=1010303
  250 PROCfAXES(0)
  260 REM  twice generate 30 random no.s
  270 REM  and count frequency.
  280 DIM Y(10),Z(10)
  290 FOR I%=1 TO 30
  295 J%=10*RND(1): Z(J%)=Z(J%)+1
  300 J%=10*RND(1): Y(J%)=Y(J%)+1: NEXT
  320 FOR I%=0 TO 9
  322 GCOL 0,2:PROCfBAR(0,I%+.1,0,Y(I%),.7)
  324 GCOL
0,1:PROCfBAR(0,I%+.1,Y(I%),Z(I%)+Y(I%),.7):NEXT
  325 PROCfMOVE(1,fSYH-1):VDU5:GCOL 0,1
  328 PRINT "Random number histogram"
  330 PROCfMOVE(1,fSYH-1):VDU10:VDU10
  332 PRINT"Level One, example 2"
  340 VDU4:VDU30:END
```

L1-SEC Draws an individual pie chart sector

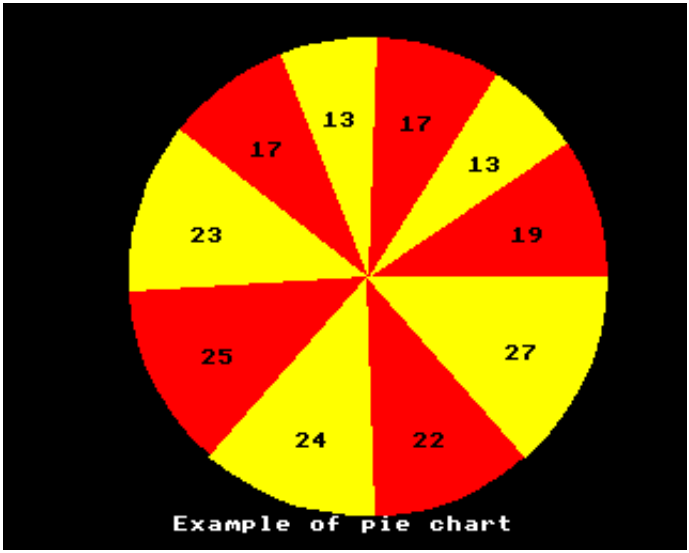
PROCfSEC(M,X,Y,R,A,B)

This procedure draws and fills in a sector of a circle, whose centre is at X,Y, and whose radius is R, and which starts at angle A and covers an angle B. All angles are measured in revolutions (so I is a full circle, .5 is half a circle, etc.), and are measured anti clockwise from the right hand horizontal direction.

The units expected for X, Y and R depend on the control variable M. This should be thought of as having two digits ab. If a=0, then X and Y are expected in screen coordinates, and if a=1, then they must be in user coordinates and a transformation must have been set up (by previous use of PROCfAXES for example). Similarly, b=0 means that R is in screen units, and b=1 means user units.

On leaving the procedure, the graphics cursor is left at a point inside the sector ready for text or labelling.

Example Program




```
400 REM L1-SEX1
420 MODE1:PROCfINIT(1):a%=@%:@%=&1000202
```

```
430 PROCfAXES(1)
440 REM  generate 200 random no.s
450 REM  and count frequency.
460 DIM Y(10)
470 FOR I%=1 TO 200
480 J%=10*RND(1):Y(J%)=Y(J%)+1:NEXT
490 A=0:FOR I%=0 TO 9:B=A+Y(I%)/200
500 IF I%MOD2=0 THEN GCOL 0,1 ELSE GCOL 0,2
505 REM ---- draw the pie sector ---
510 PROCfSEC(11,0,0,10,A,B)
512 REM
515 VDU5:VDU8:GCOL 0,0:PRINT Y(I%):VDU4
520 A=B:NEXT:GCOL 0,3
525 PRINT TAB(10,30);"Example of pie chart"
530 VDU30:@%=a%:END
```

5 Complete Listings

5.1 Level Three Programs

```
    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 : REM L3-BAR
   20 f%=5 : W=.7:off=.1
   40 ON ERROR GOTO 700
   90 MODE4:VDU19 128,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 f%=FNp(K%,4,"Mode for chart",f%)
  120 PROCfINIT(f%):fXL=0:fYL=0
  130 PROCc(6)
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
  170 PRINT TAB(2,4) "Horizontal axis:"
  180 fXL=FNp(K%,5,"Left end",fXL)
  190 fXH=FNp(K%,6,"Right end",fXH)
  200 PRINT TAB(2,8) "Vertical axis:"
  210 fYL=FNp(K%,9,"Low end",fYL)
  220 fYH=FNp(K%,10,"High end",fYH)
  230 off=FNp(K%,12,"Offset",off)
  240 W=FNp(K%,13,"Bar width",W)
  250 K%=1:UNTIL FNe
  260 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO DRAW CHART"
  320 PRINT TAB(2,6)"After axes drawn, use keys:"
  330 PRINT TAB(2,8)"SPACE: next X,Y"
  340 PRINT"  C    : colour "
  342 PRINT"  L    : change logical colours"
  350 PRINT"  B    : bar base"
  360 PRINT"  O    : offset"
  370 PRINT"  W    : width"
  375 PRINT"  T    : title"
  378 PRINT"  ?    : list of prompt keys"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=fCH%MOD1E3:fYB%=3*X:fYS%=1023-4*X
  510 REM
  520 MODE f%
  530 PROCfAXES(0):PROCfco(fXC%,2)
  540 YB=fSYO
  550 PRINTTAB(0,29)SPC(1280DIV(fCH%MOD1E3))TAB(0,29);
  560 A$=GET$
  570 IF A$=" "THENINPUT"X,Y " ,X,Y : PROCfBAR(0,X+off,YB,Y,W)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical , actual col " ,A%,B%:
VDU19 A%,B%,0,0,0
  590 IF A$="O"THENINPUT"Offset " ,off
  600 IF A$="W"THENINPUT"Bar width ",W
  610 IF A$="B"THENINPUT"Bar base ",YE
  620 IF A$="T"THENINPUT"Title,X,Y",A$,X,Y:PROCfMOVE
```

```
(X,Y):VDU5:PRINTA$:VDU4
  630 IF A$="?"THENPRINT"SPACE,C,L,O,W,B,T":A$=INKEY$(500)
  640 GOTO 550
  690 REM error branch
  700  MODE7:IF ERR=17 THEN900ELSESTOP
  900 PRINTTAB(0,10)"Stop or repeat? (S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R" THEN 90
  940 GOTO910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%) V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$) ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"BAR CHART UTILITY";SPC(5); "PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =FALSE
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCc(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FNfpr(fXC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FNfpr(fXC%,4):B%=FNfpr(fXC%,2)
 1320 C%=FNfpr(fXC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
 1350 C%=FNp(K%,L%+4,"labels",C%)
 1360 fXC%=((100+A%)*100+B%)*100+C%
 1370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 1380 ENDPROC
10040 REM
10050 REM Initialisation
10060 DEF PROCfINIT(M%): fM%=M%
10070 fXL=-10:fXH=10:fYL=-10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1: fYN%=1: fSN%=1
10100 IF M%=2 OR M%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=10102O3 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FNfI(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROCfer("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
```

```
10240 REM normalise interval size ^10
10250 A=10^INT(LOG(C)):B=C/A
10260 REM select nearest round interval
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4 THENB=2ELSEB=1
10280 =A*B
10290 REM
10300 REM calc secondary origin, lo, hi
10310 DEF PROCfax(O,LO,HI,I)
10320 REM if axes wld cross out range,
10330 REM set sec pars so cross at lo,
10340 REM and ensure orig at integer*intvl
10350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.1)*I:LO=O
10360 REM anyway make ends at intg*intvl
10370 LO=O+I*INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10390 REM return value in f0,f1,f2
10400 f0=O:f1=LO:f2=HI:ENDPROC
10410 REM
10420 REM set up sector
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%: fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIVfXN%*fSYS%:ENDPROC
10480 REM
10490 DEF FNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)
-(M%DIV1000))/f0
10500 REM
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2) MOD 3) GOTO 10540,10550,10560
10540 =0.
10550 =-f0
10560 =-f0/2
10570 REM
10580 REM Extract colour
10590 DEF PROCfco(C%,N%):IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
10600 ENDPROC
10610 REM
10620 REM scale or unscale labels
10630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10650 REM digits needed c% before, d% after
10660 IF j%>0 THEN c%=j%+1:ELSE c%=1
10670 IF NOT i%<0 THEN d%=0:p%=0: ELSE d%=-i%:p%=1
10680 REM is sign needed
10690 IF lo<0 OR hi<0 THEN s%=1 ELSEs%=0
10700 REM decide scaled/unscaled
10710 IF NOT(f%<p%+s%+c%+d%) THEN f0=0 ELSE d%=0:
p%=0:f0=FNmin(-i%,f%-j%-s%-1)
10720 REM format; geN if integer el fix
10730 IF d%=0 THEN i%=&10000+f% ELSE i%=&10200+d%
10740 =f%+&100*i%
10750 REM
10760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEF FNfpr(n%,d%)=(n%DIV(10^d%))MOD100
10790 DEF PROCfer(a$):PRINT "Graphics package error"
10800 PRINT a$:STOP:ENDPROC
10810 REM
```

```
12000 REM ----- TWO DIM ROUTINES -----
12010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
12020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X) ,FNfy(Y):ENDPROC
12030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
12040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y:fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
12050 PLOT K%,x,y:ENDPROC
12060 DEF FNfPOINT(X,Y)=POINT FNfx(X),FNfy(Y)
12070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
12080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
12090 REM two dim scales & axes
12100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
12110 IF M%=1 THEN GOSUB 12140 ELSE IF M%=2 THEN GOSUB 12300 ELSE IF
M%=3 THEN GOSUB 12400 ELSE GOSUB 12140:GOSUB 12300:GOSUB 12400
12120 ENDPROC
12130 REM set intervals & ends
12140 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH)
12150 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
12160  PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
12170 REM set sector
12180 PROCfse
12190 REM scale factors, allow margins
12200 fXFA=FNffa(fSXS%,fHM%,fSXH,fSXL):A=f0*fXFA
12210 fYFA=FNffa(fSYS%,fVM%,fSYH,fSYL):B=f0*fYFA
12220 REM equalise if within 30%
12230 f0=ABS((fXFA-fYFA)/(fXFA+fYFA))
12240 IF ABS((fXFA-fYFA)/(fXFA+fYFA))<.15 THEN fXFA=
FNmin(fXFA,fYFA):fYFA=fXFA
12250 REM position the axes cross
12260 fXCR%=fSXB%+(fHM% DIV 1000)+fXFA*(fSXO-fSXL)+(A-fXFA*(fSXH-
fSXL))/2
12270 fYCR%=fSYB%+(fVM%DIV1000)+fYFA*(fSYO-fSYL)+(B-fYFA*(fSYH-
fSYL))/2
12280 RETURN
12290 REM draw & mark off axes
12300 PROCfco(fXC%,4):PROCfMOVE(fSXL, fSYO):PROCfDRAW(fSXH,fSYO)
12310 PROCfco(fYC%,4):PROCfMOVE(fSXO,fSYL):PROCfDRAW(fSXO,fSYH)
12320 B=FNfpp(fSYI,fXP%):PROCfco(fXC%,2)
12330 FOR A=fSXL TO fSXH+fSXI/2 STEP fSXI
12340 PROCfMOVE(A,fSYO+B):PROCfDRAW(A, fSYO+B+f0):NEXT
12350 A=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
12360 FOR B=fSYL TO fSYH+fSYI/2 STEP fSYI
12370 PROCfMOVE(fSXO+A,B):PROCfPLOT(1,f0,0):NEXT
12380 RETURN
12390 REM label axes
12400 f%=FNfpr(fXP%,4):w%=fCH%DIVlE3
12410 REM decide scaled or unscaled
12420 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,f%)
12430 REM decide if OK to label all pts
12440 IFfXFA*fSXI>(1+f%)*w%ANDNOT(fM%=2ORfM%=5)THEN l%=TRUE ELSEl%=
FALSE
12450 REM now label
12460 VDU 5:PROCfco(fXC%,0):A=fSXH
12470 IFf0<>0THENL$="E"+STR$(-f0)ELSEL$=""
12480 d%=fSXB%+fSXS%:y%=FNfy(fSYO)+(fXP%DIVlE6)-500
12490 REPEATIFABS(A)<fSXI/2THENA=0
12500 L$=STR$(A*10^f0)+L$:B=LEN(L$)*w%
12510 d%=FNmin(FNfx(A)-B/2,d%-B-w%DIV2)
12520 PLOT4,d%,y%:PRINTL$:L$=""
12530 IFl%THENA=A-fSXI ELSEA=fSXL:l%=TRUE
12540 UNTILA<fSXL-fSXI/2
```

```
l2550 REM now label Y-axis
l2560 f%=FNfpr(fYP%,4):y%=fCH%MODlE3:d%=y%+fYP%DIVlE6-500
l2570 @%=FNfsu(fSYL,fSYH,fSYI,f%)
l2580 IFfYFA*fSYI>2*y%ANDNOT(fM%=2ORfM%=5)THENl%=lELSEl%=0
l2590 PROCfco(fYC%,0):B=fSYL
l2600 REPEAT PROCfMOVE(fSXO,B)
l2610 IF ABS(B)<fSYI/2 THEN B=0
l2620 REMIFB>fSYH-fSYI/2THENl%=2:d%=y%+( fYP%DIVlE6)-500
l2630 PLOT0,-w%DIV2,d%:FORA=lTOf%:VDU8:NEXT:PRINTB*l0^f0;
l2640 IFl%=lTHENB=B+fSYI ELSE B=fSYH:l%=l
l2650 UNTILB>fSYH +fSYI/2
l2660 IF f0<>0 THEN PRINT"E";-f0
l2670 VDU 4:@%=a%:RETURN
l5050 REM ----- Histogram bar ----
l5060 DEF PROCfBAR(K%,X,Y0,Y ,W):LOCAL x,y0
l5070 REM K%=0 left of bar at X
l5080 REM K%=l bar centered on X
l5090 REM YO is base, Y is top, W width
l5100 REM all in user units.
l5110 IF ABS(Y0-Y)*fYFA<5 THEN ENDPROC
l5120 IF Y0=fSYO THEN y0=Y0+SGN(Y-Y0)*l0/fYFA ELSE y0=Y0
l5130 IF K%=0 THEN x=X ELSE x=X-W/2
l5140 PROCfPLOT(84,x,Y):PROCfPLOT(84,x,y0)
l5150 PROCfPLOT(85,x+W,Y):PROCfPLOT( 85,x+W,y0)
l5160 ENDPROC
l5170 REM
```

```
  0 REM Graphs and Charts Pack
  l REM Copyright (C) Acornsoft l982
 l0 :REM L3-GRA
 20 f%=5:F$="0"
 40 ON ERROR GOTO700
 90 MODE4:VDUl9 l28,4,0,0,0
l00 K%=0:REPEAT PROCh(l)
l10 f%=FNp(K%,4,"Mode for graph",f%)
l20 PROCfINIT(f%)
l30 PROCC(6)
l40 K%=l:UNTIL FNe
l50 REM
l60 CLS:K%=0:REPEAT PROCh(2)
l70 PRINT TAB(2,4)"Horizontal axis:"
l80 fXL=FNp(K%,5,"Left end ",fXL)
l90 fXH=FNp(K%,6,"Right end",fXH)
200 PRINT TAB(2,8)"Vertical axis:"
210 fYL=FNp(K%,9,"Low end",fYL)
220 fYH=FNp(K%,l0,"High end",fYH)
250 K%=l:UNTIL FNe
260 REM
300 PROCh(3)
310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
320 PRINT TAB(2,6)"After axes drawn,use keys:"
330 PRINT TAB(2,8)"M: move to X,Y"
332 PRINT"  D: draw to X,Y"
340 PRINT"  C: colour"
342 PRINT"  L: change logical colours"
```

```
  350 PRINT"  F: input function(X)"
  360 PRINT"  P: range to plot,no. of points"
  370 PRINT"  U: display function at X,Y"
  375 PRINT"  T: title"
  378 PRINT"  ?: list of prompt keys"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=£CH%MOD1E3:£YB%=3 *X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AXES(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MOD1E3))TAB(0,29);
  560 A$=GET$
  570 IF A$="M"THENINPUT"Move to X,Y ",X,Y:PROC£MOVE(X,Y)
  572 IF A$="D"THENINPUT"Draw to X,Y ",X,Y:PROC£DRAW(X,Y)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IF A$="F"THENINPUT "Function",F$
  600 IF A$="P"THENINPUT"X1,X2,N",X1,X2,N%:PROCplot
  610 IF A$="U"THENINPUT"function at
X,Y",X,Y:PROC£MOVE(X,Y):VDU5:PRINTF$:VDU4
  620 IF A$="T"THENINPUT"Ti
tle,X,Y",A$,X,Y:PROC£MOVE(X,Y):VDU5:PRINTA$:VDU4
  630 IF A$="?"THENPRINT"M,D,C,L,T,F,P,U":X=INKEY(500)
  640 GOTO 550
  690 REM error branch
  700  MODE7:IF ERR=17 THEN 900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat?(S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R " THEN 90
  940 GOTO 910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"GRAPH PLOTTING UTILITY";SPC(5);"PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =FALSE
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCC(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FN£pr(£XC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FN£pr(£XC%,4):B%=FN£pr(£XC%,2)
 1320 C%=FN£pr(£XC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
```

```
 l350 C%=FNp(K%,L%+4,"labels",C%)
 l360 £XC%=((l00+A%)*l00+B%)*l00+C%
 l370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 l380 ENDPROC
 2000 DEF PROCplot:LOCAL D
 2005 D=(X2-Xl)/N%:X=Xl
 2010 PROC£MOVE(Xl,EVAL(F$))
 2020 FOR X=Xl TO X2+D/2 STEP D
 2030 PROC£DRAW(X,EVAL(F$)):NEXT
 2040 ENDPROC
l0040 REM
l0050 REM Initialisation
l0060 DEF PROC£INIT(M%):£M%=M%
l0070 £XL=-l0:£XH=l0:£YL=-l0:£YH=l0:£ZL=-l0:£ZH=l0
l0080 £XO=0:£YO=0:£ZO=0:£XI%=7:£YI%=7:£ZI%=7
l0090 £XN%=l:£YN%=l:£SN%=l
l0100 IF M%=2 OR M%=5 THEN £CH%=64 ELSE £CH%=32
l0110 £YB%=0:£YS%=l023:£XB%=3*£CH%:£XS%=l279-£XB%
l0120 £VM%=64064:£HM%=2*(l000*£CH%+£CH%):£CH%=1000*£CH%+32
l0130 £XP%=484040225:£YP%=504040225:£ZP%=540040225
l0140 IF M%=l OR M%=2 OR M%=5 THEN £XC%=1010203 ELSE £XC%=0
l0150 £YC%=0:£ZC%=0
l0160 £TH=l.3:£PH=-l.3
l0170 ENDPROC
l0180 REM
l0190 REM Interval chooser
l0200 DEF FN£I(N%,LO,HI):LOCAL A,B,C
l0210 IF0>=N%THENPROC£er("Bad no. of intervals")
l0220 C=(HI-LO)/N%:IF0>=C THENPROC£er("Bad axis range")
l0230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/lE6)
l0240 REM normalise interval size ^l0
l0250 A=l0^INT(LOG(C)):B=C/A
l0260 REM select nearest round interval
l0270 IFB>7.lTHENB=l0ELSEIFB>3.2THENB=5ELSEIFB>l.4 THENB=2ELSEB=l
l0280 =A*B
l0290 REM
l0300 REM calc secondary origin,lo,hi
l0310 DEF PROC£ax(O,LO,HI,I)
l0320 REM if axes wld cross out range,
l0330 REM set sec pars so cross at lo,
l0340 REM and ensure orig at integer*intvl
l0350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.l)*I:LO=O
l0360 REM anyway make ends at intg*intvl
l0370 LO=O+I*INT((LO-O)/I+.l)
l0380 HI=O-I*INT((O-HI)/I+.l)
l0390 REM return value in £0,£1,£2
l0400 £0=O:£l=LO:£2=HI:ENDPROC
l0410 REM
l0420 REM set up sector
l0430 DEF PROC£se:LOCAL S%
l0440 S%=(£SN%-l)MOD(£XN%*£YN%)
l0450 £SXS%=£XS% DIV £XN%:£SYS%=£YS% DIV £YN%
l0460 £SXB%=£XB%+S%MOD£XN%*£SXS%
l0470 £SYB%=£YB%+S%DIV£XN%*£SYS%:ENDPROC
l0480 REM
l0490 DEF FN£fa(S%,M%,HI,LO):£0=HI-LO:=(S%-(M%MODl000)-(M%DIVl000))/
£0
l0500 REM
l0510 DEF FN£pp(L,M%)
l0520 £0=FN£pr(M%,0)*L/l00
```

```
10530 ON 1+(FNfpr(M%,2)MOD 3)GOTO 10540,10550,10560
10540 =0.
10550 =-f0
10560 =-f0/2
10570 REM
10580 REM Extract colour
10590 DEF PROCfco(C%,N%):IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
10600 ENDPROC
10610 REM
10620 REM scale or unscale labels
10630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10650 REM digits needed c% before,d% after
10660 IF j%>0 THEN c%=j%+1:ELSE c%=1
10670 IF NOT i%<0 THEN d%=0:p%=0:ELSE d%=-i%:p%=1
10680 REM is sign needed
10690 IF lo<0 OR hi <0 THEN s%=1 ELSEs%=0
10700 REM decide scaled/unscaled
10710 IF NOT(f%<p%+s%+c%+d%)THEN f0=0 ELSE d%=0:p%=0:f0=FNmin(-i
%,f%-j%-s%-1)
10720 REM format; gen if integer el fix
10730 IF d%=0 THEN i%=&10000+f% ELSE i%=&10200+d%
10740 =f%+&100*i%
10750 REM '
10760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEF FNfpr(n%,d%)=(n%DIV(10^d%))MOD100
10790 DEF PROCfer(a$):PRINT "Graphics package error"
10800 PRINT a$:STOP:ENDPROC
10810 REM
12000 REM ------ TWO DIM ROUTINES ------
12010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
12020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X),FNfy(Y):ENDPROC
12030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
12040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y:fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
12050 PLOT K%,x,y:ENDPROC
12060 DEF FNfPOINT(X,Y)=POINT FNfx(X),FNfy(Y)
12070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
12080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
12090 REM two dim scales & axes
12100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,I%,w%,L$,y%
12110 IF M%=1 THEN GOSUB 12140 ELSE IF M%=2 THEN GOSUB 12300 ELSE IF
M%=3 THEN GOSUB 12400 ELSE GOSUB 12140:GOSUB 12300:GOSUB 12400
12120 ENDPROC
12130 REM set intervals & ends
12140 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH)
12150 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
12160 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
12170 REM set sector
12180 PROCfse
12190 REM scale factors,allow margins
12200 fXFA=FNffa(fSXS%,fHM%,fSXH,fSXL):A=f0*fXFA
12210 fYFA=FNffa(fSYS%,fVM%,fSYH,fSYL):B=f0*fYFA
12220 REM equalise if within 30%
12230 f0=ABS((fXFA-fYFA)/(fXFA+fYFA))
12240 IF ABS((fXFA-fYFA)/(fXFA+fYFA))<.15 THEN fXFA=
FNmin(fXFA,fYFA):fYFA=fXFA
12250 REM position the axes cross
12260 fXCR%=fSXB%+(fHM% DIV 1000)+fXFA*(fSXO-fSXL)+(A-fXFA*(fSXH-
```

```
£SXL))/2
12270 £YCR%=£SYB%+(£VM%DIV1000)+£YFA*(£SYO-£SYL)+(B-£YFA*(£SYH-
£SYL))/2
12280 RETURN
12290 REM draw & mark off axes
12300 PROC£co(£XC%,4):PROC£MOVE(£SXL,£SYO):PROC£DRAW(£SXH,£SYO)
12310 PROC£co(£YC%,4):PROC£MOVE(£SXO,£SYL):PROC£DRAW(£SXO,£SYH)
12320 B=FN£pp(£SYI,£XP%):PROC£co(£XC%,2)
12330 FOR A=£SXL TO £SXH+£SXI/2 STEP £SXI
12340 PROC£MOVE(A,£SYO+B):PROC£DRAW(A,£SYO+B+£0):NEXT
12350 A=FN£pp(£SXI,£YP%):PROC£co(£YC%,2)
12360 FOR B=£SYL TO £SYH+£SYI/2 STEP £SYI
12370 PROC£MOVE(£SXO+A,B):PROC£PLOT(1,£0,0):NEXT
12380 RETURN
12390 REM label axes
12400 f%=FN£pr(£XP%,4):w%=£CH%DIV1E3
12410 REM decide scaled or unscaled
12420 a%=@%:@%=FN£su(£SXL,£SXH,£SXI,f%)
12430 REM decide if OK to label all pts
12440 IF£XFA*£SXI>(1+f%)*w%ANDNOT(£M%=2OR£M%=5)THEN1%=TRUE ELSE1%=
FALSE
12450 REM now label
12460 VDU 5:PROC£co(£XC%,0):A=£SXH
12470 IF£0<>0THENL$="E"+STR$(-£0)ELSEL$=""
12480 d%=£SXB%+£SXS%:y%=FN£y(£SYO)+(£XP%DIV1E6)-500
12490 REPEATIFABS(A)<£SXI/2THENA=0
12500 L$=STR$(A*10^£0)+L$:B=LEN(L$)*w%
12510 d%=FNmin(FN£x(A)-B/2,d%-B-w%DIV2)
12520 PLOT 4,d%,y%:PRINTL$:L$=""
12530 IF1%THENA=A-£SXI ELSEA=£SXL:1%=TRUE
12540 UNTILA<£SXL-£SXI/2
12550 REM now label Y-axis
12560 f%=FN£pr(£YP%,4):y%=£CH%MOD1E3:d%=y%+£YP%DIV1E6-500
12570 @%=FN£su(£SYL,£SYH,£SYI,f%)
12580 IF£YFA*£SYI>2*y%ANDNOT(£M%=2OR£M%=5)THEN1%=1ELSE1%=0
12590 PROC£co(£YC%,0):B=£SYL
12600 REPEAT PROC£MOVE(£SXO,B)
12610 IF ABS(B)<£SYI/2 THEN B=0
12630 PLOT0,-w%DIV2,d%:FORA=1TOf%:VDU8:NEXT:PRINTB*10^£0;
12640 IF1%=1THENB=B+£SYI ELSE B=£SYH:1%=1
12650 UNTILB>£SYH+£SYI/2
12660 IF £0<>0 THEN PRINT"E";-£0
12670 VDU 4:@%=a%:RETURN
```

```
    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 : REM L3-CUR
   20 £M%=5:X$="0":Y$="0"
   40 ON ERROR GOTO700
   90 MODE4:VDU19 128,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 £M%=FNp(K%,4,"Mode for graph",£M%)
  120 PROC£INIT(£M%)
  130 PROCc(6)
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
```

```
  170 PRINT TAB(2,4)"Horizontal axis:"
  180 fXL=FNp(K%,5,"Left end",fXL)
  190 fXH=FNp(K%,6,"Right end ",fXH)
  200 PRINT TAB(2,8)"Vertical axis:"
  210 fYL=FNp(K%,9,"Low end ",fYL)
  220 fYH=FNp(K%,10,"High end ",fYH)
  250 K%=1:UNTIL FNe
  260 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
  320 PRINT TAB(2,6)"After axes drawn,use keys:"
  330 PRINT TAB(2,8)"M: move to X,Y"
  332 PRINT"  D: draw to X,Y"
  340 PRINT"  C: colour"
  342 PRINT"  L: change logical colours"
  350 PRINT"  X: input function X(T)"
  352 PRINT"  Y: input function Y(T)"
  354 PRINT"  U: display X(T)at X,Y"
  356 PRINT"  V: display Y(T)at X,Y"
  360 PRINT"  P: T-range to plot,no. of points"
  375 PRINT"  T:title"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=£CH%MOD1E3:£YB%=3*X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AXES(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MOD1E3))TAB(0,29);
  560 A$=GET$
  570 IF A$="M"THENINPUT"Move to X,Y ",X,Y:PROC£MOVE(X,Y)
  572 IF A$="D"THENINPUT"Draw to X,Y ",X,Y:PROC£DRAW(X,Y)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IFA$="X"THENINPUT "Function X(T)",X$
  592 IFA$="Y "THENINPUT"Function Y(T)",Y$
  600 IF A$="P"THENINPUT"T1,T2,N",T1,T2,N%:PROCplot
  610 IF A$="U"THENINPUT"X(T)at
X,Y",X,Y:PROC£MOVE(X,Y):VDU5:PRINT"X(T)=";X$:VDU4
  615 IF A$="V"THENINPUT"Y(T)at
X,Y",X,Y:PROC£MOVE(X,Y):VDU5:PRINT"Y(T)=";Y$:VDU 4
  620 IF A$=
"T"THENINPUT"Title,X,Y",A$,X,Y:PROC£MOVE(X,Y):VDU5:PRINTA$:VDU 4
  630 IF A$=" ? "THENPRINT "M,D,C,L,X,Y,U,V,P,T":X=INKEY(500)
  640 GOTO 550
  690 REM error branch
  700 MODE 7:IF ERR=17 THEN 900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat ?(S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R" THEN90
  940 GOTO 910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =" ;V;
 1020 IF M%>0 THEN PRINT" ? " ;:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
```

```
 1060 GOTO 1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"CURVE PLOTTING UTILITY";SPC(5);"PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =FALSE
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCc(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FNfpr(fXC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FNfpr(fXC%,4):B%=FNfpr(fXC%,2)
 1320 C%=FNfpr(fXC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
 1350 C%=FNp(K%,L%+4,"labels",C%)
 1360 fXC%=((100+A%)*100+B%)*100+C%
 1370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 1380 ENDPROC
 2000 DEF PROCplot:LOCAL D
 2005 D=(T2-T1)/N%:T=T1
 2010 PROCfMOVE(EVAL(X$),EVAL(Y$))
 2020 FOR T=T1 TO T2+D/2 STEP D
 2030 PROCfDRAW(EVAL(X$),EVAL(Y$)):NEXT
 2040 ENDPROC
10040 REM
10050 REM Initialisation
10060 DEF PROCfINIT(M%):fM%=M%
10070 fXL=-10:fXH=10:fYL=-10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1:fYN%=1:fSN%=1
10100 IF M%=2 OR M%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=1010203 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FNfI(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROCfer("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10240 REM normalise interval size ^10
10250 A=10^INT(LOG(C)):B=C/A
10260 REM select nearest round interval
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10290 REM
10300 REM calc secondary origin,lo,hi
10310 DEF PROCfax(O,LO,HI,I)
10320 REM if axes wld cross out range,
10330 REM set sec pars so cross at lo,
10340 REM and ensure orig at integer*intvl
```

```
10350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.1)*I:LO=O
10360 REM anyway make ends at intg*intvl
10370 LO=O+I*INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10390 REM return value in f0,f1,f2
10400 f0=O:f1=LO:f2=HI:ENDPROC
10410 REM
10420 REM set up sector
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%:fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIVfXN%*fSYS%:ENDPROC
10480 REM
10490 DEF FNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)-(M%DIV1000))/
f0
10500 REM
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2)MOD 3)GOTO 10540,10550,10560
10540 =0.
10550 =-f0
10560 =-f0/2
10570 REM
10580 REM Extract colour
10590 DEF PROCfco(C%,N%):IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
10600 ENDPROC
10610 REM
10620 REM scale or unscale labels
10630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10650 REM digits needed c% before,d% after
10660 IF j%>0 THEN c%=j%+1:ELSE c%=1
10670 IF NOT i%<0 THEN d%=0:p%=0:ELSE d%=-i%:p%=1
10680 REM is sign needed
10690 IF lo<0 OR hi <0 THEN s%=1 ELSEs%=0
10700 REM decide scaled/unscaled
10710 IF NOT(f%<p%+s%+c%+d%)THEN f0=0 ELSE d%=0:p%=0:f0=FNmin(-
i%,f%-j%-s%-1)
10720 REM format; gen if integer el fix
10730 IF d%=0 THEN i%=&10000+f% ELSE i%=&10200+d%
10740 =f%+&100*i%
10750 REM
10760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEF FNfpr(n%,d%)=(n%DIV(10^d%))MOD100
10790 DEF PROCfer(a$):PRINT "Graphics package error"
10800 PRINT a$:STOP:ENDPROC
10810 REM
12000 REM ------ TWO DIM ROUTINES -------
12010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
12020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X),FNfy(Y):ENDPROC
12030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
12040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y:fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
12050 PLOT K%,x,y:ENDPROC
12060 DEF FNfPQfNT(X,Y)=POINT FNfx(X),FNfy(Y)
12070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
12080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
12090 REM two dim scales & axes
```

```
12100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
12110 IF M%=1 THEN GOSUB 12140 ELSE IF M%=2 THEN GOSUB 12300 ELSE IF
M%=3 THEN GOSUB 12400 ELSE GOSUB 12140:GOSUB 12300:GOSUB 12400
12120 ENDPROC
12130 REM set intervals & ends
12140 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH)
12150 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
12160 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
12170 REM set sector
12180 PROCfse
12190 REM scale factors,allow margins
12200 fXFA=FNffa(fSXS%,fHM%,fSXH,fSXL):A=f0*fXFA
12210 fYFA=FNffa(fSYS%,fVM%,fSYH,fSYL):B=f0*fYFA
12220 REM equalise if within 30%
12240 IF ABS((fXFA-fYFA)/(fXFA+fYFA))<.15 THEN fXFA=
FNmin(fXFA,fYFA):fYFA=fXFA
12250 REM position the axes cross
12260 fXCR%=fSXB%+(fHM% DIV 1000)+fXFA*(fSXO-fSXL)+(A-fXFA*(fSXH-
fSXL))/ 2
12270 fYCR%=fSYB%+(fVM%DIV1000)+fYFA*(fSYO-fSYL)+(B-fYFA*(fSYH-
fSYL))/2
12280 RETURN
12290 REM draw & mark off axes
12300 PROCfco(fXC%,4):PROCfMOVE(fSXL,fSYO):PROCfDRAW(fSXH,fSYO)
12310 PROCfco(fYC%,4):PROCfMOVE(fSXO,fSYL):PROCfDRAW(fSXO,fSYH)
12320 B=FNfpp(fSYI,fXP%):PROCfco(fXC%,2)
12330 FOR A=fSXL TO fSXH+fSXI/2 STEP fSXI
12340 PROCfMOVE(A,fSYO+B):PROCfDRAW(A,fSYO+B+f0):NEXT
12350 A=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
12360 FOR B=fSYL TO fSYH+fSYI/2 STEP fSYI
12370 PROCfMOVE(fSXO+A,B):PROCfPLOT(1,f0,0):NEXT
12380 RETURN
12390 REM label axes
12400 f%=FNfpr(fXP%,4):w%=fCH%DIV1E3
12410 REM decide scaled or unscaled
12420 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,f%)
12430 REM decide if OK to label all pts
12440 IFfXFA*fSXI>(1+f%)*w%ANDNOT(fM%=2ORfM%=5)THENl%=TRUE ELSEl%=
FALSE
12450 REM now label
12460 VDU 5:PROCfco(fXC%,0):A=fSXH
12470 IFf0<>0THENL$="E"+STR$(-f0)ELSEL$=""
12480 d%=fSXB%+fSXS%:y%=FNfy(fSYO)+(fXP%DIV1E6)-500
12490 REPEATIFABS(A)<fSXI/2THENA=0
12500 L$=STR$(A*10^f0)+L$:B=LEN(L$)*w%
12510 d%=FNmin(FNfx(A)-B/2,d%-B-w%DIV2)
12520 PLOT4,d%,y%:PRINTL$:L$=""
12530 IFl%THENA=A-fSXI ELSEA=fSXL:l%=TRUE
12540 UNTILA<fSXL-fSXI/2
12550 REM now label Y-axis
12560 f%=FNfpr(fYP%,4):y%=fCH%MOD1E3:d%=y%+fYP%DIV1E6-500
12570 @%=FNfsu(fSYL,fSYH,fSYI,f%)
12580 IFfYFA*fSYI>2*y%ANDNOT(fM%=2ORfM%=5)THENl%=1ELSEl%=0
12590 PROCfco(fYC%,0):B=fSYL
12600 REPEAT PROCfMOVE(fSXO,B)
12610 IF ABS(B)<fSYI/2 THEN B=0
12620 REMIFB>fSYH-fSYI/2THENl%=2:d%=y%+(fYP%DIV1E6)-500
12630 PLOT0,-w%DIV2,d%:FORA=1TOf%:VDU8:NEXT:PRINTB*10^f0;
12640 IFl%=1THENB=B+fSYI ELSE B=fSYH:l%=1
12650 UNTILB>fSYH+fSYI/2
```

```
12660 IF £0<>0 THEN PRINT"E";-£0
12670 VDU 4:@%=a%:RETURN


    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 :REM L3-PIE
   20 £M%=5
   40 ON ERROR GOTO 700
   90 MODE4: VDU19 128,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 £M%=FNp(K%,4,"Mode for chart",£M%)
  120 PROC£INIT(£M%)
  140 K%=1:UNTIL FNe
  150 REM
  260 REM
  300 PROCh(2)
  310 PRINT TAB(2,4)"READY TO DRAW CHART"
  320 PRINT TAB(2,6)"After screen cleared,use keys to set:"
  330 PRINT TAB(2,8)"S: starting angle for next sector"
  332 PRINT"  D: draw next sector,size in %"
  340 PRINT"  C: colour"
  342 PRINT"  L: change logical colours"
  375 PRINT"  T: title"
  378 PRINT"  W: write sector size in colour 0"
  379 PRINT"     of the last sector drawn."
  382 PRINT"  ?: display list of prompt keys"
  388 PRINTTAB(2,17)"For pie charts,axes are not drawn,"
  390 PRINT"  but the screen is addressed as if"
  400 PRINT"  there were axes with origin at the"
  410 PRINT"  centre of the screen. "
  415 PRINT"  Each axis spans from -10 to +10."
  420 PRINT"  The pie radius is 8 in these units."
  428 PRINT
  430 PRINT"  Angles must be given in percent,"
  440 PRINT"  where 100% gives a full circle. "
  445 PRINT"  The starting angle is 0 initially."
  450 PRINTTAB(2,29)"press";:GOSUB 1270
  460 A$=GET$:IF NOT(A$=" ")THEN450
  500 X=£CH%MOD1E3:£YB%=3*X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AXES(1):A=0:C%=1:GCOL0,1
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MOD1E3))TAB(0,29);
  560 A$=GET$
  570 IF A$="S"THENINPUT"Starting angle",A
  572 IF A$="D"THENINPUT"Draw sector of %",P:B=A+
P:PROC£SEC(11,0,0,8,A/100,B/100):A=B
  580 IF A$="C"THENINPUT"Colour ",C%:GCOL0,C%
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IF A$="W"THEN PROCw
  620 IF A$=
"T"THENINPUT"Title,X,Y",A$,X,Y:PROC£MOVE(X,Y):VDU5:PRINTA$:VDU4
  630 IF A$="?"THENPRINT"S,D,C,L,W,T" :X=INKEY(500)
  640 GOTO 550
  690 REM error branch
  700 MODE7:IF ERR=17 THEN900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat?(S/R)";
  910 A$=GET$
```

```
  920 IF A$="S" THENPRINT:END
  930 IF A$="R " THEN 90
  940 GOTO 910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"PIE CHART UTILITY";SPC(5); "PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =£
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 2000 DEF PROCw
 2010 A%=@%:@%=&01000202
 2020 GCOL0,0:VDU5:VDU8:PRINT P:VDU 4
 2030 @%=A%:GCOL0,C%:ENDPROC
10040 REM
10050 REM Initialisation
10060 DEF PROC£INIT(M%): £M%=M%
10070 £XL=-10:£XH=10:£YL=-10:£YH=10 :£ZL=-10:£ZH=10
10080 £XO=0:£YO=0:£ZO=0: £XI%=7:£YI%=7:£ZI%=7
10090 £XN%=1:£YN%=1:£SN%=1
10100 IF M%=2 OR M%=5 THEN £CH%=64 ELSE £CH%=32
10110 £YB%=0:£YS%=1023:£XB%=3*£CH%: £XS%=1279-£XB%
10120 £VM%=64064:£HM%=2*(1000*£CH%+£CH%):£CH%=1000*£CH%+32
10130 £XP%=484040225:£YP%=504040225:£ZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN £XC%=1010203 ELSE £XC%=
10150 £YC%=0:£ZC%=0
10160 £TH=1.3:£PH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FN£I(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROC£er("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROC£er("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10240 REM normalise interval size ''10
10250 A=10^INT(LOG(C)):B=C/A
10260 REM select nearest round interval
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10290 REM
10300 REM calc secondary origin,lo,hi
10310 DEF PROC£ax(O,LO,HI,I)
10320 REM if axes wld cross out range,
10330 REM set sec pars so cross at lo,
10340 REM and ensure orig at integer*intvl
10350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.1)*I:LO=O
10360 REM anyway make ends at intg*intvl
10370 LO=O+I*INT((LO-O)/I+.1)
```

```
10380 HI=O-I*INT((O-HI)/I+.1)
10390 REM return value in f0,f1,f2
10400 f0=O:f1=LO:f2=HI:ENDPROC
10410 REM
10420 REM set up sector
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%:fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIV fXN%*fSYS%:ENDPROC
10480 REM
10490 DEF FNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)-(M%DIV1000))/
f0
10500 REM
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2)MOD 3)GOTO 10540,10550,10560
10540 =0.
10550 =-f0
10560 =-f0/2
10570 REM
10580 REM Extract colour
10590 DEF PROCfco(C%,N%): IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
10600 ENDPROC
10610 REM
10620 REM scale or unscale labels
10630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10650 REM digits needed c% before,d% after
10660 IF j%>0 THEN c%=j%+1: ELSE c%=1
10670 IF NOT i%<0 THEN d%=0:p%=0: ELSE d%=-i%:p%=1
10680 REM is sign needed
10690 IF lo<0 OR hi <0 THEN s%=1 ELSEs%=0
10700 REM decide scaled/unscaled
10710 IF NOT(f%<p%+s%+c%+d%)THEN f0=0 ELSE d%=0:p%=0:f0=FNmin(-
i%,f%-j%-s%-1)
10720 REM format; gen if integer el fix
10730 IF d%=0 THEN i%=&10000+f% ELSE i%=&10200+d%
10740  =f%+&100*i %
10750 REM
10760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEF FNfpr(n%,d%)=(n%DIV(10''d%))MODIOO
10790 DEF PROCfer(a$):PRINT "Graphics package error"
10800 PRINT a$:STOP:ENDPROC
10810 REM
12000 REM ------ TWO DIM ROUTINES -------
12010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
12020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X),FNfy(Y):ENDPROC
12030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
12040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y :fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
12050 PLOT K%,x,y:ENDPROC
12060 DEF FNfPOINT(X,Y)=POINT FNfx(X),FNfy(Y)
12070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
12080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
12090 REM two dim scales & axes
12100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
12110 IF M%=1 THEN GOSUB 12140 ELSE IF M%=2 THEN GOSUB 12300 ELSE IF
M%=3 THEN GOSUB 12400 ELSE GOSUB 12140:GOSUB 12300:GOSUB 12400
```

```
l2120 ENDPROC
l2130 REM set intervals & ends
l2140 £SXI=FN£I(£XI%,£XL,£XH): £SYI=FN£I(£YI%,£YL,£YH)
l2150 PROC£ax(£XO,£XL,£XH,£SXI):£SXO=£0: £SXL=£l:£SXH=£2
l2160 PROC£ax(£YO,£YL,£YH,£SYI): £SYO=£0:£SYL=£l: £SYH=£2
l2170 REM set sector
l2180 PROC£se
l2190 REM scale factors,allow margins
l2200 £XFA=FN£fa(£SXS%,£HM%,£SXH,£SXL):A=£0*£XFA
l2210 £YFA=FN£fa(£SYS%,£VM%,£SYH,£SYL):B=£0*£YFA
l2220 REM equalise if within 30%
l2230 £0=ABS((£XFA-£YFA)/(£XFA+£YFA))
l2240 IF ABS((£XFA-£YFA)/(£XFA+£YFA))<.l5 THEN £XFA=
FNmin(£XFA,£YFA):£YFA=£XFA
l2250 REM position the axes cross
l2260 £XCR%=£SXB%+(£HM% DIV l000)+£XFA*(£SXO-£SXL)+(A-£XFA*(£SXH-
£SXL))/2
l2270 £YCR%=£SYB%+(£VM%DIVl000)+£YFA*(£SYO-£SYL)+(B-£YFA*(£SYH-
£SYL))/ 2
l2280 RETURN
l2290 REM draw & mark off axes
l2300 PROC£co(£XC%,4):PROC£MOVE(£SXL,£SYO):PROC£DRAW(£SXH,£SYO)
l2310 PROC£co(£YC%,4):PROC£MOVE(£SXO,£SYL):PROC£DRAW(£SXO,£SYH)
l2320 B=FN£pp(£SYI,£XP%):PROC£co(£XC%,2)
l2330 FOR A=£SXL TO £SXH+£SXI/2 STEP £SXI
l2340 PROC£MOVE(A,£SYO+B):PROC£DRAW(A,£SYO+B+£0):NEXT
l2350 A=FN£pp(£SKI,£YP%): PROC£co(£YC%,2)
l2360 FOR B=£SYL TO £SYH+£SYI/2 STEP £SYI
l2370 PROC£MOVE(£SXO+A,BJ):PROCF-PLOT(l,£0,0):NEXT
l2380 RETURN
l2390 RE& label axes
l2400 f%=FN£pr(£XP%,4):w%=£CH%DIVlE3
l2410 REM decide scaled or unscaled
l2420 a%=@%:@%=FN£su(£SXL,£SXH,£SXI,f%)
l2430 REM decide if OK to label all pts
l2440 IF£XFA*£SXI>(l+f%)*w%ANDNOT(£M%=2OR£M%=5)THENl%=TRUEELSEl%=£
l2450 REM now label
l2460 VDU 5:PROC£co(£XC%,0):A=£SXH
l2470 IF£0<>0THENL$="E"+STR$(-£0)ELSEL$=""
l2480 d%=£SXB%+£SXS% :y%=FN£y(£SYO)+(£XP%DIVlE6)-500
l2490 REPEATIFABS(A)<£SXI/2THENA=0
l2500 L$=STR$(A*l0^£0)+L$:B=LEN(L$)*w%
l2510 d%=FNmin(FN£x(A)-B/2,d%-B-w%DIV2)
l2520 PLOT4,d%,y%:PRINTL$:L$=""
l2530 IFl%THENA=A-£SXI ELSEA=£SXL:l%=TRUE
l2540 UNTILA<£SXL-£SXI/2
l2550 REM now label Y-axis
l2560 f%=FN£pr(£YP%,4):y%=£CH%MODlE3: d%=y%+£YP%DIVlE6-500
l2570 @%=FN£su(£SYL,£SYH,£SYI,f%)
l2580 IF£YFA*£SYI>2*y%ANDNOT(£M%=2OR£M%=5)THENl%=lELSEl%=0
l2590 PROC£co(£YC%,0):B=£SYL
l2600 REPEAT PROC£MOVE(£SXO,B)
l2610 IF ABS(B)<£SYI/2 THEN B=0
l2630 PLOT0,-w%DIV2,d%:FORA=lTOf%:VDU8:NEXT:PRINTB*l0^£0;
l2640 IFl%=lTHENB=B+£SYI ELSE B=£SYH:l%=l
l2650 UNTILB>£SYH+£SYI/2
l2660 IF £0<>0 THEN PRINT"E";-£0
l2670 VDU 4:@%=a%:RETURN
l5180 REM -------- Circular sector ---
l5190 DEF PROC£SEC(M%,X,Y,R,A,B):LOCAL a,b,I%
```

```
15200 REM tens digit of M%: center
15210 REM 0=screen,1-user
15220 IF M%DIV10=1 THEN X=FN£x(X):Y=FN£y(Y)
15230 REM ones digit of M%: radius
15240 IF M%MOD10=1 THEN R=R*£YFA
15250 M%=1+50*ABS(B-A):b=(B-A)*2*PI/M%
15260 A=2*PI*A:PLOT 84,X+R*COS(A),Y+R*SIN(A)
15270 FOR I%=1 TO M%:a=A+I%*b:PLOT 84,X,Y
15280 PLOT 81,R*COS(a),R*SIN(a):NEXT
15290 a=A+M%*b/2:MOVE X+R*COS(a)*.7,Y+R*SIN(a)*.7
15300 ENDPROC
15310 REM ----------------------------
```

```
    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 :REM L3-CV3D
   20 £M%=5:X$="0":Y$="0":Z$="0"
   40 ON ERROR GOTO 700
   90 MODE4:VDU19 128,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 £M%=FNp(K%,4,"Mode for graph",£M%)
  120 PROC£INIT(£M%)
  130 PROCC(6)
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
  170 PRINTTAB(2,4)"X-axis:"
  180 £XL=FNp(K%,5,"Low end",£XL)
  190 £XH=FNp(K%,6,"High end",£XH)
  200 PRINTTAB(2,8)"Y-axis:"
  210 £YL=FNp(K%,9,"Low end ",£YL)
  220 £YH=FNp(K%,10,"High end",£YH)
  230 PRINTTAB(2,12)"Z-axis:"
  235 £ZL=FNp(K%,13,"Low end ",£ZL)
  240 £ZH=FNp(K%,14,"High end",£ZH)
  260 PRINTTAB(2,16)"Viewing angles in degrees:"
  270 £TH=PI*FNp(K%,17,"Theta",180*£TH/PI)/180
  272 £PH=PI*FNp(K%,18,"Phi ",180*£PH/PI)/180
  280 K%=1:UNTIL FNe
  290 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
  320 PRINTTAB(2,6)"After axes drawn,use keys:"
  325 PRINTTAB(2,8)"A:change viewing angles"
  330 PRINT"  M: move to X,Y,Z"
  332 PRINT"  D: draw to X,Y,Z"
  340 PRINT"  C: colour"
  342 PRINT"  L: change logical colours"
  350 PRINT"  X: input function X(T)"
  352 PRINT"  Y: input function Y(T)"
  354 PRINT"  Z: input function Z(T)"
  360 PRINT"  P: T-range to plot,no. of points"
  375 PRINT"  T: title"
  376 PRINT"  U: print X(T)at X,Y,Z"
  377 PRINT"  V: print Y(t)"
  378 PRINT"  W: print Z(t)"
  379 PRINT"  ?:list of prompts"
```

```
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=£CH%MODlE3:£YB%=3 *X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AX3(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MODlE3))TAB(0,29);
  560 A$=GET$
  570 IF A$="M"THENINPUT"Move to X,Y,Z ",X,Y,Z:PROC£MV3(X,Y,Z)
  572 IF A$="D"THENINPUT"Draw to X,Y,Z ",X,Y,Z:PROC£DR3(X,Y,Z)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IFA$="X"THENINPUT "Function X(T)",X$
  592 IFA$="Y"THENINPUT "Function Y(T)",Y$
  594 IFA$="Z "THENINPUT"Function Z(T)",Z$
  600 IF A$="P"THENINPUT"Tl,T2,N",Tl,T2,N%:PROCplot
  620 IF A$="T"THENINPUT"Ti
tle,X,Y,Z",A$,X,Y,Z:PROC£MV3(X,Y,Z):VDU5:PRINTA$:VDU4
  630 IF A$="U"THEN PROCfn("X")
  640 IF A$="V"THEN PROCfn("Y")
  650 IF A$="W"THEN PROCfn("Z")
  660 IF A$="A"THENINPUT"Theta,Phi(deg)",X,Y:£TH=PI*X/180:£PH=PI*Y/
180:CLG:PROC£AX3(0)
  670 IF A$="?"THENPRINT" A,M,D,C,L,P,T,U,V,W,X,Y,Z":A$=INKEY$(500)
  680 GOTO 550
  690 REM error branch
  700 MODE 7:IF ERR=17 THEN 900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat?(S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R" THEN 90
  940 GOTO910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"3-D CURVE PLOTTER";SPC(5); "PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =£
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCC(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FN£pr(£XC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FN£pr(£XC%,4):B%=FN£pr(£XC%,2)
 1320 C%=FN£pr(£XC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
```

```
 1350 C%=FNp(K%,L%+4,"labels",C%)
 1360 £XC%=((100+A%)*100+B%)*100+C%
 1370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 1380 ENDPROC
 1400 DEF PROCfn(F$)
 1410 PRINTF$;"(T)at X,Y,Z";:INPUT,X,Y,Z
 1420 PROC£MV3(X,Y,Z):VDU5
 1430 PRINTF$; "(T)=";EVAL(F$+"$")
 1440 VDU4:ENDPROC
 2000 DEF PROCplot:LOCAL D
 2005 D=(T2-T1)/N%:T=T1
 2010 PROC£MV3(EVAL(X$),EVAL(Y$),EVAL(Z$))
 2020 FOR T=T1 TO T2+D/2 STEP D
 2030 PROC£DR3(EVAL(X$),EVAL(Y$),EVAL(Z$)):NEXT
 2040 ENDPROC
10060 DEFPROC£INIT(M%):£M%=M%
10070 £XL=-10:£XH=10:£YL=-10:£YH=10:£ZL=-10:£ZH=10
10080 £XO=0:£YO=0:£ZO=0:£XI%=7:£YI%=7:£ZI%=7
10090 £XN%=1:£YN%=1:£SN%=1
10100 IFM%=2ORM%=5THEN£CH%=64ELSE£CH%=32
10110 £YB%=0:£YS%=1023:£XB%=3*£CH%:£XS%=1279-£XB%
10120 £VM%=64064:£HM%=2*(1000*£CH%+£CH%):£CH%=1000*£CH%+32
10130 £XP%=484040225:£YP%=504040225:£ZP%=540040225
10140 IFM%=1ORM%=2ORM%=5THEN£XC%=1010203ELSE£XC%=0
10150 £YC%=0:£ZC%=0
10160 £TH=1.3:£PH=-1.3
10170 ENDPROC
10200 DEFFN£I(N%,LO,HI):LOCALA,B,C
10210 IF0>=N%THENPROC£er("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROC£er("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10250 A=10^INT(LOG(C)):B=C/A
10270 IFB>7.1THENB=1 0ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10310 DEF PROC£ax(O,LO,HI,I)
10350 IFLO-I/2>=O ORO>=HI THENO=INT(LO/I-.1)*I:LO=O
10370 LO=O+I*INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10400 £0=O:£1=LO:£2=HI:ENDPROC
10430 DEF PROC£se:LOCAL S%
10440 S%=(£SN%-1)MOD(£XN%*£YN%)
10450 £SXS%=£XS%DIV£XN%:£SYS%=£YS%DIV£YN%
10460 £SXB%=£XB%+S%MOD£XN%*£SXS%
10470 £SYB%=£YB%+S%DIV£XN%*£SYS%:ENDPROC
10490 DEFFN£fa(S%,M%,HI,LO):£0=HI-LO:=(S%-(M%MOD1000)(M%DIV1000))/£0
10510 DEF FN£pp(L,M%)
10520 £0=FN£pr(M%,0)*L/100
10530 ON1+(FN£pr(M%,2)MOD3)GOTO10540,10550,10560
10540 =0.
10550 =-£0
10560 =-£0/2
10590 DEFPROC£co(C%,N%):IFFN£pr(C%,6)>0THENGCOL0,FN£pr(C%,N%)
10600 ENDPROC
10630 DEFFN£su(lo,hi,I,f%):LOCALi%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10660 IFj%>0THENc%=j%+1:ELSEc%=1
10670 IFNOTi%<0THENd%=0:p%=0:ELSEd%=-i%:p%=1
10690 IFlo<0ORhi <0THENs%=1ELSEs%=0
10710 IFNOT(f%<p%+s%+c%+d%)THEN£0=0ELSEd%=0:p%=0:£0=FNmin(-i%,f%-j%-
s%-1)
```

```
l0730 IFd%=0THENi%=&l0000 +f%ELSEi%=&l0200+d%
l0740 =f%+&l00*i%
l0760 DEFFNmax(a,b):IFb>a THEN=b ELSE=a
l0770 DEFFNmin(a,b):IFb<a THEN=b ELSE=a
l0780 DEFFNfpr(n%,d%)=(n%DIV(l0^d%))MODl00
l0790 DEFPROCfer(a$):PRINT"Graphics package error"
l0800 PRINTa$:STOP:ENDPROC
l3010 DEFPROCfMV3(X,Y,Z):PROCfPL3(4,X,Y,Z):ENDPROC
l3020 DEFPROCfDR3(X,Y,Z):PROCfPL3(5,X,Y,Z):ENDPROC
l3030 DEFPROCfPL3(K%,X,Y,Z):LOCAL RX,RY,RZ,x%,y%
l3040 IF3<K%MOD8THENRX=fSXO:RY=fSYO:fRZ =fSZO:x%=fXCR%:y%=
fYCR%:fXSC%=0:fYSC%=0
l3050 RX=(X-RX)*fXFA:RY=(Y-RY)*fYFA:RZ=(Z-RZ)*fZFA
l3060 x%=x%+fPll*RX+fPl2*RY
l3070 y%=y%+fP2l *RX+fP22*RY+fP23*RZ
l3080 fXSC%=fXSC%+x%:fYSC%=fYSC%+y%
l3090 PLOTK%,x%,y%:ENDPROC
l3100 DEFFNfPT3(X,Y,Z):PROCfPL3(4,X,Y,Z):=POINT(fXSC%,fYSC%)
l3130 DEFPROCfAX3(M%):LOCALPl,P2,P3,P4,P5,P6,LM%,DM%,a%
l3140 IFM%=lTHENGOSUBl3l70ELSEIFM%=2 THENGOSUBl3560ELSEIFM%=3
THENGOSUBl3700ELSEGOSUBl3l70:GOSUBl3560:GOSUBl3700
l3150 ENDPROC
l3170 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH):fSZI=
FNfI(fZI%,fZL,fZH)
l3180 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=fl:fSXH=f2
l3190 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=fl:fSYH=f2
l3200 PROCfax(fZO,fZL,fZH,fSZI):fSZO=f0:fSZL=fl:fSZH=f2
l3220 PROCfse
l3240 P4=fSXH-fSXL:P5=fSYH-fSYL:P6=fSZH-fSZL
l3260 LM%=fHM%DIVlE3:DM%=fVM%DIVlE3
l3270 fl=fSXS%-LM%-fHM%MODlE3:Pl=fl/P4:P2=fl/P5
l3280 f2=fSYS%-DM%-fVM%MODlE3:P3=f2/P6
l3300 IFABS((Pl-P2)/(Pl+P2))<.4THENPl=FNmin(Pl,P2):P2=Pl
l3310 IFABS((Pl-P3)/(Pl+P3))>.2THENl3340
l3320 IFPI <P3 THENP3=Pl ELSEf0=Pl:Pl=P3:IFf0=P2 THENP2=P3
l3340 f0=COS(fTH):fPll=-SIN(fPH):fP22=f0*fPll
l3350 fPl2=COS(fPH):fP2l=-f0*fPl2:fP23=SIN(fTH)
l3370 P6=Pl*P4*ABS(fP2l)+P2*P5*ABS(fP22)+P3*P6*ABS(fP23)
l3380 P5=Pl*P4*ABS(fPll)+P2*P5*ABS(fPl2)
l3400 f0=FNmin(fl/P5,f2/P6)
l3410 fXFA=f0*Pl:fYFA=f0 *P2:fZFA=f0*P3:P5=f0*P5:P6=f0*P6
l3430 IFfPll<0THENPl=fSXH ELSEPl=fSXL
l3440 IFfPl2<0THENP2=fSYH ELSEP2=fSYL
l3450 P3=(P2-fSYO)*fYFA*fPl2+(Pl-fSXO)*fXFA*fPll
l3470 IFfP2l<0THENPl=fSXH ELSEPl=fSXL
l3480 IFfP22<0THENP2=fSYH ELSEP2=fSYL
l3490 IFfP23<0THENP4=fSZH ELSEP4=fSZL
l3500 P4=(Pl-fSXO)*fXFA*fP2l+(P2-fSYO)*fYFA*fP22+(P4-fSZO)*fZFA*fP23
l3520 fXCR%=fSXB%+LM%+(fl-P5)/2-P3
l3530 fYCR%=fSYB%+DM%+(f2-P6)/2-P4
l3540 RETURN
l3560
PROCfco(fXC%,4):PROCfMV3(fSXL,fSYO,fSZO):PROCfDR3(fSXH,fSYO,fSZO)
l3570
PROCfco(fYC%,4):PROCfMV3(fSXO,fSYL,fSZO):PROCfDR3(fSXO,fSYH,fSZO)
l3580
PROCfco(fZC%,4):PROCfMV3(fSXO,fSYO,fSZL):PROCfDR3(fSXO,fSYO,fSZH)
l3590 B=FNfpp(fSZI,fXP%):PROCfco(fXC%,2)
l3600 FORA=fSXL TOfSXH+fSXI/2STEPfSXI
l3610 PROCfMV3(A,fSYO,fSZO+B):PROCfDR3(A,fSYO,fSZO+B+f0):NEXT
l3620 B=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
```

```
13630 FORA=£SYL TO£SYH+£SYI/2STEP£SYI
13640 PROC£MV3(£SXO+B,A,£SZO):PROC£DR3(£SXO+B+£0,A,£SZO):NEXT
13650 B=FN£pp(£SXI,£ZP%):PROC£co(£ZC%,2)
13660 FORA=£SZL TO£SZH+£SZI/2STEP£SZI
13670 PROC£MV3(£SXO+B,£SYO,A):PROC£DR3(£SXO+B+£0,£SYO,A):NEXT
13680 RETURN
13700 Pl=FN£pr(£XP%,4):P2=(£XP%DIVlE6)-500:P3=£CH%DIVlE3
13720 a%=@%:@%=FN£su(£SXL,£SXH,£SXI,Pl)
13740 VDU5:PROC£co(£XC%,0):PROC£MV3(£SXH,£SYO,£SZO)
13750 PLOTO,0,P2:FORP4=lTOPl DIV2:VDU8:NEXT:PRINT£SXH*l0^£0:IF£0 <>
0THENPRINT"E";-£0
13760 Pl=FN£pr(£YP%,4):F2=(£YP%DIVlE6)-500
13770 @%=FN£su(£SYL,£SYH,£SYI,Pl)
13780 PROC£co(£YC%,0):PROC£MV3(£SXO,£SYH,£SZO)
13790 PLOT0,0,P2:PRINT£SYH*l0''£0:IF£0<>0THENPRINT"E";-£0
13800 Pl=FN£pr(£ZP%,4):P2=(£ZP%DIVlE6)-500
13810 @%=FN£su(£SZL,£SZH,£SZI,Pl)
13820 PROC£co(£ZC%,0):PROC£MV3(£SXO,£SYO,£SZH)
13830 PLOT 0,0,P2:FORP4=lTOPl DIV2:VDU8:NEXT:PRINT£SZH*l0^£0:IF£0<>
0THENPRINT"E";-£0
13840 VDU4:@%=a%:RETURN
```

```
    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 :REM L3-CO2D
   20 £M%=5:IM%=10:JM%=10
   30 F$="0":DIM £W(IM%,JM%)
   40 ON ERROR GOTO 700
   90 MODE4:VDU19 l28,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 £M%=FNp(K%,4,"Mode for graph",£M%)
  112 IF £M%<4 THEN PRINTTAB(2,5)"No room for MODE";£M%:GOTO 110
  120 PROC£INIT(£M%)
  130 PROCc(6)
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
  170 PRINTTAB(2,4)"Horizontal axis:"
  180 £XL=FNp(K%,5,"Left end",£XL)
  190 £XH=FNp(K%,6,"Right end",£XH)
  200 PRINTTAB(2,8)"Vertical axis:"
  210 £YL=FNp(K%,9,"Low end",£YL)
  220 £YH=FNp(K%,10,"High end",£YH)
  250 K%=1:UNTIL FNe
  260 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
  320 PRINTTAB(2,6)"After axes drawn,use keys:"
  342 PRINT"  L: change logical colours"
  350 PRINT"  F: input function F(X,Y)"
  360 PRINT"  P: plot contours"
  375 PRINT"  T: title "
  377 PRINT"  U: print fn at X,Y"
  379 PRINT"  ?: list of prompts"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
```

```
  500 X=£CH%MODlE3:£YB%=3 *X:£YS%=l023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AXES(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(l280DIV(£CH%MODlE3))TAB(0,29);
  560 A$=GET$
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IF A$="F"THENINPUT "Function F(X,Y)",F$:PROCw
  600 IF A$="P "THENINPUT"Zl, Z2, N ",HI,B2,N%:PROCplot
  620 IF A$=
"T"THENINPUT"Title,X,Y,Z",A$,X,Y,Z:PROC£MV3(X,Y,Z):VDU5:PRINTA$:VDU4
  630 IF A$="?"THENPRINT"C,L,P,T,F,U":A$=INKEY$(300)
  680 GOTO550
  690 REM error branch
  700  MODE7:IF ERR=17 THEN900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat? (S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R " THEN 90
  940 GOTO910
  950 REM
 l000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 l010 PRINT TAB(2,L%);P$;" =";V;
 l020 IF M%>0 THEN PRINT" ? ";:P%=POS
 l030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 l040 INPUT TAB(P%,L%)V$
 l050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 l060 GOTOl0l0
 l100 DEF PROCh(N%):CLS
 l120 PRINT TAB(2,2);"3-D CONTOURS";SPC(5); "PAGE ";N%
 l130 ENDPROC
 l200 DEF FNe:LOCAL A$
 l210 PRINT TAB(0,29)"Type C to change data,";:GOSUB l270
 l220 A$=GET$
 l230 IF A$=" " THEN =TRUE
 l240 IF A$="C" THEN =FALSE
 l250 GOTO l220
 l260 REM
 l270 PRINT" SPACE to continue":RETURN
 l300 DEF PROCc(L%)
 l302 PRINTTAB(2,L%)"Logical colours for:"
 l304 IF FN£pr(£XC%,6)=0 THEN A%=l:B%=l:C%=l:GOTOl330
 l310 A%=FN£pr(£XC%,4):B%=FN£pr(£XC%,2)
 l320 C%=FN£pr(£XC%,0)
 l330 A%=FNp(K%,L%+2,"axes  ",A%)
 l340 B%=FNp(K%,L%+3,"pips  ",B%)
 l350 C%=FNp(K%,L%+4,"labels",C%)
 l360 £XC%=((l00+A%)*l00+B%)*l00+C%
 l370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 l380 ENDPROC
 2000  DEF PROCeval
 2010 d=(£XH-£XL)/IM%:e=(£YH-£YL)/JM%
 2020 FOR J%=0 TO JM%:Y=£YL+e*J%
 2030 FOR I%=0 TO IM%:X=£XL+d*I%
 2040 £W(I%,J%)=EVAL(F$)
 2050 NEXT:NEXT
 2060 ENDPROC
 2070 REN
```

```
 2100 DEF PROCfPLOTCON(K%,I,J)
 2110 PROCfPL3(K%,fXL+I*d,fYL+J*e,z)
 2120 ENDPROC
 2200 DEF PROCplot:LOCAL DZ
 2210 DZ=(H2-H1)/N%
 2220 FOR z=H1 TO H2+DZ/2 STEP DZ
 2230 PROCfCNTR(Z,IM%,JM%):
 2240 NEXT
 2250 ENDPROC
10060 DEFPROCfINIT(M%): fM%=M%
10070 fXL=-10:fXH=10:fYL=-10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1:fYN%=1:fSN%=1
10100 IF M%=2 O RM%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=10102O3 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10200 DEFFNfI(N%,LO,HI):LOCALA,B,C
10220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10250 A=10^INT(LOG(C)):B=C/A
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10310 DEF PROCfax(O,LO,HI,I)
10350 IFLO-I/2>=O ORO>=HI THENO=INT(LO/I-.1)*I:LO=O
10370 LO=O+I*INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10400 f0=O:f1=LO:f2=HI:ENDPROC
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%:fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIVfXN%*fSYS%:ENDPROC
10490 DEFFNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)-(M%DIV1000))/
f0
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2) MOD 3) GOTO 10540,10550,10560
10540 =0.
10550 =-f0
10560 =-f0/2
10590 DEFPROCfco(C%,N%):IFFNfpr(C%,6)>0THENGCOL0,FNfpr(C%,N%)
10600 ENDPROC
10630 DEFFNfsu(lo,hi,I,f%):LOCALi%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10660 IFj%>0THENc%=j%+1:ELSEc%=1
10670 IFNOTi%<0THENd%=0:p%=0:ELSEd%=-i%:p%=1
10690 IFlo<0ORhi<0THENs%=1ELSEs%=0
10710 IFNOT(f%<p%+s%+c%+d%)THENf0=0ELSEd%=0:p%=0:f0=FNmin(-i%,f%-j%-
s%-1)
10730 IFd%=0THENi%=&10000+f%ELSEi%=&10200+d%
10740 =f%+&100*i%
10760 DEFFNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEFFNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEFFNfpr(n%,d%)=(n%DIV(10^d%))MOD100
10790 DEFPROCfer(a$):PRINT"Graphics package error"
```

```
l0800 PRINTa$:STOP:ENDPROC
l2010 DEF PROC£MOVE(X,Y):PLOT 4,FN£x(X),FN£y(Y):ENDPROC
l2020 DEF PROC£DRAW(X,Y):PLOT 5,FN£x(X),FN£y(Y):ENDPROC
l2030 DEF PROC£PLOT(K%,X,Y) :LOCAL x,y
l2040 IF K%MOD8<4 THEN x=£XFA*X:y=£YFA*Y :£XSC%=£XSC%+x:£YSC%=£YSC%+
y:ELSE x=FN£x(X):y=FN£y(Y)
l2050 PLOT K%,x,y:ENDPROC
l2060 DEF FN£POINT(X,Y)=POINT FN£x(X),FN£y(Y)
l2070 DEF FN£x(X):£XSC%=£XCR%+£XFA*(X-£SXO):=£XSC%
l2080 DEF FN£y(Y):£YSC%=£YCR%+£YFA*(Y-£SYO):=£YSC%
l2100 DEF PROC£AXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
l2110 IF M%=1 THEN GOSUB l2140 ELSE IF M%=2 THEN GOSUB l2300 ELSE IF
M%=3 THEN GOSUB l2400 ELSE GOSUB l2140:GOSUB l2300:GOSUB l2400
l2120 ENDPROC
l2140 £SXI=FN£I(£XI%, £XL,£XH):£SYI=FN£I (£YI%,£YL,£YH)
l2150 PROC£ax(£XO,£XL,£XH,£SXI):£SXO=£0:£SXL=£1:£SXH=£2
l2160 PROC£ax(£YO,£YL,£YH,£SYI):£SYO=£0:£SYL=£1:£SYH=£2
l2180 PROC£se
l2200 £XFA=FN£fa(£SXS%,£HM%,£SXH,£SXL):A=£0*£XFA
l2210 £YFA=FN£fa(£SYS%,£VM%,£SYH,£SYL):B=£0*£YFA
l2230 £0=ABS((£XFA-£YFA)/(£XFA+£YFA))
l2240 IF ABS((£XFA-£YFA)/(£XFA+£YFA))<.15 THEN £XFA=
FNmin(£XFA,£YFA):£YFA=£XFA
l2260 £XCR%=£SXB%+(£HM% DIV 1000)+£XFA*(£SXO-£SXL)+(A-£XFA*(£SXH-
£SXL))/2
l2270 £YCR%=£SYB%+(£VM%DIV1000)+£YFA*(£SYO-£SYL)+(B-£YFA*(£SYH-
£SYL))/2
l2280 RETURN
l2300 PROC£co(£XC%,4):PROC£MOVE(£SXL,£SYO) :PROC£DRAW(£SXH,£SYO)
l2310 PROC£co(£YC%,4):PROC£MOVE(£SXO,£SYL) :PROC£DRAW(£SXO,£SYH)
l2320 B=FN£pp(£SYI,£XP%) :PROC£co(£XC%,2)
l2330 FOR A=£SXL TO £SXH+£SXI/2 STEP £SXI
l2340 PROC£MOVE(A,£SYO+B):PROC£DRAW(A,£SYO+B+£0):NEXT
l2350 A=FN£pp(£SXI,£YP%):PROC£co(£YC%,2)
l2360 FOR B=£SYL TO £SYH+£SYI/2 STEP £SYI
l2370 PROC£MOVE(£SXO+A,B):PROC£PLOT(1,£0,0):NEXT
l2380 RETURN
l2400 f%=FN£pr(£XP%,4):w%=£CH%DIV1E3
l2420 a%=@%:@%=FN£su(£SXL,£SXH,£SXI,f%)
l2440 IF£XFA*£SXI>(1+f%)*w%ANDNOT(£M%=2OR£M%=5 )THENl%=TRUEELSEl%=£
l2460 VDU 5:PROC£co(£XC%,0):A=£SXH
l2470 IF£0<>0THENL$="E"+STR$(-£0)ELSEL$=""
l2480 d%=£SXB%+£SXS%:y%=FN£y(£SYO)+(£XP%DIV1E6)-500
l2490 REPEATIFABS(A)<£SXI/2THENA=0
l2500 L$=STR$(A*10^£0)+L$:B=LEN(L$)*w%
l2510 d%=FNmin(FN£x(A)-B/2,d%-B-w%DIV2)
l2520 PLOT4,d%,y%:PRINTL$:L$=""
l2530 IFl%THENA=A-£SXI ELSEA=£SXL:l%=TRUE
l2540 UNTILA<£SXL-£SXI/2
l2560 f%=FN£pr(£YP%,4):y%=£CH%MOD1E3:d%=y%+£YP%DIV1E6-500
l2570 @%=FN£su(£SYL,£SYH,£SYI,f%)
l2580 IF£YFA*£SYI>2*y%ANDNOT(£M%=2OR£M%=5)THENl%=1ELSEl%=0
l2590 PROC£co(£YC%,0 ):B=£SYL
l2600 REPEAT PROC£MOVE(£SXO,B)
l2610 IF ABS(B)<£SYI/2 THEN B=0
l2630 PLOT0,-w%DIV2, d%:FORA=1TOf%:VDU8: NEXT:PRINTB*10^£0;
l2640 IFl%=1THENB=B+£SYI ELSE B=£SYH:l%=1
l2650 UNTILB>£SYH+£SYI/2
l2660 IF £0<>0 THEN PRINT"E";-£0
l2670 VDU 4:@%=a%:RETURN
```

```
14000 REM
14020 DEFPROCfCNTR(Z,NI%,NJ%):LOCALK%,X,Y,F%,I%,J%,Z0,Z1
14070 FORJ%=0TONJ%-1:FORI%=0TONI%-1:K%=1
14080 Z0=fW(I%,J%):Z1=fW(I%+1,J%):GOSUB14180
14090 IFF%=1THENX=I%+D:Y=J%:GOSUB14210
14100 Z0=Z1:Z1=fW(I%+1,J%+1):GOSUB14180
14110 IFF%=1THENX=I%+1:Y=J%+D:GOSUB14210
14120 Z0=Z1:Z1=fW(I%,J%+1):GOSUB14180
14130 IFF%=1THENX=I%+1-D:Y=J%+1:GOSUB14210
14140 Z0=Z1:Z1=fW(I%,J%):GOSUB14180
14150 IFF%=1THENX=I%:Y=J%+1-D:GOSUB14210
14160 NEXT:NEXT:ENDPROC
14180 IFZ0=Z THEND=0:K%=1-K%:F%=1:RETURN
14190 IF(Z0-Z)*(Z1-Z)<0THEND=(Z-Z0)/(Z1-Z0):K%=1-K%:F%=1:RETURN
ELSEF%=0:RETURN
14210 PROCfPLOTCON(K%+4,X,Y):RETURN
```

```
    0 REM Graphs and Charts Pack
    1 REM Copyright (C) Acornsoft 1982
   10 :REM L3-CO3D
   20 fM%=5:IM%=10:JM%=10
   30 F$="0":DIM fW(IM%,JM%)
   40 ON ERROR GOTO 700
   90 MODE4:VDU19 128,4,0,0,0
  100 K%=0:REPEAT PROCh(1)
  110 fM%=FNp(K%,4,"Mode for graph",fM%)
  112 IF fM%<4 THEN PRINTTAB(2,5)"No room for MODE";fM%:GOTO110
  120 PROCfINIT(fM%)
  130 PROCc(6)
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
  170 PRINTTAB(2,4)"X-axis:"
  180 fXL=FNp(K%,5,"Low end",fXL)
  190 fXH=FNp(K%,6,"High end",fXH)
  200 PRINTTAB(2,8)"Y-axis:"
  210 fYL=FNp(K%,9,"Low end ",fYL)
  220 fYH=FNp(K%,10,"High end",fYH)
  230 PRINTTAB(2,12)"Z-axis:"
  235 fZL=FNp(K%,13,"Low end",fZL)
  240 fZH=FNp(K%,14,"High end",fZH)
  260 PRINTTAB(2,16)"Viewing angles in degrees:"
  270 PROCth(17):PROCph(18)
  280 K%=1:UNTIL FNe
  290 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
  320 PRINTTAB(2,6)"After axes drawn,use keys:"
  325 PRINTTAB(2,8)"A:change view angles"
  330 PRINT"  M: move to X,Y,Z"
  332 PRINT"  D: draw to X,Y,Z"
  340 PRINT"  C: colour "
  342 PRINT"  L: change logical colours"
  350 PRINT"  F: input function F(X,Y)"
  360 PRINT"  P: plot contours"
  375 PRINT"  T: title"
  376 PRINT"  U: print F(X,Y) at X,Y,Z"
```

```
  379 PRINT"  ?: list of prompts"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=£CH%MODlE3:£YB%=3*X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AX3(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MODlE3))TAB(0,29);
  560 A$=GET$
  570 IF A$="M"THENINPUT"Move to X,Y,Z ",X,Y,Z:PROC£MV3(X,Y,Z)
  572 IF A$="D"THENINPUT"Draw to X,Y,Z ",X,Y,Z:PROC£DR3(X,Y,Z)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical, actual col",A%,B%:VDU19
A%,B%,0,0,0
  590 IFA$="F"THENINPUT "Function F(X,Y)",F$:PROCw
  600 IF A$="P "THENINPUT"Z1, Z2, N ",HI,B2,N%:PROCplot
  620 IF A$="T"THENINPUT"Ti
tle,X,Y,Z",A$,X,Y,Z:PROC£MV3(X,Y,Z):VDU5:PRINTA$:VDU4
  630 IF A$="U"THEN PROCfn
  640 IF A$="A"THENINPUT"Theta,Phi ",X,Y:£TH=PI*X/180:£PH=PI*Y/
180:CLG:PROC£AX3(0):PROC£co(£XC%,2)
  670 IF A$="?"THENPRINT"A,M,D,C,L,P,T,F,U":A$=INKEY$(500)
  680 GOTO550
  690 REM error branch
  700  MODE7:IF ERR=17 THEN900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat?(S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R" THEN 90
  940 GOTO910
  950 REM
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"3-D CONTOURS";SPC(5); "PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =FALSE
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCc(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FN£pr(£XC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FN£pr(£XC%,4):B%=FN£pr(£XC%,2)
 1320 C%=FN£pr(£XC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
 1350 C%=FNp(K%,L%+4,"labels",C%)
 1360 £XC%=((100+A%)*100+B%)*100+C%
```

```
 1370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 1380 ENDPROC
 1400 DEF PROCfn
 1410 INPUT"F(X,Y)at X,Y,Z",X,Y,Z
 1420 PROC£MV3(X,Y,Z):VDU5
 1430 PRINT"F(X,Y)=";F$
 1440 VDU 4:ENDPROC
 1500 DEF PROCth(l%)
 1510 £TH=PI*FNp(K%,l%,"Theta",180*£TH/PI)/180:ENDPROC
 1520 DEF PROCph(l%)
 1530 £PH=PI*FNp(K%,18,"Phi ",180*£PH/PI)/180:ENDPROC
 2000 DEF PROCplot:LOCAL DZ
 2010 DZ=(H2-H1)/N%
 2020 FOR z=H1 TO H2+DZ/2 STEP DZ
 2030 PROC£CNTR(z,IM%,JM%):NEXT:ENDPROC
 2040 DEF PROC£PLOTCON(K%,I,J)
 2050 PROC£PL3(K%,£XL+I*d,£YL+J*e,z)
 2060 ENDPROC
 2070 DEF PROCw
 2080 d=(£XH-£XL)/IM%:e=(£YH-£YL)/JM%
 2090 FOR I%=0 TO IM%:X=£XL+I%*d
 2100 FOR J%=0 TO JM%:Y=£YL+J%*e
 2110 £W(I%,J%)=EVAL(F$):NEXT:NEXT
 2120 ENDPROC
10060 DEFPROC£INIT(M%):£M%=M%
10070 £XL=-10:£XH=10:£YL=-10:£YH=10:£ZL=-10:£ZH=10
10080 £XO=0:£YO=0:£ZO=0:£XI%=7:£YI%=7:£ZI%=7
10090 £XN%=1:£YN%=1:£SN%=1
10100 IFM%=2ORM%=5THEN£CH%=64ELSE£CH%=32
10110 £YB%=0:£YS%=1023:£XB%=3*£CH%:£XS%=1279-£XB%
10120 £VM%=64064:£HM%=2*(1000*£CH%+£CH%):£CH%=1000*£CH%+32
10130 £XP%=484040225:£YP%=504040225:£ZP%=540040225
10140 IFM%=1ORM%=2ORM%=5THEN£XC%=10102O3ELSE£XC%=0
10150 £YC%=0:£ZC%=0
10160 £TH=1.3:£PH=-1.3
10170 ENDPROC
10200 DEFFN£I(N%,LO,HI):LOCALA,B,C
10210 IF0>=N%THENPROC£er("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROC£er("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10250 A=10^INT(LOG(C)):B=C/A
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10310 DEF PROC£ax(O,LO,HI,I)
10350 IFLO-I/2>=O ORO>=HI THENO=INT(LO/I-.1)*I:LO=O
10370 LO=O+I*INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10400 £0=O:£1=LO:£2=HI:ENDPROC
10430 DEF PROC£se:LOCAL S%
10440 S%=(£SN%-1)MOD(£XN%*£YN%)
10450 £SXS%=£XS%DIV£XN%:£SYS%=£YS%DIV£YN%
10460 £SXB%=£XB%+S%MOD£XN%*£SXS%
10470 £SYB%=£YB%+S%DIV£XN%*£SYS%:ENDPROC
10490 DEFFN£fa(S%,M%,HI,LO):£0=HI-LO:=(S%-(M%MOD1000)(M%DIV1000))/£0
10510 DEF FN£pp(L,M%)
10520 £0=FN£pr(M%,0)*L/100
10530 ON1+(FN£pr(M%,2)MOD3)GOTO10540,10550,10560
10540 =0.
10550 =-£0
10560 =-£0/2
```

```
10590 DEFPROCfco(C%,N%):IFFNfpr(C%,6)>0THENGCOL0,FNfpr(C%,N%)
10600 ENDPROC
10630 DEFFNfsu(lo,hi,I,f%):LOCALi%,j%,c%,d%,s%,p%
10640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
10660 IFj%>0THENc%=j%+1:ELSEc%=1
10670 IFNOTi%<0THENd%=0:p%=0:ELSEd%=-i%:p%=1
10690 IFlo<0ORhi<0THENs%=1ELSEs%=0
10710 IFNOT(f%<p%+s%+c%+d%)THENf0=0ELSEd%=0:p%=6:f0=FNmin(-i%,f%-j%-
s%-1)
10730 IFd%=0THENi%=&10000+f%ELSEi%=&10200+d%
10740 =f%+&100*i%
10760 DEFFNmax(a,b):IFb>a THEN=b ELSE=a
10770 DEFFNmin(a,b):IFb<a THEN=b ELSE=a
10780 DEFFNfpr(n%,d%)=(n%DIV(10^d%))MOD100
10790 DEFPROCfer(a$):PRINT"Graphics package error"
10800 PRINTa$:STOP:ENDPROC
13010 DEFPROCfMV3(X,Y,Z):PROCfPL3(4,X,Y,Z):ENDPROC
13020 DEFPROCfDR3(X,Y,Z):PROCfPL3(5,X,Y,Z):ENDPROC
13030 DEFPROCfPL3(K%,X,Y,Z):LOCAL RX,RY,RZ,x%,y%
13040 IF3<K%MOD8THENRX=fSXO:RY=fSYO:fRZ=fSZO:x%=fXCR%:y%=
fYCR%:fXSC%=0:fYSC%=0
13050 RX=(X-RX)*fXFA:RY=(Y-RY)*fYFA:RZ=(Z-RZ)*fZFA
13060 x%=x%+fP11*RX+fP12*RY
13070 y%=y%+fP21*RX+fP22*RY+fP23*RZ
13080 fXSC%=fXSC%+x%:fYSC%=fYSC%+y%
13090 PLOTK%,x%,y%:ENDPROC
13100 DEFFNfPT3(X,Y,Z):PROCfPL3(4,X,Y,Z):=POINT(fXSC%,fYSC%)
13130 DEFPROCfAX3(M%):LOCALP1,P2,P3,P4,P5,P6,LM%,DM%,a%
13140 IFM%=1THENGOSUB13170ELSEIFM%=2THENGOSUB13560ELSEIFM%=
3THENGOSUB13700ELSEGOSUB13170:GOSUB13560:GOSUB13700
13150 ENDPROC
13170 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH):fSZI=
FNfI(fZI%,fZL,fZH)
13180 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
13190 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
13200 PROCfax(fZO,fZL,fZH,fSZI):fSZO=f0:fSZL=f1:fSZH=f2
13220 PROCfse
13240 P4=fSXH-fSXL:P5=fSYH-fSYL:P6=fSZH-fSZL
13260 LM%=fHM%DIV1E3:DM%=fVM%DIV1E3
13270 f1=fSXS%-LM%-fHM%MOD1E3:P1=f1/P4:P2=f1/P5
13280 f2=fSYS%-DM%-fVM%MOD1E3:P3=f2/P6
13300 IFABS((P1-P2)/(P1+P2))<.4THENP1=FNmin(P1,P2):P2=P1
13310 IFABS((P1-P3)/(P1+P3))>.2THEN13340
13320 IFP1<P3 THENP3=P1 ELSEf0=P1:P1=P3:IFf0=P2 THENP2=P3
13340 f0=COS(fTH):fP11=-SIN(fPH):fP22=f0*fP11
13350 fP12=COS(fPH):fP21=-f0*fP12:fP23=SIN(fTH)
13370 P6=P1*P4*ABS(fP21)+P2*P5*ABS(fP22)+P3*P6*ABS(fP23)
13380 P5=P1*P4*ABS(fP11)+P2*P5*ABS(fP12)
13400 f0=FNmin(f1/P5,f2/P6)
13410 fXFA=f0*P1:fYFA=f0 *P2:fZFA=f0*P3:P5=f0*P5:P6=f0*P6
13430 IFfP11<0THENP1=fSXH ELSEP1=fSXL
13440 IFfP12<0THENP2=fSYH ELSEP2=fSYL
13450 P3=(P2-fSYO)*fYFA*fP12+(P1-fSXO)*fXFA*fP11
13470 IFfP21<0THENP1=fSXH ELSEP1=fSXL
13480 IFfP22<0THENP2=fSYH ELSEP2=fSYL
13490 IFfP23<0THENP4=fSZH ELSEP4=fSZL
13500 P4=(P1-fSXO)*fXFA*fP21+(P2-fSYO)*fYFA*fP22+(P4-fSZO)*fZFA*fP23
13520 fXCR%=fSXB%+LM%+(f1-P5)/2-P3
13530 fYCR%=fSYB%+DM%+(f2-P6)/2-P4
13540 RETURN
```

```
l3560
PROCfco(fXC%,4):PROCfMV3(fSXL,fSYO,fSZO):PROCfDR3(fSXH,fSYO,fSZO)
l3570
PROCfco(fYC%,4):PROCfMV3(fSXO,fSYL,fSZO):PROCfDR3(fSXO,fSYH,fSZO)
l3580
PROCfco(fZC%,4):PROCfMV3(fSXO,fSYO,fSZL):PROCfDR3(fSXO,fSYO,fSZH)
l3590 B=FNfpp(fSZI,fXP%):PROCfco(fXC%,2)
l3600 FORA=fSXL TOfSXH+fSXI/2STEPfSXI
l3610 PROCfMV3(A,fSYO,fSZO+B):PROCfDR3(A,fSYO,fSZO+B+f0):NEXT
l3620 B=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
l3630 FORA=fSYL TOfSYH+fSYI/2STEPfSYI
l3640 PROCfMV3(fSXO+B,A,fSZO):PROCfDR3(fSXO+B+f0,A,fSZO):NEXT
l3650 B=FNfpp(fSXI,fZP%):PROCfco(fZC%,2)
l3660 FORA=fSZL TOfSZH+fSZI/2STEPfSZI
l3670 PROCfMV3(fSXO+B,fSYO,A):PROCfDR3(fSXO+B+f0,fSYO,A):NEXT
l3680 RETURN
l3700 Pl=FNfpr(fXP%,4):P2=(fXP%DIVlE6)-500:P3=fCH%DIVlE3
l3720 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,Pl)
l3740 VDU5:PROCfco(fXC%,0):PROCfMV3(fSXH,fSYO,fSZO)
l3750 PLOT0,0,P2:FORP4=lTOPl DIV2:VDU8:NEXT:PRINTfSXH*l0^f0:IFf0<>
0THENPRINT"E";-f0
l3760 Pl=FNfpr(fYP%,4):P2=(fYP%DIVlE6)-500
l3770 @%=FNfsu(fSYL,fSYH,fSYI,Pl)
l3780 PROCfco(fYC%,0):PROCfMV3(fSXO,fSYH,fSZO)
l3790 PLOT0,0,P2:PRINTfSYH*l0^f0:IFf0<>0THENPRINT"E";-f0
l3800 Pl=FNfpr(fZP%,4):P2=(fZP%DIVlE6)-500
l3810 @%=FNfsu(fSZL,fSZH,fSZI,Pl)
l3820 PROCfco(fZC%,0):PROCfMV3(fSXO,fSYO,fSZH)
l3830 PLOT0,0,P2:FORP4=lTOPl DIV2:VDU8:NEXT:PRINTfSZH*l0^f0:IFf0<>
0THENPRINT"E";-f0
l3840 VDU4:@%=a%:RETURN
l4020 DEFPROCfCNTR(Z,NI%,NJ%):LOCALK%,X,Y,F%,I%,J%,Z0,Zl
l4070 FORJ%=0TONJ%-l:FORI%=0TONI%-l:K%=l
l4080 Z0=fW(I%,J%):Zl=fW(I%+l,J%):GOSUBl4l80
l4090 IFF%=lTHENX=I%+D:Y=J%:GOSUBl4210
l4100 Z0=Zl:Zl=fW(I%+l,J%+l):GOSUBl4l80
l4110 IFF%=lTHENX=I%+l:Y=J%+D:GOSUBl4210
l4120 Z0=Zl:Zl=fW(I%,J%+l):GOSUBl4l80
l4130 IFF%=lTHENX=I%+l-D:Y=J%+l:GOSUBl4210
l4140 Z0=Zl:Zl=fW(I%,J%):GOSUBl4l80
l4150 IFF%=lTHENX=I%:Y=J%+l-D:GOSUBl4210
l4160 NEXT:NEXT:ENDPROC
l4180 IFZ0=Z THEND=0:K%=l-K%:F%=l:RETURN
l4190 IF(ZO-Z)*(Zl-Z)<OTHEND=(Z-Z0)/(Zl-Z0):K%=l-K%:F%=l:RETURN
ELSEF%=0:RETURN
l4210 PROCfPLOTCON(K%+4,X,Y):RETURN
```

```
    0 REM Graphs and Charts Pack
    l REM Copyright(C)Acornsoft l982
   l0 :REM L3-SURF
   20 f M%=5:F$="0"
   40 ON ERROR GOTO 700
   90 MODE4:VDUl9 l28,4,0,0,0
  l00 K%=0:REPEAT PROCh(l)
  ll0 f M%=FNp(K%,4,"Mode for graph",f M%)
  l20 PROCfINIT(f M%)
  l30 PROCc(6)
```

```
  140 K%=1:UNTIL FNe
  150 REM
  160 CLS:K%=0:REPEAT PROCh(2)
  170 PRINTTAB(2,4)"X-axis:"
  180 £XL=FNp(K%,5,"Low end",£XL)
  190 £XH=FNp(K%,6,"High end",£XH)
  200 PRINTTAB(2,8)"Y-axis:"
  210 £YL=FNp(K%,9,"Low end",£YL)
  220 £YH=FNp(K%,10,"High end",£YH)
  230 PRINTTAB(2,12)"Z-axis:"
  235 £ZL=FNp(K%,13,"Low end",£ZL)
  240 £ZH=FNp(K%,14,"High end",£ZH)
  260 PRINTTAB(2,16)"Viewing angles in degees:"
  270 PROCth(17):PROCph(18)
  280 K%=1:UNTIL FNe
  290 REM
  300 PROCh(3)
  310 PRINT TAB(2,4)"READY TO PLOT GRAPH"
  320 PRINTTAB(2,6)"After axes drawn,use keys:"
  325 PRINTTAB(2,8)"A:change view angles"
  330 PRINT"  M: move to X,Y,Z"
  332 PRINT"  D: draw to X,Y,Z"
  340 PRINT"  C: colour"
  342 PRINT"  L: change logical colours"
  350 PRINT"  F: input function F(X,Y)"
  360 PRINT"  P: plot"
  375 PRINT"  T: title"
  376 PRINT"  U: print F(X,Y)at X,Y,Z"
  379 PRINT"  ?: list of prompts"
  380 PRINTTAB(2,29)"press";:GOSUB 1270
  390 A$=GET$:IF NOT(A$=" ")THEN380
  400 REM
  500 X=£CH%MOD1E3:£YB%=3*X:£YS%=1023-4*X
  510 REM
  520 MODE £M%
  530 PROC£AX3(0):PROC£co(£XC%,2)
  550 PRINTTAB(0,29)SPC(1280DIV(£CH%MOD1E3))TAB(0,29);
  560 A$=GET$
  570 IF A$="M"THENINPUT"Move to X,Y,Z ",X,Y,Z:PROC£MV3(X,Y,Z)
  572 IF A$="D "THENINPUT"Draw to X,Y,Z ",X,Y,Z:PROC£DR3(X,Y,Z)
  580 IF A$="C"THENINPUT"Colour ",X:GCOL0,X
  582 IF A$="L"THENINPUT"logical,actual col",A%,B%:VDU19 A%,B%,0,0,0
  590 IFA$="F "THENINPUT"Function F(X,Y)",F$
  600 IF A$="P"THENINPUT"No. intervals NX,NY "NX%,NY%:PROCplot
  620 IF A$="T"THENINPUT"Ti tle,X,Y,Z",A$,X,Y,Z:PROC£MV3(X,Y,Z):VDU
5:PRINTA$:VDU4
  630 IF A$="U"THEN PROCfn
  640 IF A$="A"THENINPUT"Theta,Phi ",X,Y:£TH=PI*X/180:£PH=PI*Y/
180:CLG:PROC£AX3(0):PROC£co(£XC%,2)
  670 IF A$="?"THENPRINT"A,M,D,C,L,P,T,F,U":A$=INKEY$(500)
  680 GOTO 550
  690 REM error branch
  700 MODE 7:IF ERR=17 THEN 900
  710 REPORT
  900 PRINTTAB(0,10)"Stop or repeat?(S/R)";
  910 A$=GET$
  920 IF A$="S" THENPRINT:END
  930 IF A$="R" THEN 90
  940 GOTO910
  950 REM
```

```
 1000 DEF FNp(M%,L%,P$,V):LOCAL P%,V$
 1010 PRINT TAB(2,L%);P$;" =";V;
 1020 IF M%>0 THEN PRINT" ? ";:P%=POS
 1030 PRINT SPC(39-P%);:IF M%=0 THEN =V
 1040 INPUT TAB(P%,L%)V$
 1050 IF LEN(V$)>0 THEN V=VAL(V$)ELSE M%=0
 1060 GOTO1010
 1100 DEF PROCh(N%):CLS
 1120 PRINT TAB(2,2);"3-D SURFACE VIEWER";SPC(5);"PAGE ";N%
 1130 ENDPROC
 1200 DEF FNe:LOCAL A$
 1210 PRINT TAB(0,29)"Type C to change data,";:GOSUB 1270
 1220 A$=GET$
 1230 IF A$=" " THEN =TRUE
 1240 IF A$="C" THEN =FALSE
 1250 GOTO 1220
 1260 REM
 1270 PRINT" SPACE to continue":RETURN
 1300 DEF PROCc(L%)
 1302 PRINTTAB(2,L%)"Logical colours for:"
 1304 IF FN£pr(£XC%,6)=0 THEN A%=1:B%=1:C%=1:GOTO1330
 1310 A%=FN£pr(£XC%,4):B%=FN£pr(£XC%,2)
 1320 C%=FN£pr(£XC%,0)
 1330 A%=FNp(K%,L%+2,"axes  ",A%)
 1340 B%=FNp(K%,L%+3,"pips  ",B%)
 1350 C%=FNp(K%,L%+4,"labels",C%)
 1360 £XC%=((100+A%)*100+B%)*100+C%
 1370 PRINTTAB(2,L%+6)"Actual colours may be changed later."
 1380 ENDPROC
 1400 DEF PROCfn
 1410 INPUT"F(X,Y) at X,Y,Z",X,Y,Z
 1420 PROC£MV3(X,Y,Z):VDU5
 1430 PRINT"F(X,Y)=";F$
 1440 VDU4:ENDPROC
 1500 DEF PROCth(l%)
 1510 £TH=PI*FNp(K%,l%,"Theta",180*£TH/PI)/180:ENDPROC
 1520 DEF PROCph(l%)
 1530 £PH=PI*FNp(K%,18,"Phi ",180*£PH/PI)/180:ENDPROC
 2000 DEF PROCplot:LOCAL D,E
 2010 D=(£XH-£XL)/NX%:E=(£YH-£YL)/NY%
 2020 FOR X=£XL TO £XH+D/2 STEP D
 2030 Y=£YL:PROC£MV3(X,Y,EVAL(F$))
 2040 FOR Y=£YL+E TO £YH+E/2 STEP E
 2050 PROC£DR3(X,Y,EVAL(F$))
 2060 NEXT:NEXT
 2070 FOR Y=£YL TO £YH+E/2 STEP E
 2080 X=£XL:PROC£MV3(X,Y,EVAL(F$))
 2090 FOR X=£XL+D TO £XH+D/2 STEP D
 2100 PROC£DR3(X,Y,EVAL(F$))
 2110 NEXT:NEXT
 2120 ENDPROC
10060 DEFPROC£INIT(M%):£M%=M%
10070 £XL=-10:£XH=10:£YL=-10:£YH=10:£ZL=-10:£ZH=10
10080 £XO=0:£YO=0:£ZO=0:£XI%=7:£YI%=7:£ZI%=7
10090 £XN%=1:£YN%=1:£SN%=1
10100 IFM%=2ORM%=5THEN £CH%=64ELSE£CH%=32
10110 £YB%=0:£YS%=1023:£XB%=3*£CH%:£XS%=1279-£XB%
10120 £VM%=64064:£HM%=2*(1000*£CH%+£CH%):£CH%=1000*£CH%+32
10130 £XP%=484040225:£YP%=504040225:£ZP%=540040225
10140 IFM%=1ORM%=2ORM%=5THEN£XC%=1010203ELSE£XC%=0
```

```
l0150 £YC%=0:£ZC%=0
l0160 £TH=l.3:£PH=-l.3
l0170 ENDPROC
l0200 DEFFN£I(N%,LO,HI):LOCALA,B,C
l0210 IF0>=N%THENPROC£er("Bad no. of intervals")
l0220 C=(HI-LO)/N%:IF0>=C THENPROC£er("Bad axis range")
l0230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/lE6)
l0250 A=10^INT(LOG(C)):B=C/A
l0270 IFB>7.lTHENB=l0ELSEIFB>3.2THENB=5ELSEIFB>l.4THENB=2ELSEB=l
l0280 =A*B
l0310 DEF PROC£ax(O,LO,HI,I)
l0350 IFLO-I/2>=O ORO>=HI THENO=INT(LO/I-.l)*I:LO=O
l0370 LO=O+I*INT((LO-O)/I+.l)
l0380 HI=O-I*INT((O-HI)/I+.l)
l0400 £0=O:£l=LO:£2=HI:ENDPROC
l0430 DEF PROC£se:LOCAL S%
l0440 S%=(£SN%-l)MOD(£XN%*£YN%)
l0450 £SXS%=£XS%DIV£XN%:£SYS%=£YS%DIV£YN%
l0460 £SXB%=£XB%+S%MOD£XN%*£SXS%
l0470 £SYB%=£YB%+S%DIV£XN%*£SYS%:ENDPROC
l0490 DEFFN£fa(S%,M%,HI,LO):£0=HI-LO:=(S%-(M%MODl000)-(M%DIVl000))/
£0
l0510 DEF FN£pp(L,M%)
l0520 £0=FN£pr(M%,0)*L/l00
l0530 ONl+(FN£pr(M%,2)MOD3)GOTOl0540,l0550,l0560
l0540 =0.
l0550 =-£0
l0560 =-£0/2
l0590 DEFPROC£co(C%,N%):IFFN£pr(C%,6)>0THENGCOL0,FN£pr(C%,N%)
l0600 ENDPROC
l0630 DEFFN£su(lo,hi,I,f%):LOCALi%,j%,c%,d%,s%,p%
l0640 i%=INT(LOG(I)+.0l):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.0l)
l0660 IFj%>0THENc%=j%+l:ELSEc%=l
l0670 IFNOTi%<0THENd%=0:p%=0:ELSEd%=-i%:p%=l
l0690 IFlo<0ORhi<0THENs%=lELSEs%=0
l0710 IFNOT(f%<p%+s%+c%+d%)THEN£0=0ELSEd%=0:p%=0:£0=FNmin(-i%,f%-j%-
s%-l)
l0730 IFd%=0THENi%=&l0000+f%ELSEi%=&l0200+d%
l0740 =f%+&l00*i%
l0760 DEFFNmax(a,b):IFb>a THEN=b ELSE=a
l0770 DEFFNmin(a,b):IFb<a THEN=b ELSE=a
l0780 DEFFN£pr(n%,d%)=(n%DIV(l0^d%))MODl00
l0790 DEFPROC£er(a$):PRINT"Graphics package error"
l0800 PRINTa$:STOP:ENDPROC
l3010 DEFPROC£MV3(X,Y,Z):PROC£PL3(4,X,Y,Z):ENDPROC
l3020 DEFPROC£DR3(X,Y,Z):PROC£PL3(5,X,Y,Z):ENDPROC
l3030 DEFPROC£PL3(K%,X,Y,Z):LOCAL RX,RY,RZ,x%,y%
l3040 IF3<K%MOD8THENRX=£SXO:RY=£SYO:£RZ=£SZO:x%=£XCR%:y%=
£YCR%:£XSC%=0:£YSC%=0
l3050 RX=(X-RX)*£XFA:RY=(Y-RY)*£YFA:RZ=(Z-RZ)*£ZFA
l3060 x%=x%+£Pll*RX+£Pl2*RY
l3070 y%=y%+£P2l*RX+£P22*RY+£P23*RZ
l3080 £XSC%=£XSC%+x%:£YSC%=£YSC%+y%
l3090 PLOTK%,x%,y%:ENDPROC
l3100 DEFFN£PT3(X,Y,Z):PROC£PL3(4,X,Y,Z):=POINT(£XSC%,£YSC%)
l3130 DEFPROC£AX3(M%):LOCALPl,P2,P3,P4,P5,P6,LM%,DM%,a%
l3140 IFM%=lTHENGOSUBl3170ELSEIFM%=2THENGOSUBl3560ELSEIFM%=
3THENGOSUBl3700ELSEGOSUBl3170:GOSUBl3560:GOSUBl3700
l3150 ENDPROC
l3170 £SXI=FN£I(£XI%,£XL,£XH):£SYI=FN£I(£YI%,£YL,£YH):£SZI=
```

```
FNfI(fZI%,fZL,fZH)
13180 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
13190 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
13200 PROCfax(fZO,fZL,fZH,fSZI):fSZO=f0:fSZL=f1:fSZH=f2
13220 PROCfse
13240 P4=fSXH-fSXL:P5=fSYH-fSYL:P6=fSZH-fSZL
13260 LM%=fHM%DIV1E3:DM%=fVM%DIV1E3
13270 f1=fSXS%-LM%-fHM%MOD1E3:P1=f1/P4:P2=f1/P5
13280 f2=fSYS%-DM%-fVM%MOD1E3:P3=f2/P6
13300 IFABS((P1-P2)/(P1+P2))<.4THENP1=FNmin(P1,P2):P2=P1
13310 IFABS((P1-P3)/(P1+P3))>.2THEN13340
13320 IFP1<P3 THENP3=P1 ELSEf0=P1:P1=P3:IFf0=P2 THENP2=P3
13340 f0=COS(fTH):fP11=-SIN(fPH):fP22=f0*fP11
13350 fP12=COS(fPH):fP21=-f0*fP12:fP23=SIN(fTH)
13370 P6=P1*P4*ABS(fP21)+P2*P5*ABS(fP22)+P3*P6*ABS(fP23)
13380 P5=P1*P4*ABS(fP11)+P2*P5*ABS(fP12)
13400 f0=FNmin(f1/P5,f2/P6)
13410 fXFA=f0*P1:fYFA=f0*P2:fZFA=f0*P3:P5=f0*P5:P6=f0*P6
13430 IFfP11<0THENP1=fSXH ELSEP1=fSXL
13440 IFfP12<0THENP2=fSYH ELSEP2=fSYL
13450 P3=(P2-fSYO)*fYFA*fP12+(P1-fSXO)*fXFA*fP11
13470 IFfP21<0THENP1=fSXH ELSEP1=fSXL
13480 IFfP22<0THENP2=fSYH ELSEP2=fSYL
13490 IFfP23<0THENP4=fSZH ELSEP4=fSZL
13500 P4=(P1-fSXO)*fXFA*fP21+(P2-fSYO)*fYFA*fP22+(P4-fSZO)*fZFA*fP23
13520 fXCR%=fSXB%+LM%+(f1-P5)/2-P3
13530 fYCR%=fSYB%+DM%+(f2-P6)/2-P4
13540 RETURN
13560
PROCfco(fXC%,4):PROCfMV3(fSXL,fSYO,fSZO):PROCfDR3(fSXH,fSYO,fSZO)
13570
PROCfco(fYC%,4):PROCfMV3(fSXO,fSYL,fSZO):PROCfDR3(fSXO,fSYH,fSZO)
13580
PROCfco(fZC%,4):PROCfMV3(fSXO,fSYO,fSZL):PROCfDR3(fSXO,fSYO,fSZH)
13590 B=FNfpp(fSZI,fXP%):PROCfco(fXC%,2)
13600 FORA=fSXL TOfSXH+fSXI/2STEPfSXI
13610 PROCfMV3(A,fSYO,fSZO+B):PROCfDR3(A,fSYO,fSZO+B+f0):NEXT
13620 B=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
13630 FORA=fSYL TOfSYH+fSYI/2STEPfSYI
13640 PROCfMV3(fSXO+B,A,fSZO):PROCfDR3(fSXO+B+f0,A,fSZO):NEXT
13650 B=FNfpp(fSXI,fZP%):PROCfco(fZC%,2)
13660 FORA=fSZL TOfSZH+fSZI/2STEPfSZI
13670 PROCfMV3(fSXO+B,fSYO,A):PROCfDR3(fSXO+B+f0,fSYO,A):NEXT
13680 RETURN
13700 P1=FNfpr(fXP%,4):P2=(fXP%DIV1E6)-500:P3=fCH%DIV1E3
13720 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,P1)
13740 VDU5:PROCfco(fXC%,0):PROCfMV3(fSXH,fSYO,fSZO)
13750 PLOT0,0,P2:FORP4=1TOP1 DIV2:VDU8:NEXT:PRINTfSXH*10^f0:IFf0<>
0THENPRINT"E";-f0
13760 P1=FNfpr(fYP%,4):P2=(fYP%DIV1E6)-500
13770 @%=FNfsu(fSYL,fSYH,fSYI,P1)
13780 PROCfco(fYC%,0):PROCfMV3(fSXO,fSYH,fSZO)
13790 PLOT0,0,P2:PRINTfSYH*10''f0:IFf0<>0THENPRINT"E";-f0
13800 P1=FNfpr(fZP%,4):P2=(fZP%DIV1E6)-500
13810 @%=FNfsu(fSZL,fSZH,fSZI,P1)
13820 PROCfco(fZC%,0):PROCfMV3(fSXO,fSYO,fSZH)
13830 PLOT0,0,P2:FORP4=1TOP1 DIV2:VDU8:NEXT:PRINTfSZH*10^f0:IFf0<>
0THENPRINT"E";-f0
13840 VDU4:@%=a%:RETURN
```

5.2 Level Two Programs

```
10000 REM Graphs and Charts
10020 REM Copyright (C) Acornsoft 1982
10030 REM L1-2D
10040 REM
10050 REM Initialisation
10060 DEF PROCfINIT(M%):f%=M%
10070 fXL=-10:fXH=10:fYL=-10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1:fYN%=1:fSN%=1
10100 IF M%=2 OR M%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=10102 03 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FNfI(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROCfer("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10240 REM normalise interval size ^10
10250 A=10^INT(LOG(C)):B=C/A
10260 REM select nearest round interval
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10290 REM
10300 REM calc secondary origin,lo,hi
10310 DEF PROCfax(O,LO,HI,I)
10320 REM if axes wld cross out range,
10330 REM set sec pars so cross at lo,
10340 REM and ensure orig at integer*intvl
10350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.1)*I:LO=O
10360 REM anyway make ends at intg*intvl
10370 LO=O+I *INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10390 REM return value in f0,f1,f2
10400 f0=O:f1=LO:f2=HI:ENDPROC
10410 REM
10420 REM set up sector
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%:fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIVfXN%*fSYS%:ENDPROC
10480 REM
10490 DEF FNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)-(M%DIV1000))/
f0
10500 REM
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2)MOD 3)GOTO 10540,10550,10560
10540 =0.
10550 =-f0
```

```
l0560 =-f0/2
l0570 REM
l0580 REM Extract colour
l0590 DEF PROCfco(C%,N%):IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
l0600 ENDPROC
l0610 REM
l0620 REM scale or unscale labels
l0630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
l0640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
l0650 REM digits needed c% before,d% after
l0660 IF j%>0 THEN c%=j%+1:ELSE c%=1
l0670 IF NOT i%<0 THEN d%=0:p%=0:ELSE d%=-i%:p%=1
l0680 REM is sign needed
l0690 IF lo<0 OR hi<0 THEN s%=1 ELSEs%=0
l0700 REM decide scaledy'unscaled
l0710 IF NOT(f%<p%+s%+c%+d%)THEN f0=0 ELSE d%=0:p%=0:f0=FNmin(-
i%,f%-j%-s%-I)
l0720 REM format; gen if integer el fix
l0730  IF d%=0 THEN i%=&l0000+f% ELSE i%=&l0200+d%
l0740 =f%+&l00*i%
l0750 REM
l0760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
l0770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
l0780 DEF FNfpr(n%,d%)=(n%DIV(10^d%))MODl00
l0790 DEF PROCfer(a$):PRINT "Graphics package error"
l0800 PRINT a$:STOP:ENDPROC
l0810 REM
l2000 REM --- TWO DIM ROUTINES ---
l2010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
l2020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X),FNfy(Y):ENDPROC
l2030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
l2040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y:fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
l2050 PLOT K%,x,y:ENDPROC
l2060 DEF FNfPOINT(X,Y)=POINT FNfx(X),FNfy(Y)
l2070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
l2080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
l2090 REM two dim scales & axes
l2100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
l2110 IF M%=1 THEN GOSUB l2140 ELSE IF M%=2 THENGOSUB l2300ELSE IF
M%=3 THEN GOSUB l2400 ELSEGOSUB l2140:GOSUB l2300:GOSUB l2400
l2120 ENDPROC
l2130 REM set intervals & ends
l2140 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH)
l2150 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
l2160 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
l2170 REM set sector
l2180 PROCfse
l2190 REM scale factors,allow margins
l2200 fXFA=FNffa(fSXS%,fHM%,fSXH,fSXL):A=f0*fXFA
l2210 fYFA=FNffa(fSYS%,fVM%,fSYH,fSYL):B=f0*fYFA
l2220 REM equalise if within 30%
l2230 f0=ABS((fXFA-fYFA)/(fXFA+fYFA))
l2240 IF ABS((fXFA-fYFA)/(fXFA+fYFA))<.15 THEN fXFA=
FNmin(fXFA,fYFA):fYFA=fXFA
l2250 REM position the sixes cross
l2260 fXCR%=fSXB%+(fHM% DIV l000)+fXFA*(fSXO-fSXL)+(A-fXFA*(fSXH-
fSXL))/2
l2270 fYCR%=fSYB%+(fVM%DIVl000)+fYFA*(fSYO-fSYL)+(B-fYFA*(fSYH-
fSYL))/2
l2280 RETURN
```

```
12290 REM draw & mark off axes
12300 PROCfco(fXC%,4):PROCfMOVE(fSXL,fSYO):PROCfDRAW(fSXH,fSYO)
12310 PROCfco(fYC%,4):PROCfMOVE(fSXO,fSYL):PROCfDRAW(fSXO,fSYH)
12320 B=FNfpp(fSYI,fXP%):PROCfco(fXC%,2)
12330 FOR A=fSXL TO fSXH+fSXI/2 STEP fSXI
12340 PROCfMOVE(A,fSYO+B):PROCfDRAW(A,fSYO+B+f0):NEXT
12350 A=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
12360 FOR B=fSYL TO fSYH+fSYI/2 STEP fSYI
12370 PROCfMOVE(fSXO+A,B):PROCfPLOT(1,f0,0):NEXT
12380 RETURN
12390 REM label axes
12400 f%=FNfpr(fXP%,4):w%=fCH%DIVlE3
12410 REM decide scaled or unscaled
12420 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,f%)
12430 REM decide if OK to label all pts
12440 IFfXFA*fSXI>(l+f%)*w%ANDNOT(fM%=2ORfM%=5)THENl%=TRUE ELSEl%=
FALSE
12450 REM now label
12460 VDU 5:PROCfco(fXC%,0):A=fSXH
12470 IFf0<>0THENL$="E"+STR$(-f0)ELSEL$=""
12480 d%=fSXB%+fSXS%:y%=FNfy(fSYO)+(fXP%DIVlE6)-500
12490 REPEATIFABS(A)<fSXI/2THENA=0
12500 L$=STR$(A*l0^f0)+L$:B=LEN(L$)*w%
12510 d%=FNmin(FNfx(A)-B/2,d%-B-w%DIV2)
12520 PLOT4,d%,y%:PRINTL$:L$=""
12530 IFl%THENA=A-fSXI ELSEA=fSXL:l%=TRUE
12540 UNTILA<fSXL-fSXI/2
12550 REM now label Y-axis
12560 f%=FNfpr(fYP%,4):y%=fCH%MODlE3:d%=y%+fYP%DIVlE6-500
12570 @%=FNfsu(fSYL,fSYH,fSYI,f%)
12580 IFfYFA*fSYI>2*y%ANDNOT(fM%=2ORfM%=5)THENl%=lELSEl%=0
12590 PROCfco(fYC%,0):B=fSYL
12600 REPEAT PROCfMOVE(fSXO,B)
12610 IF ABS(B)<fSYI/2 THEN B=0
12630 PLOT0,-w%DIV2,d%:FORA=lTOf%:VDU8:NEXT:PRINTB*l0^f0;
12640 IFl%=lTHENB=B+fSYI ELSE B=fSYH:l%=l
12650 UNTILB>fSYH+fSYI/2
12660 IF f0<>0 THEN PRINT"E";-f0
12670 VDU 4:@%=a%:RETURN
15050 :REM L2-HIS
15060 DEF PROCfBAR(K%,X,Y0,Y,W):LOCAL x,y0
15070 REM K%=0 left of bar at X
15080 REM K%=l bar centered on X
15090 REM YO is base,Y is top,W width
15100 REM all in user units.
15110  IF ABS(Y0-Y)*fYFA<5 THEN ENDPROC
15120 IF Y0=fSYO THEN y0=Y0+SGN(Y-Y0)*l0/fYFA ELSE y0=Y0
15130 IF K%=0 THEN x=X ELSE x=X-W/2
15140 PROCfPLOT(84,x,Y):PROCfPLOT(84,x,y0)
15150 PROCfPLOT(85,x+W,Y):PROCfPLOT(85,x+W,y0)
15160 ENDPROC
15170 REM
16000 REM --- Level Two:histogram --
16010 REM data expected in fX(),fY()
16020 DEF PROCfHIS(I%,N%,W)
16030 IF I%>9 THEN GOTO 16140
16040 PROCfINIT(I%)
16050 REM find min/max X/Y
16060 fXL=fX(0):fXH=fXL:fYH=0:fYL=0
16070 FOR I%=0 TO N%
```

```
16080 £XL=FNmin(£XL,£X(I%))
16090 £YL=FNmin(£YL,£Y(I%))
16100 £XH=FNmax(£XH,£X(I%))
16110 £YH=FNmax(£YH,£Y(I%))
16120 NEXT:£XH=£XH+W
16130 PROC£AXES(0):PROC£co(£XC%,2)
16140 FOR I%=0 TO N%
16150 PROC£BAR(0,£X(I%)+W/10,0,£Y(I%),W)
16160 NEXT:ENDPROC
16170 REM ---------------------------

1518 0:REM L2-PIE
15190 DEF PROC£SEC(M%,X,Y,R,A,B):LOCAL a,b,I%
15200 REM tens digit of M%: center
15210 REM 0=screen, 1=user
15220 IF M%DIV10=1 THEN X=FN£x(X):Y=FN£y(Y)
15230 REM ones digit of M%:radius
15240 IF M%MOD10=1 THEN R=R*£YFA
15250 M%=1+50*ABS(B-A):b=(B-A)*2*PI/M%
15260 A=2*PI*A:PLOT 84,X+R*COS(A),Y+R*SIN(A)
15270 FOR I%=1 TO M%:a=A+I%*b:PLOT 84,X,Y
15280 PLOT 81,R*COS(a),R*SIN(a):NEXT
15290 a=A+M%*b/2:MOVE X+R*COS(a)*.7,Y+R*SIN(a)*.7
15300 ENDPROC
15310 REM ---------------------------
16180 REM ---- Level Two: pie chart --
16190 REM data expected in £Y() in %.
16200 DEF PROC£PIE(I%,N%):LOCAL A,B,a%,c%
16210 IF I%>8 THEN PRINT"Bad mode":STOP
16220 a%=@%:@%=&01000202:VDU5:c%=1
16230 PROC£INIT(I%):PROC£AXES(1)
16240 A=0:FOR I%=0 TO N%:B=A+£Y(I%)/100
16250 GCOL0,c%:PROC£SEC(11,0,0,8,A,B)
16260 VDU8:c%=3-c%:GCOL0,c%:PRINT £Y(I%)
16270 A=B:NEXT:@%=a%:VDU4:ENDPROC
16280 REM ---------------------------

1629 0:REM L2-XY
16300 REM data expected in £X,£Y
16310 DEF PROC£XY(I%,N%)
16320 IF I%>9 THEN GOTO 16420
16330 PROC£INIT(I%)
16340 £XL=£X(0):£XH=£XL:£YL=£Y(0):£YH=£YL
16350 FOR I%=1 TO N%
16360 £XL=FNmin(£XL,£X(I%))
16370 £YL=FNmin(£YL,£Y(I%))
16380 £XH=FNmax(£XH,£X(I%))
16390 £YH=FNmax(£YH,£Y(I%))
16400 NEXT
16410 PROC£AXES(0):PROC£co(£XC%,2)
16420 PROC£MOVE(£X(0),£Y(0))
16430 FOR I%=1 TO N%
16440 PROC£DRAW(£X(I%),£Y(I%)):NEXT
16450 ENDPROC
16460 REM ---------------------------

1686 0:REM L2-XYZ
16870 REM data expected in £X,£Y,£Z
16880 DEF PROC£XYZ(I%,N%)
16890 IF I%>9 THEN 17020
```

```
16900 PROCfINIT(I%)
16910 fXL=fX(0):fXH=fXL:fYL=fY(0)
16920 fYH=fYL:fZL=fZ(0):fZH=fZL
16930 FOR I%=1 TO N%
16940 fXL=FNmin(fXL,fX(I%))
16950 fYL=FNmin(fYL,fY(I%))
16960 fXH=FNmax(fXH,fX(I%))
16970 fYH=FNmax(fYH,fY(I%))
16980 fZL=FNmin(fZL,fZ(I%))
16990 fZH=FNmax(fZH,fZ(I%))
17000 NEXT
17010 PROCfAX3(0):PROCfco(fXC%,2)
17020 PROCfMV3(fX(0),fY(0),fZ(0))
17030 FOR I%=1 TO N%
17040 PROCfDR3(fX(I%),fY(I%),fZ(I%)):NEXT
17050 ENDPROC
17060 REM ----------------------------

14000 0 :REM L2-C2
14020 DEF PROCfCNTR(Z,NI%,NJ%):LOCAL D,K%,X,Y,F%,I%,J%,Z0,Z1
14030 REM scans fW(I,J)where 0:I,J:NI,NJ
14040 REM for mesh coords where W takes
14050 REM value Z. Calls PROCfPLOTCON(K,X,Y)
14060 REM where X,Y in mesh coords.
14070 FOR J%=0 TO NJ%-1:FOR I%=0 TO NI%-1:K%=1
14080 Z0=fW(I%,J%):Z1=fW(I%+1,J%):GOSUB 14180
14090 IF F%=1 THEN X=I%+D:Y=J%:GOSUB 14210
14100 Z0=Z1:Z1=fW(I%+1,J%+1):GOSUB 14180
14110 IF F%=1 THEN X=I%+1:Y=J%+D:GOSUB 14210
14120 Z0=Z1:Z1=fW(I%,J%+1):GOSUB 14180
14130 IF F%=1 THEN X=I%+1-D:Y=J%+1:GOSUB 14210
14140 Z0=Z1:Z1=fW(I%,J%):GOSUB 14180
14150 IF F%=1 THEN X=I%:Y=J%+1-D:GOSUB 14210
14160 NEXT:NEXT:ENDPROC
14170 REM test sides for crossing
14180 IF Z0=Z THEN D=0:K%=1-K%:F%=1:RETURN
14190 IF(Z0-Z)*(Z1-Z)<0THEN D=(Z-Z0)/(Z1-Z0):K%=1-K%:F%=1:RETURN
ELSE F%=0:RETURN
14200 REM call to PROC defined by user
14210 PROCfPLOTCON(K%+4,X,Y):RETURN
15050 :REM L2-HIS
15060 DEF PROCfBAR(K%,X,Y0,Y,W):LOCAL x,y0
15070 REM K%=0 left of bar at X
15080 REM K%=1 bar centered on X
15090 REM YO is base,Y is top,W width
15100 REM all in user units.
15110  IF ABS(Y0-Y)*fYFA<5 THEN ENDPROC
15120 IF Y0=fSYO THEN y0=Y0+SGN(Y-Y0)*10/fYFA ELSE y0=Y0
15130 IF K%=0 THEN x=X ELSE x=X-W/2
15140 PROCfPLOT(84,x,Y):PROCfPLOT(84,x,y0)
15150 PROCfPLOT(85,x+W,Y):PROCfPLOT(85,x+W,y0)
15160 ENDPROC
15170 REM
16470 REM --- L2-CN2D 2 dim contour ---
16480 REM expects fW(I,J)
16490 DEF PROCfCN2D(K%,IM%,JM%,N%)
16500 LOCAL I%,J%,Z
16510 IF K%>9 THEN GOTO 16630
16520 PROCfINIT(K%):fXL=0:fYL=0:fXH=IM%:fYH%=JM%
16530 REM find min/max Z
```

```
16540 £ZL=£W(0,0):£ZH=£ZL
16550 FOR I%=0 TO IM%:FOR J%=0 TO JM%
16560 Z=£W(I%,J%):£ZL=FNmin(Z,£ZL)
16570 £ZH=FNmax(Z,£ZH):NEXT:NEXT
16580 REM round the contour heights
16590 £SZI=FN£I(N%,£ZL,£ZH)
16600 PROC£ax(£ZL,£ZL,£ZH,£SZI):£SZL=£1:£SZH=£2
16610 PROC£AXES(0):PROC£co(£XC%,2)
16620 REM plot contours
16630 FOR Z=£SZL TO £SZH+.1*£SZI STEP £SZI
16640 PROC£CNTR(Z,IM%,JM%):NEXT:ENDPROC
16650 DEF PROC£PLOTCON(K%,I,J)
16660 PROC£PLOT(K%,I,J):ENDPROC
16670 REM ----------------------------

14010 REM L2-C3
14020 DEF PROC£CNTR(Z,NI%,NJ%):LOCAL D,K%,X,Y,F%,I%,J%,Z0,Z1
14070 FOR J%=0 TO NJ%-1:FOR I%=0 TO NI%-1:K%=1
14080 Z0=£W(I%,J%):Z1=£W(I%+1,J%):GOSUB 14180
14090 IF F%=1 THEN X=I%+D:Y=J%:GOSUB 14210
14100 Z0=Z1:Z1=£W(I%+1,J%+1):GOSUB 14180
14110 IF F%=1 THEN X=I%+1:Y=J%+D:GOSUB 14210
14120 Z0=Z1:Z1=£W(I%,J%+1):GOSUB 14180
14130 IF F%=1 THEN X=I%+1-D:Y=J%+1:GOSUB 14210
14140 Z0=Z1:Z1=£W(I%,J%):GOSUB 14180
14150 IF F%=1 THEN X=I%:Y=J%+1-D:GOSUB 14210
14160 NEXT:NEXT:ENDPROC
14180 IF Z0=Z THEN D=0:K%=1-K%:F%=1:RETURN
14190 IF(Z0-Z)*(Z1-Z)<0THEN D=(Z-Z0)/(Z1-Z0):K%=1-K%:F%=1:RETURN
ELSE F%=0:RETURN
14210 PROC£PLOTCON(K%+4,X,Y):RETURN
16680 REM --- L2-CN3D 3 dim contour --
16690 REM expects £W(I,J)
16700 DEF PROC£CN3D(K%,IM%,JM%,N%)
16710 LOCAL I%,J%,Z
16720 IF K%>9 THEN GOTO 16810
16730 PROC£INIT(K%):£XL=0:£YL=0:£XH=IM%:£YH%=JM%
16740 REM find min/max Z
16750 £ZL=£W(0,0):£ZH=£ZL
16760 FOR I%=0 TO IM%:FOR J%=0 TO JM%
16770 Z=£W(I%,J%):£ZL=FNmin(Z,£ZL)
16780 £ZH=FNmax(Z,£ZH):NEXT:NEXT
16790 PROC£AX3(0):PROC£co(£XC%,2)
16800 REM plot contours
16810 Z=(£ZH-£ZL)/(N%+1):FOR I%=1 TO N%:£2=£ZL+I%*Z
16820 PROC£CNTR(£2,IM%,JM%):NEXT:ENDPROC
16830 DEF PROC£PLOTCON(K%,I,J)
16840 PROC£PL3(K%,I,J,£2):ENDPROC
16850 REM ----------------------------

17070 REM L2-SURF
17090 DEF PROC£SURF(K%,IM%,JM%)
17100 LOCAL I%,J%,Z
17110 IF K%>9 THEN GOTO 17200
17120 PROC£INIT(K%):£XL=0:£YL=0:£XH=IM%:£YH%=JM%
17130 REM find min/max Z
17140 £ZL=£W(0,0):£ZH=£ZL
17150 FOR I%=0 TO IM%:FOR J%=0 TO JM%
17160 Z=£W(I%,J%):£ZL=FNmin(Z,£ZL)
17170 £ZH=FNmax(Z,£ZH):NEXT:NEXT
```

```
17180 PROCfAX3(0):PROCfco(fXC%,2)
17190 REM plot the surface
17200 FOR I%=0 TO IM%:PROCfMV3(I%,0,fW(I%,0))
17210 FOR J%=1 TO JM%:PROCfDR3(I%,J%,fW(I%,J%))
17220 NEXT:NEXT
17230 FOR J%=0 TO JM%:PROCfMV3(0,J%,fW(0,J%))
17240 FOR I%=1 TO IM%:PROCfDR3(I%,J%,fW(I%,J%))
17250 NEXT:NEXT:ENDPROC
17260 REM----------------------------

1727 0:REM L2-STER
17280 REM data expected in fX,fY,fZ
17290 DEF PROCfSTEREO(I%,N%)
17300 IF I%>9 THEN 17420
17310 PROCfINIT(I%)
17320 fXL=fX(0):fXH=fXL:fYL=fY(0)
17330 fYH=fYL:fZL=fZ(0):fZH=fZL
17340 FOR I%=1 TO N%
17350 fXL=FNmin(fXL,fX(I%))
17360 fYL=FNmin(fYL,fY(I%))
17370 fXH=FNmax(fXH,fX(I%))
17380 fYH=FNmax(fYH,fY(I%))
17390 fZL=FNmin(fZL,fZ(I%))
17400 fZH=FNmax(fZH,fZ(I%))
17410 NEXT
17420 fXB%=0:fYB%=0:fXS%=1130:fYS%=924
17430 fHM%=0:fVM%=0:fXC%=0:fPH=-1.3
17440 GCOL 0,1:GOSUB 17470
17450 fXB%=150:fYB%=50:fPH=-1.2
17460 GCOL 0,2:GOSUB 17470:ENDPROC
17470 PROCfAX3(1):PROCfAX3(2)
17480 PROCfMV3(fX(0),fY(0),fZ(0))
17490 FOR I%=1 TO N%
17500 PROCfDR3(fX(I%),fY(I%),fZ(I%))
17510 NEXT:RETURN
17520 REM ----------------------------

1753 0:REM L2-STSU
17540 REM expects fW(I,J)
17550 DEF PROCfSTSURF(K%,IM%,JM%)
17560 LOCAL I%,J%,Z
17570 IF K%>9 THEN GOTO 17650
17580 PROCfINIT(K%):fXL=0:fYL=0:fXH=IM%:fYH%=JM%
17590 REM find miny/max Z
17600 fZL=fW(0,0):fZH=fZL
17610 FOR I%=0 TO IM%:FOR J%=0 TO JM%
17620 Z=fW(I%,J%):fZL=FNmin(Z,fZL)
17630 fZH=FNmax(Z,fZH):NEXT:NEXT
17640 REM set up for first view
17650 fXB%=0:fYB%=0:fXS%=1130:fYS%=924
17660 fHM%=0:fVM%=0:fXC%=0:fPH=-1.3
17670 GCOL 0,1:GOSUB 17700
17680 fXB%=150:fYB%=50:fPH=-1.2
17690 GCOL 0,2:GOSUB 17700:ENDPROC
17700 PROCfAX3(1):PROCfAX3(2)
17710 FOR I%=0 TO IM%:PROCfMV3(I%,0,fW(I%,0))
17720 FOR J%=1 TO JM%:PROCfDR3(I%,J%,fW(I%,J%))
17730 NEXT:NEXT
17740 FOR J%=0 TO JM%:PROCfMV3(0,J%,fW(0,J%))
17750 FOR I%=1 TO IM%:PROCfDR3(I%,J%,fW(I%,J%))
```

```
17760 NEXT:NEXT:RETURN
17770 REM ----------------------------
```

5.3 Level One Procedures

```
10000 REM Graphs and Charts
10020 REM Copyright (C) Acornsoft 1982
10030 REM L1-2D
10040 REM
10050 REM Initialisation
10060 DEF PROCfINIT(M%):fM%=M%
10070 fXL=-10:fXH=10:fYL=-10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1:fYN%=1:fSN%=1
10100 IF M%=2 OR M%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=10102 03 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FNfI(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROCfer("Bad no. of intervals")
10220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
10230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/1E6)
10240 REM normalise interval size ^10
10250 A=10^INT(LOG(C)):B=C/A
10260 REM select nearest round interval
10270 IFB>7.1THENB=10ELSEIFB>3.2THENB=5ELSEIFB>1.4THENB=2ELSEB=1
10280 =A*B
10290 REM
10300 REM calc secondary origin,lo,hi
10310 DEF PROCfax(O,LO,HI,I)
10320 REM if axes wld cross out range,
10330 REM set sec pars so cross at lo,
10340 REM and ensure orig at integer*intvl
10350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.1)*I:LO=O
10360 REM anyway make ends at intg*intvl
10370 LO=O+I *INT((LO-O)/I+.1)
10380 HI=O-I*INT((O-HI)/I+.1)
10390 REM return value in f0,f1,f2
10400 f0=O:f1=LO:f2=HI:ENDPROC
10410 REM
10420 REM set up sector
10430 DEF PROCfse:LOCAL S%
10440 S%=(fSN%-1)MOD(fXN%*fYN%)
10450 fSXS%=fXS% DIV fXN%:fSYS%=fYS% DIV fYN%
10460 fSXB%=fXB%+S%MODfXN%*fSXS%
10470 fSYB%=fYB%+S%DIVfXN%*fSYS%:ENDPROC
10480 REM
10490 DEF FNffa(S%,M%,HI,LO):f0=HI-LO:=(S%-(M%MOD1000)-(M%DIV1000))/
f0
10500 REM
10510 DEF FNfpp(L,M%)
10520 f0=FNfpr(M%,0)*L/100
10530 ON 1+(FNfpr(M%,2)MOD 3)GOTO 10540,10550,10560
10540 =0.
10550 =-f0
```

```
l0560 =-f0/2
l0570 REM
l0580 REM Extract colour
l0590 DEF PROCfco(C%,N%):IF FNfpr(C%,6)>0 THEN GCOL 0,FNfpr(C%,N%)
l0600 ENDPROC
l0610 REM
l0620 REM scale or unscale labels
l0630 DEF FNfsu(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
l0640 i%=INT(LOG(I)+.01):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.01)
l0650 REM digits needed c% before,d% after
l0660 IF j%>0 THEN c%=j%+1:ELSE c%=1
l0670 IF NOT i%<0 THEN d%=0:p%=0:ELSE d%=-i%:p%=1
l0680 REM is sign needed
l0690 IF lo<0 OR hi<0 THEN s%=1 ELSEs%=0
l0700 REM decide scaledy'unscaled
l0710 IF NOT(f%<p%+s%+c%+d%)THEN f0=0 ELSE d%=0:p%=0:f0=FNmin(-
i%,f%-j%-s%-I)
l0720 REM format; gen if integer el fix
l0730  IF d%=0 THEN i%=&l0000+f% ELSE i%=&l0200+d%
l0740 =f%+&l00*i%
l0750 REM
l0760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
l0770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
l0780 DEF FNfpr(n%,d%)=(n%DIV(10^d%))MODl00
l0790 DEF PROCfer(a$):PRINT "Graphics package error"
l0800 PRINT a$:STOP:ENDPROC
l0810 REM
l2000 REM --- TWO DIM ROUTINES ---
l2010 DEF PROCfMOVE(X,Y):PLOT 4,FNfx(X),FNfy(Y):ENDPROC
l2020 DEF PROCfDRAW(X,Y):PLOT 5,FNfx(X),FNfy(Y):ENDPROC
l2030 DEF PROCfPLOT(K%,X,Y):LOCAL x,y
l2040 IF K%MOD8<4 THEN x=fXFA*X:y=fYFA*Y:fXSC%=fXSC%+x:fYSC%=fYSC%+
y:ELSE x=FNfx(X):y=FNfy(Y)
l2050 PLOT K%,x,y:ENDPROC
l2060 DEF FNfPOINT(X,Y)=POINT FNfx(X),FNfy(Y)
l2070 DEF FNfx(X):fXSC%=fXCR%+fXFA*(X-fSXO):=fXSC%
l2080 DEF FNfy(Y):fYSC%=fYCR%+fYFA*(Y-fSYO):=fYSC%
l2090 REM two dim scales & axes
l2100 DEF PROCfAXES(M%):LOCAL A,B,f%,d%,a%,l%,w%,L$,y%
l2110 IF M%=1 THEN GOSUB l2140 ELSE IF M%=2 THENGOSUB l2300ELSE IF
M%=3 THEN GOSUB l2400 ELSEGOSUB l2140:GOSUB l2300:GOSUB l2400
l2120 ENDPROC
l2130 REM set intervals & ends
l2140 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH)
l2150 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
l2160 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
l2170 REM set sector
l2180 PROCfse
l2190 REM scale factors,allow margins
l2200 fXFA=FNffa(fSXS%,fHM%,fSXH,fSXL):A=f0*fXFA
l2210 fYFA=FNffa(fSYS%,fVM%,fSYH,fSYL):B=f0*fYFA
l2220 REM equalise if within 30%
l2230 f0=ABS((fXFA-fYFA)/(fXFA+fYFA))
l2240 IF ABS((fXFA-fYFA)/(fXFA+fYFA))<.15 THEN fXFA=
FNmin(fXFA,fYFA):fYFA=fXFA
l2250 REM position the sixes cross
l2260 fXCR%=fSXB%+(fHM% DIV l000)+fXFA*(fSXO-fSXL)+(A-fXFA*(fSXH-
fSXL))/2
l2270 fYCR%=fSYB%+(fVM%DIVl000)+fYFA*(fSYO-fSYL)+(B-fYFA*(fSYH-
fSYL))/2
l2280 RETURN
```

```
12290 REM draw & mark off axes
12300 PROCfco(fXC%,4):PROCfMOVE(fSXL,fSYO):PROCfDRAW(fSXH,fSYO)
12310 PROCfco(fYC%,4):PROCfMOVE(fSXO,fSYL):PROCfDRAW(fSXO,fSYH)
12320 B=FNfpp(fSYI,fXP%):PROCfco(fXC%,2)
12330 FOR A=fSXL TO fSXH+fSXI/2 STEP fSXI
12340 PROCfMOVE(A,fSYO+B):PROCfDRAW(A,fSYO+B+f0):NEXT
12350 A=FNfpp(fSXI,fYP%):PROCfco(fYC%,2)
12360 FOR B=fSYL TO fSYH+fSYI/2 STEP fSYI
12370 PROCfMOVE(fSXO+A,B):PROCfPLOT(1,f0,0):NEXT
12380 RETURN
12390 REM label axes
12400 f%=FNfpr(fXP%,4):w%=fCH%DIVlE3
12410 REM decide scaled or unscaled
12420 a%=@%:@%=FNfsu(fSXL,fSXH,fSXI,f%)
12430 REM decide if OK to label all pts
12440 IFfXFA*fSXI>(l+f%)*w%ANDNOT(fM%=2ORfM%=5)THENl%=TRUEELSEl%=f
12450 REM now label
12460 VDU 5:PROCfco(fXC%,0):A=fSXH
12470 IFf0<>0THENL$="E"+STR$(-f0)ELSEL$=""
12480 d%=fSXB%+fSXS%:y%=FNfy(fSYO)+(fXP%DIVlE6)-500
12490 REPEATIFABS(A)<fSXI/2THENA=0
12500 L$=STR$(A*10^f0)+L$:B=LEN(L$)*w%
12510 d%=FNmin(FNfx(A)-B/2,d%-B-w%DIV2)
12520 PLOT4,d%,y%:PRINTL$:L$=""
12530 IFl%THENA=A-fSXI ELSEA=fSXL:l%=TRUE
12540 UNTILA<fSXL-fSXI/2
12550 REM now label Y-axis
12560 f%=FNfpr(fYP%,4):y%=fCH%MODlE3:d%=y%+fYP%DIVlE6-500
12570 @%=FNfsu(fSYL,fSYH,fSYI,f%)
12580 IFfYFA*fSYI>2*y%ANDNOT(fM%=2ORfM%=5)THENl%=lELSEl%=0
12590 PROCfco(fYC%,0):B=fSYL
12600 REPEAT PROCfMOVE(fSXO,B)
12610 IF ABS(B)<fSYI/2 THEN B=0
12630 PLOT0,-w%DIV2,d%:FORA=lTOf%:VDU8:NEXT:PRINTB*10^f0;
12640 IFl%=lTHENB=B+fSYI ELSE B=fSYH:l%=1
12650 UNTILB>fSYH+fSYI/2
12660 IF f0<>0 THEN PRINT"E";-f0
12670 VDU 4:@%=a%:RETURN

10000 REM Graphs and Charts
10010 REM Copyright(C)Acornsoft 1982
10020 REM L1-3D
10040 REM
10050 REM Initialisation
10060 DEF PROCfINIT(M%):fM%=M%
10070 fXL=-10:fXH=10:fYL=- 10:fYH=10:fZL=-10:fZH=10
10080 fXO=0:fYO=0:fZO=0:fXI%=7:fYI%=7:fZI%=7
10090 fXN%=1:fYN%=1:fSN%=7
10100 IF M%=2 OR M%=5 THEN fCH%=64 ELSE fCH%=32
10110 fYB%=0:fYS%=1023:fXB%=3*fCH%:fXS%=1279-fXB%
10120 fVM%=64064:fHM%=2*(1000*fCH%+fCH%):fCH%=1000*fCH%+32
10130 fXP%=484040225:fYP%=504040225:fZP%=540040225
10140 IF M%=1 OR M%=2 OR M%=5 THEN fXC%=10102O3 ELSE fXC%=0
10150 fYC%=0:fZC%=0
10160 fTH=1.3:fPH=-1.3
10170 ENDPROC
10180 REM
10190 REM Interval chooser
10200 DEF FNfI(N%,LO,HI):LOCAL A,B,C
10210 IF0>=N%THENPROCfer("Bad no. of intervals")
```

```
l0220 C=(HI-LO)/N%:IF0>=C THENPROCfer("Bad axis range")
l0230 C=FNmax(C,FNmax(ABS(LO),ABS(HI))/lE6)
l0240 REM normalise interval size ^l0
l0250 A=l0^INT(LOG(C)):B=C/A
l0260 REM select nearest round istterval
l0270 IFB>7.lTHENB=l0ELSEIFB>3.2THENB=5ELSEIFB>l.4THENB=2ELSEB=l
l0280 =A*B
l0290 REM
l0300 REM calc secondary origin,lo,hi
l0310 DEF PROCfax(O,LO,HI,I)
l0320 REM if axes wld cross out range,
l0330 REM set sec pars so cross at lo,
l0340 REM and ensure orig at integer*intvl
l0350 IF LO-I/2>=O OR O>=HI THEN O=INT(LO/I-.l)*I:LO=O
l0360 REM anyway make ends at intg*intvl
l0370 LO=O+I*INT((LO-O)/I+.l)
l0380 HI=O-I*INT((O-HI)/I+.l)
l0390 REM return value in f0,£1,f2
l0400 f0=O:£l=LO:£2=HI:ENDPROC
l0410 REM
l0420 REM set up sector
l0430 DEF PROC£se:LOCAL S%
l0440 S%=(£SN%-l)MOD(£XN%*£YN%)
l0450 £SXS%=£XS% DIV £XN%:£SYS%=£YS% DIV £YN%
l0460 £SXB%=£XB%+S%MOD£XN%*£SXS%
l0470 £SYB%=£YB%+S%DIV £XN%*£SYS%:ENDPROC
l0480 REM
l0490 DEF FN£fa(S%,M%,HI,LO):£0=HI-LO:=(S%-(M%MODl000)-(M%DIVl000))/
£0
l0500 REM
l0510 DEF FN£pp(L,M%)
l0520 £0=FN£pr(M%,0)*L/l00
l0530 ON l+(FN£pr(M%,2)MOD 3)GOTO l0540,l0550,l0560
l0540 =0.
l0550 =-£0
l0560 =-£0/2
l0570 REM
l0580 REM Extract colour
l0590 DEF PROC£co(C%,N%):IF FN£pr(C%,6)>0 THEN GCOL 0,FN£pr(C%,N%)
l0600 ENDPROC
l0610 REM
l0620 REM scale or unscale labels
l0630 DEF FN£su(lo,hi,I,f%):LOCAL i%,j%,c%,d%,s%,p%
l0640 i%=INT(LOG(I)+.0l):j%=INT(LOG(FNmax(ABS(lo),ABS(hi)))+.0l)
l0650 REM digits needed c% before,d% after
l0660 IF j%>0 THEN c%=j%+l:ELSE c%=l
l0670 IF NOT i%<0 THEN d%=0:p%=0:ELSE d%=-i%:p%=l
l0680 REM is sign needed
l0690 IF lo<0 OR hi<0 THEN s%=l ELSEs%=0
l0700 REM decide scaled/unscaled
l07l0 IF NOT(f%<p%+s%+c%+d%)THEN £0=0 ELSE d%=0:p%=0:£0=FNmin(-
i%,f%-j%-s%-l)
l0720 REM format; gen if integer el fix
l0730 IF d%=0 THEN i%=&l0000+f% ELSE i%=&l0200+d%
l0740 =f%+&l00*i%
l0750 REM
l0760 DEF FNmax(a,b):IFb>a THEN=b ELSE=a
l0770 DEF FNmin(a,b):IFb<a THEN=b ELSE=a
l0780 DEF FN£pr(n%,d%)=(n%DIV(l0^d%))MODl00
l0790 DEF PROCfer(a$):PRINT "Graphics package error"
```

```
10800 PRINT a$:STOP:ENDPROC
10810 REM
13000 REM --- THREE DIM LEVEL 1 --
13010 DEF PROCfMV3(X,Y,Z):PROCfPL3(4,X,Y,Z):ENDPROC
13020 DEF PROCfDR3(X,Y,Z):PROCfPL3(5,X,Y,Z):ENDPROC
13030 DEF PROCfPL3(K%,X,Y,Z):LOCAL RX,RY,RZ,x%,y%
13040 IF 3<K%MOD8 THEN RX=fSXO:RY=fSYO:fRZ=fSZO:x%=fXCR%:y%=
fYCR%:fXSC%=0:fYSC%=0
13050 RX=(X-RX)*fXFA:RY=(Y-RY)*fYFA:RZ=(Z-RZ)*fZFA
13060 x%=x%+fP11*RX+fP12*RY
13070 y%=y%+fP21 *RX+fP22*RY+fP23*RZ
13080 fXSC%=fXSC%+x%:fYSC%=fYSC%+y%
13090 PLOT K%,x%,y%:ENDPROC
13100 DEF FNfPT3(X,Y,Z):PROCfPL3(4,X,Y,Z):=POINT(fXSC%,fYSC%)
13110 REM
13120 REM 3-dim axes
13130 DEF PROCfAX3(M%):LOCAL P1,P2,P3,P4,P5,P6,LM%,DM%,a%
13140 IF M%=1 THEN GOSUB 13170 ELSE IF M%=2 THENGOSUB 13560ELSE IF
M%=3 THEN GOSUB 13700 ELSEGOSUB 13170:GOSUB 13560:GOSUB 13700
13150 ENDPROC
13160 REM set intervals,ends
13170 fSXI=FNfI(fXI%,fXL,fXH):fSYI=FNfI(fYI%,fYL,fYH):fSZI=
FNfI(fZI%,fZL,fZH)
13180 PROCfax(fXO,fXL,fXH,fSXI):fSXO=f0:fSXL=f1:fSXH=f2
13190 PROCfax(fYO,fYL,fYH,fSYI):fSYO=f0:fSYL=f1:fSYH=f2
13200 PROCfax(fZO,fZL,fZH,fSZI):fSZO=f0:fSZL=f1:fSZH=f2
13210 REM set sector
13220 PROCfse
13230 REM ranges
13240 P4=fSXH-fSXL:P5=fSYH -fSYL:P6=fSZH-fSZL
13250 REM margins,provisional scales
13260 LM%=fHM%DIV1E3:DM%=fVM%DIV1E3
13270 f1=fSXS%-LM%-fHM%MOD1E3:P1=f1/P4:P2=f1/P5
13280 f2=fSYS%-DM%-fVM%MOD1E3:P3=f2/P6
13290 REM scale equalisation
13300 IF ABS((P1-P2)/(P1+P2))<.4 THEN P1=FNmin(P1,P2):P2=P1
13310 IF ABS((P1-P3)/(P1+P3))>.2 THENGOTO 13340
13320 IF P1<P3 THEN P3=P1 ELSE f0=P1:P1=P3:IF f0=P2 THENP2=P3
13330 REM projection matrix
13340 f0=COS(fTH):fP11=-SIN(fPH):fP22=f0*fP11
13350 fP12=COS(fPH):fP21=-f0*fP12:fP23=SIN(fTH)
13360 REM screen span
13370 P6=P1*P4*ABS(fP21)+P2*P5*ABS(fP22)+P3*P6*ABS(fP23)
13380 P5=P1*P4*ABS(fP11)+P2*P5*ABS(fP12)
13390 REM shrink factor
13400 f0=FNmin(f1/P5,f2/P6)
13410 fXFA=f0*P1:fYFA=f0*P2:fZFA=f0*P3:P5=f0*P5:P6=f0*P6
13420 REM leftmost displacement(screen)
13430 IF fP11<0 THEN P1=fSXH ELSE P1=fSXL
13440 IF fP12<0 THEN P2=fSYH ELSE P2=fSYL
13450 P3=(P2-fSYO)*fYFA*fP12+(P1-fSXO)*fXFA*fP11
13460 REM lowest displacement
13470 IF fP21<0 THEN P1=fSXH ELSE P1=fSXL
13480 IF fP22<0 THEN P2=fSYH ELSE P2=fSYL
13490 IF fP23<0 THEN P4=fSZH ELSE P4=fSZL
13500 P4=(P1-fSXO)*fXFA*fP21+(P2-fSYO)*fYFA*fP22+(P4-fSZO)*fZFA*fP23
13510 REM position axis cross
13520 fXCR%=fSXB%+LM%+(f1-P5)/2-P3
13530 fYCR%=fSYB%+DM%+(f2-P6)/2-P4
13540 RETURN
```

```
l3550 REM draw & mark off axes
l3560
PROC£co(£XC%,4):PROC£MV3(£SXL,£SYO,£SZO):PROC£DR3(£SXH,£SYO,£SZO)
l3570
PROC£co(£YC%,4):PROC£MV3(£SXO,£SYL,£SZO):PROC£DR3(£SXO,£SYH,£SZO)
l3580
PROC£co(£ZC%,4):PROC£MV3(£SXO,£SYO,£SZL):PROC£DR3(£SXO,£SYO,£SZH)
l3590 B=FN£pp(£SZI,£XP%):PROC£co(£XC%,2)
l3600 FOR A=£SXL TO £SXH+£SXI/2 STEP £SXI
l3610 PROC£MV3(A,£SYO,£SZO+B):PROC£DR3(A,£SYO,£SZO+B+£0):NEXT
l3620 B=FN£pp(£SXI,£YP%):PROC£co(£YC%,2)
l3630 FOR A=£SYL TO £SYH+£SYI/2 STEP£SYI
l3640 PROC£MV3(£SXO+B,A,£SZO):PROC£DR3(£SXO+B+£0,A,£SZO):NEXT
l3650 B=FN£pp(£SXI,£ZP%):PROC£co(£ZC%,2)
l3660 FOR A=£SZL TO £SZH+£SZI/2 STEP£SZI
l3670 PROC£MV3(£SXO+B,£SYO,A):PROC£DR3(£SXO+B+£0,£SYO,A):NEXT
l3680 RETURN
l3690 REM label ends only
l3700 Pl=FN£pr(£XP%,4):P2=(£XP%DIVlE6)-500:P3=£CH%DIVlE3
l3710 REM decide scaled or unscaled
l3720 a%=@%:@%=FN£su(£SXL,£SXH,£SXI,Pl)
l3730 REM now label
l3740 VDU 5:PROC£co(£XC%,0):PROC£MV3(£SXH,£SYO,£SZO)
l3750 PLOT 0,0,P2:FOR P4=l TO PI DIV2:VDU8:NEXT:PRINT £SXH*l0^£0:IF
£0<>0 THEN PRINT "E";-£0
l3760 Pl=FN£pr(£YP%,4):P2=(£YP%DIVlE6)-500
l3770 @%=FN£su(£SYL,£SYH,£SYI,Pl)
l3780 PROC£co(£YC%,0):PROC£MV3(£SXO,£SYH,£SZO)
l3790 PLOT 0,0,P2:PRINT £SYH*l0^£0:IF £0<>0 THEN PRINT "E";-£0
l3800 Pl=FN£pr(£ZP%,4):P2=(£ZP%DIVlE6)-500
l38l0 @%=FN£su(£SZL,£SZH,£SZI,Pl)
l3820 PROC£co(£ZC%,0):PROC£MV3(£SXO,£SYO,£SZH)
l3830 PLOT 0,0,P2:FOR P4=l TO PI DIV2:VDU8:NEXT:PRINT £SZH*l0^£0:IF
£0<>0 THEN PRINT "E";-£0
l3840 VDU4:@%=a%:RETURN

l4000 REM Ll-CNTR
l4010 REM
l4020 DEF PROC£CNTR(Z,NI%,NJ%):LOCAL D,K%,X,Y,F%,I%,J%,Z0,Zl
l4030 REM scans £W(I,J)where 0:I,J:NI,NJ
l4040 REM for mesh coords where W takes
l4050 REM value Z. Calls PROC£PLOTCON(K,X,Y)
l4060 REM where X,Y in mesh coords.
l4070 FOR J%=0 TO NJ%-l:FOR I%=0 TO NI%-l:K%=l
l4080 Z0=£W(I%,J%):Zl=£W(I%+l,J%):GOSUB l4180
l4090 IF F%=l THEN X=I%+D:Y=J%:GOSUB l42l0
l4100 Z0=Zl:Zl=£W(I%+l,J%+l):GOSUB l4180
l4110 IF F%=l THEN X=I%+l:Y=J%+D:GOSUB l42l0
l4120 Z0=Zl:Zl=£W(I%,J%+l):GOSUB l4180
l4130 IF F%=l THEN X=I%+l-D:Y=J%+l:GOSUB l42l0
l4140 Z0=Zl:Zl=£W(I%,J%):GOSUB l4180
l4150 IF F%=l THEN X=I%:Y=J%+l-D:GOSUB l42l0
l4160 NEXT:NEXT:ENDPROC
l4170 REM test sides for crossing
l4180 IF Z0=Z THEN D=0:K%=l-K%:F%=l:RETURN
l4190 IF(Z0-Z)*(Zl-Z)<0THEN D=(Z-Z0)/(Zl-Z0):K%=l-K%:F%=l:RETURN
ELSE F%=0:RETURN
l4200 REM call to PROC defined by user
l42l0 PROC£PLOTCON(K%+4,X,Y):RETURN

l5000 REM Ll-BOX
```

```
15010 DEF PROC£BOX
15020 REM box round current sector
15030 PLOT 4,£SXB%,£SYB%:PLOT 1,£SXS%,0:PLOT 1,0,£SYS%
15040 PLOT 1,-£SXS%,0:PLOT 1,0,-£SYS%:ENDPROC

15050 :REM L2-HIS
15060 DEF PROC£BAR(K%,X,Y0,Y,W):LOCAL x,y0
15070 REM K%=0 left of bar at X
15080 REM K%=1 bar centered on X
15090 REM YO is base,Y is top,W width
15100 REM all in user units.
15110  IF ABS(Y0-Y)*£YFA<5 THEN ENDPROC
15120 IF Y0=£SYO THEN y0=Y0+SGN(Y-Y0)*10/£YFA ELSE y0=Y0
15130 IF K%=0 THEN x=X ELSE x=X-W/2
15140 PROC£PLOT(84,x,Y):PROC£PLOT(84,x,y0)
15150 PROC£PLOT(85,x+W,Y):PROC£PLOT(85,x+W,y0)
15160 ENDPROC
15170 REM

15180 :REM L1-SEC
15190 DEF PROC£SEC(M%,X,Y,R,A,B):LOCAL a,b,I%
15200 REM tens digit of M%: center
15210 REM 0=screen, 1=user
15220 IF M%DIV10=1 THEN X=FN£x(X):Y=FN£y(Y)
15230 REM ones digit of M%:radius
15240 IF M%MOD10=1 THEN R=R*£YFA
15250 M%=1+50*ABS(B-A):b=(B-A)*2*PI/M%
15260 A=2*PI*A:PLOT 84,X+R*COS(A),Y+R*SIN(A)
15270 FOR I%=1 TO M%:a=A+I%*b:PLOT 84,X,Y
15280 PLOT 81,R*COS(a),R*SIN(a):NEXT
15290 a=A+M%*b/2:MOVE X+R*COS(a)*.7,Y+R*SIN(a)*.7
15300 ENDPROC
15310 REM ----------------------------
```