GPLOT

A Library of Tools

for use with

the Graphical Kernel System (GKS)

T. G. Worlton
Intense Pulsed Neutron Source
Argonne National Laboratory

and

D. Mikkelson
University of Wisconsin

June 3, 1993

Table of Contents

1. Introduction 1
2. Plot Orientation 1
3. Device Nomination Routines 2
4. Basic Plot Routines 4
5. Modifying the Plot Layout 5
5.1 Reducing the Plot Size 5
5.2 Changing the Plot Origin 5
5.3 Terminating the Plot 6
5.3.1 Normal Plot Termination 6
5.3.2 Creating Multiple Plots on a Page 6
5.3.3 Changing Plot Devices 7
5.3.4 Storing Plot Segments 7
5.3.5 End All Plotting 9
5.4 Clipping the Plot 9
5.5 Framing the Plot/Subplot Areas 9
5.6 Data Transforms 10
6. Writing Text on the Plot 10
6.1 Text Size and Orientation 12
6.2 Text Fonts 12
6.3 Instruction Font 14
6.3.1 Macro Instruction Z 15
7. Drawing Axes 16
7.1 Linear Axes 17
7.2 Producing Grids and/or Tick Marks 18
7.3 Logarithmic Axes 19
8. Drawing Curves 20
8.1 Error Bars 20
8.2 Line Types 21
8.3 Colors 22
8.4 Data Markers 23
8.5 Interpolation Routines 24
9. Legends 26
10. Vectors With and Without Arrowheads 27
11. Graphic Primitives 28
12. Bar Graphs 30
13. Contouring Routines 31
14. 3 Dimensional Plotting 34
14.1 Setting the "workbox" 35
14.2 Setting the Viewing Parameters. 35
14.3 Establishing a "relative" coordinate system 37
14.4 3D Output 38
15. Graphics Input 41

15.1 Reading the Cursor Position 41
15.2 Using the Cursor to Modify the Plot 41
15.3 Printing the Screen 43
Index 44

1. Introduction

GPLOT is a collection of routines written to support generation of plots of scientific data with GKS graphics. It has been developed and tested using DEC VAX-GKS which is a GKS level 2c product. Since DEC GKS extensions to the GKS standard have been used, some special features may need to be modified before GPLOT will function correctly with another GKS package. Standard GKS calls may be mixed with calls to GPLOT routines, but you should be aware of attributes and transforms set by GPLOT. DEC made major changes to GKS in version 5.0A which makes both the source and executable modules incompatable with DEC mGKS 4.X. To convert old source code to the new GKS, change the Fortran include file from 'SYS$LIBRARY:GKSDEFS.BND' to 'SYS$LIBRARY:GKS.F', then recompile and relink on a node with a version of GPLOT compiled for DEC GKS 5.0A.

One of the primary purposes of GPLOT was to make it easy to convert graphics written for CA-DISSPLA, so many GPLOT routines have the same name and function in the same way as DISSPLA routines of the same name. To link to GPLOT include "GPLOT_DIR:GPLOT/OPT" at the end of your link statement.

GPLOT provides routines that allow you to create axes and plot data easily. It provides both linear and logarithmic axes and allows marking the data with a variety of symbols and error bars. It also provides routines for placing a legend which labels the curves on the plot. This will automatically show the correct symbol, curve type, and color for each data curve. GPLOT can also do a number of interpolations to fit experimental data. GPLOT has routines for drawing bar graphs, contour plots, and 3D surface plots with hidden-line removal. There are routines for drawing polygons, circles, ellipses, arcs of circles or ellipses, and for drawing vectors with arrow heads. These may be filled or open. GPLOT does not provide blanking of areas like DISSPLA does because GKS supports clipping only at the plot boundary, but for some devices it is possible to erase an area by filling the area with the background color. GPLOT also has some conventions which differ from DISSPLA. For instance, the tic marks are by default to the inside for GPLOT and to the outside for DISSPLA, and GPLOT draws axes at the top and right hand side of the plot area.

2. Plot Orientation

For the laser printers supported by DEC VAX GKS the default orientation is landscape. If you draw a plot intended to be in portrait orientation, you can make the following GPLOT call before calling AREA2D:

CALL ORIENT('PORTRAIT')

Otherwise your 8.5" X 11" plot will appear scaled down to fit in the landscape orientation. GPLOT will also accept the DISSPLA calls:

CALL HWROT('AUTO')
CALL HWROT('MOVIE')
CALL HWROT('COMIC')

to control plot orientation.

For screen devices such as the VAXstation or TK4014 terminal, the plot orientation cannot be changed. The Tektronix 4014 output only device which is used to produce plots on the DEC LN03-Plus also does not support changing to portrait orientation.

3. Device Nomination Routines

Before a plot can be generated, one or more workstation devices must be nominated through a call to GSTART, GMENU, or a specific device nomination routine. The devices for a given site are defined in the routine GPLOT_SRC:GDINIT.DAT. Information in this file allows plot files to be sent to a particular print queue automatically. Plot files which are to be sent to a print queue automatically are created in the SYS$SCRATCH directory and will be deleted after printing. If GDINIT does not specify a print queue, the output file will be created in your current directory and not printed. Some options in the menu, such as *Postscript-file* or *Tektronix-file* are designed to create an output file for later use. Encapsulated Postscript (EPS) files are particularly useful since they may be included in word processor documents. The figures in this manual were created by GPLOT as Encapsulated Postscript files and linked to a MASS11 document.

You can create your own customized menu by copying the file GPLOT_SRC:GDINIT.DAT to SYS$LOGIN:. And then issuing the command "EDIT_GMENU" to modify it as desired. When adding a new device, the workstation type number can be selected from a list obtained from a data file, GPLOT_DIR:WSTYPES.V50. For plotters to print automatically you will also need the queue name and the name of any form needed to put the printer into the graphics mode. The device name for a hardcopy device usually is the queue name, but can be any name of your choosing. When running in batch mode you will need to include the command:

$ DEFINE PLOTTER device-name

where "device-name" is the name you have specified in setting up your menu.

Information about each device can be obtained during the GMENU display by selecting the device and pressing the help key. The system manager uses a text editor to store site-dependent help information in the file SYS$HELP:GPLOT.HLP then uses the LIBRARY command to insert the new information in SYS$HELP:GPLOT.HLB.

It is recommended that you make the following the first call to GPLOT:

CALL GSTART

GSTART will look at the logical names SCREEN and PLOTTER to see which device(s) to activate. If neither of these is defined to be a valid device, and the processing mode is interactive, GSTART will call GMENU to produce a menu of available graphics devices. If both SCREEN and PLOTTER are defined as valid devices when GSTART is called, the plot will be generated on both at the same time. To stop getting plots on a particular device, deassign SCREEN and/or PLOTTER or define them to be different devices. For instance if you have a program, GRAPH, which calls GSTART you could use the following series of commands to preview a plot and then make a hard copy on the LW print queue (assuming this is a valid queue on your node).

```
$ DEFINE SCREEN TK4014
$ RUN GRAPH
$ DEASSIGN SCREEN
$ DEFINE PLOTTER LW
$ RUN GRAPH
```

If you omitted the second and third commands, the two plots would be generated simultaneously. Note that when plotting simultaneously to two devices, GPLOT always uses software characters so that the screen plot and hardcopy plot will match. When plotting to a postscript printer only, postscript characters will be used, resulting in faster plotting and higher quality characters.

We do not currently have a way of plotting CGM metafiles because DEC GKS only supports CGM output, not input. GKS metafiles can be plotted using a postprocessing routine (use the command POSTGKS to run the GKS postprocessor).

Although calling GSTART is the recommended method of nominating a graphics device, it is possible to nominate many devices with a direct subroutine call. Most of these are included to emulate CA-DISSPLA. The device nomination routine calls currently supported are:

Device Call	**Device**
CALL GKSM	Produce GKS Graphics Metafile.
CALL COMPRS	Same as GKSM
CALL CGMCO	Produce CGM Character metafile
CALL CGMTO	Produce CGM Clear Text metafile
CALL POSTSCRIPT	Apple Laser Writer
CALL PSCRPT(X0,Y0,XW,YW)	Same as POSTSCRIPT (arguments ignored)
CALL TK4014F	Graphics file to be printed on TK4014 emulator
CALL LN03TK	Same as TK4014F
CALL TK4014	Graphics terminal with TK4014 emulation
CALL VAXSTATION	Produce plot on VAXstation window
CALL REGIS(I1,I2)	Plot on VT240 (I2=0) or VT241 (I2=1) (I1 is ignored)
CALL VT330	Plot on VT330.
CALL VT340	Plot on VT340.
CALL MOTIF	Motif using DEC GKS widget
CALL MOTIFW	Motif using user widget
CALL XWNDOW(IWKID,IARGRY,NARG)	(For DISSPLA compatibility
CALL GMENU	Choose device from screen menu
CALL GSTART	(See above)
CALL PRINTER_SETUP(NAME, TYPE, WKCAT, QUE, FORM)	Define printer for screen dumps.

4. Basic Plot Routines

A basic plot can be generated using the following basic GPLOT calls:

CALL GSTART
(GSTART will select plotting device(s))

CALL PAGE (PAGEX,PAGEY)
(PAGEX and PAGEY are the dimensions of the page. If there is no call to PAGE, then PAGEX and PAGEY take the default values of 8.5 and 11.0. The page orientation will be landscape unless ORIENT('PORTRAIT') or HWROT('AUTO') is called before GSTART)

CALL AREA2D (XAXIS,YAXIS)
(XAXIS and YAXIS are the lengths of the axes. AREA2D is one routine which must always be called, and it will activate the graphics devices which have been previously selected if none are currently active)

CALL XNAME (XNAME,LXNAME)
(XNAME is the name of a text string containing the x-axis label. LXNAME is the length of that name. If XNAME ends in a dollar sign ($), LXNAME can be specified as 100 and the dollar sign will be ignored. Text can be specified in single quotes or as a character string.

CALL YNAME (YNAME,LYNAME)

CALL GRAF (XORIG,XSTEP,XMAX,YORIG,YSTEP,YMAX)
(XORIG is the x-data value at the origin of the x-axis. XSTEP is the change in x-data value per axis division, and XMAX is the x-data value at the end of the AXIS. If 'SCALE' is used for XSTEP and/or YSTEP, GPLOT will calculate appropriate values to use.)

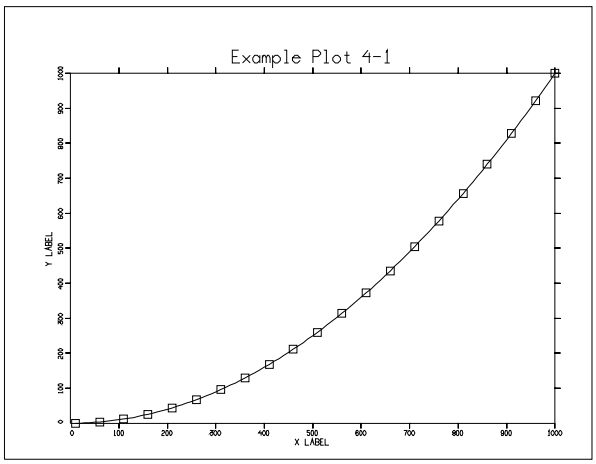
CALL CURVE (XARRAY,YARRAY,NPNTS,IMARK)
(XARRAY and YARRAY are the arrays of x-data values and y-data values, NPNTS is the number of data points in the arrays, and IMARK is an indicator of the data marker frequency. If IMARK=0, no symbols will be plotted and if IMARK > 0 symbols will be plotted every IMARKth point. If IMARK < 0 symbols will be plotted, but no curve will be drawn.)

CALL ENDPL (IPLOT)
(ENDPL updates the plot and if the mode is interactive and the device is of type input/output, ENDPL pauses for the user to view the plot. When the user types a RETURN, it closes and deactivates the workstation and zeros some of the plot parameters, so that AREA2D and GRAF must be called again to set up another plot**.** If IPLOT is greater than zero, information about the plot routine will be written in the lower left corner of the plot. If Workstation Independent Segment Storage (WISS) is active and there is no hardcopy device active, the user will be asked if he wants a hardcopy of that plot frame.)

CALL GSTOP

A simple plot created using the above routines is illustrated in Example 4-1.

```
PROGRAM GPE1  ! Example 4-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/

CALL GSTART                  ! Nominate Device
CALL PAGE(11.0,8.5)          ! Specify page size
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL AREA2D(9.0,6.5)         ! Define Subplot area
CALL XNAME('X LABEL$',100)   ! Define X Axis label
CALL YNAME('Y LABEL$',100)   ! Define Y Axis label
CALL HEADIN('Example Plot 4-1$',100,2.,1)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)

!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*10.0
  Y(I) = 0.1*I*I
END DO

CALL CURVE(X,Y,100,5)
CALL ENDPL(0)
CALL GSTOP

CALL EXIT
END
```



5. Modifying the Plot Layout

The values of PAGEX and PAGEY set by the call to PAGE determine the aspect ratio of the plot. If PAGE is not called, an 8.5" X 11.0" page size is assumed. [1] The axes lengths set by the call to AREA2D together with the page variables determine the sub plot area within the plot area. These values are used to calculate the normalization viewport in Normalized Display Coordinates (NDC) for the plot area and for the sub plot area. The workstation window and viewport are then calculated for these same transforms using the currently active workstation(s). When the data values at the axis extremes have been set by calls to GRAF or XGRAXS and YGRAXS, the data transform is calculated.

5.1 Reducing the Plot Size

GPLOT will generate the largest plot that can be produced on the workstation surface while maintaining the aspect ratio indicated by the two PAGE parameters. If you want to generate a smaller plot, you can scale the size of the plot through a call to BLOWUP. For example, the call:

CALL BLOWUP(0.4)

will cause the plot to be scaled to 40% of the page size. All sample plots in the manual have been scaled to 40% so that they will be a convenient size for inclusion in this manual. All sample plots also use portrait orientation for simple inclusion in this manual also.

5.2 Changing the Plot Origin

GPLOT will try to center the sub plot area within the plot area.[2] For special cases such as when you want to produce multiple plots on a page, you can use one of the following calls to change the sub-plot origin:

- -

1. All examples in the manual have been drawn with portrait orientation and with 40% size reduction for convenience in inclusion in the manual.
2. GPLOT will actually shift the plot a little to the right and down to allow for a plot summary on the left and a heading on the top.

CALL PHYSOR(XPHYS,YPHYS)
(Where XPHYS and YPHYS are the coordinates of the origin of the sub-plot area with respect to the corner of the page)

CALL OREL(XOREL,YOREL)
(Where XOREL, YOREL is the offset of the sub-plot origin with respect to the previous origin)

5.3 Terminating the Plot

5.3.1 Normal Plot Termination

The normal method of terminating a plot is to use the call:

CALL ENDPL(IPLOT)

where "IPLOT" is the plot number if a plot summary is desired on the plot and is zero otherwise. Calling ENDPL will close the current plot, but leave the devices active and GKS open. The last call to ENDPL in the program should be followed by a call to GSTOP.

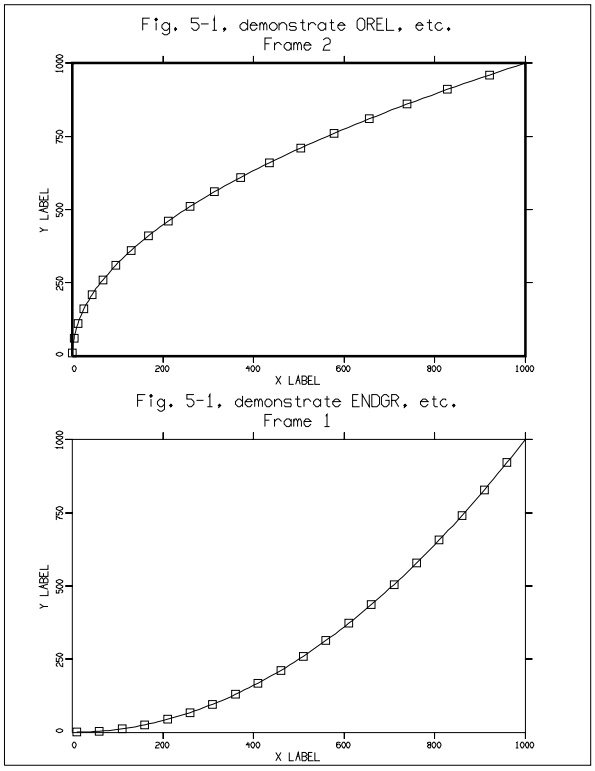
5.3.2 Creating Multiple Plots on a Page

Calls to change the plot origin are particularly useful when you want to create multiple subplots on a page. Each subplot except the last one on a page should be terminated with the call:

CALL ENDGR(IPLOT)

The call to ENDGR will terminate the current plot, but not eject a page so you can produce multiple plots on the same page. Note that a call to AREA2D or YGRAXS together with a call to PHYSOR or OREL determines the portion of the page which will contain the plot. An example of using ENDGR, PHYSOR, and OREL to put multiple plots on a page is given in Example 5-1.

```
PROGRAM GPE2  ! GPLOT Example 5-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,200.,1000./,YORIG,DY,YMAX/0.0,250.0,1000.0/

CALL GSTART
CALL PAGE(8.5,11.0) ! Specify page size
CALL BLOWUP(0.4)
CALL ORIENT('PORTRAIT')
! Subplot one
CALL PHYSOR(1.0,0.5)
CALL AREA2D(6.5,4.2) ! Define Subplot area
CALL XNAME('X LABEL$',100)   ! Define X Axis label
CALL YNAME('Y LABEL$',100)   ! Define Y Axis label
CALL HEADIN('Fig. 5-1, demonstrate ENDGR, etc.$',100,1.5,2)
CALL HEADIN('Frame 1$',100,1.5,2)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*10.0
  Y(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)
CALL ENDGR(0) ! End first subplot
! Subplot two
CALL OREL(0.0,5.4)
CALL AREA2D(6.5,4.2) ! Define Subplot area
CALL XNAME('X LABEL$',100) ! Define X Axis label
CALL YNAME('Y LABEL$',100) ! Define Y Axis label
CALL HEADIN('Fig. 5-1, demonstrate OREL, etc.$',100,1.5,2)
CALL HEADIN('Frame 2$',100,1.5,2)
CALL THKFRM(4.0)
CALL FRAME
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
! Compute and plot a quadratic curve
DO I=1,100
  Y(I) = I*10.0
  X(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)

CALL GSTOP
CALL EXIT
END
```



5.3.3 Changing Plot Devices

An interactive program can change plot devices by calling **ENDDEV** to close the current devices and then calling GSTART to choose a new device. The call to ENDDEV would normally follow a call to ENDPL, but would precede any call to GSTOP.

5.3.4 Storing Plot Segments

GKS allows groups of output primitives to be stored in segments. Plot segments can be highlighted, deleted, restored, set invisible, copied to a workstation, and set detectable or undetectable. Segments are particularly useful for windowing terminals since when a window is covered up, only those parts of the plot stored in segments can be restored when the window area is uncovered. They are also useful for interactive graphics since they allow you to select individual plot elements and modify or replace those elements.

GPLOT has some routines designed to make working with segments easier. In some cases GPLOT will create segments automatically. In other cases you will want to create segments yourself. To create a segment, you can use either the GPLOT routine MAKESEG or the GKS routine GCRSG. They are called as follows:

CALL MAKESEG(NEWSEG)

CALL GCRSG(IDSEG)

The difference in these two routines is that MAKESEG automatically selects the segment number, NEWSEG, and returns the number to the calling routine, whereas GCRSG uses the calling parameter to create a segment with the number IDSEG. Other GPLOT segment routine calls are:

CALL SEGBEG (IDSEG,XFIX,YFIX)

CALL SEGEND

CALL SEGDEL (IDSEG)

CALL SEGDLA

CALL SEGREN (IDOLD,IDNEW)

CALL SETPID (IDPICK)

CALL SETVIS (LSTAT)

CALL SETHI (LSTAT)

CALL SETDET (LSTAT)

CALL SETPRI (IPRI)

CALL CHVIS (IDSEG,LSTAT)

CALL CHHI (IDSEG,LSTAT)

CALL CHDET (IDSEG,LSTAT)

CALL CHPRI (IDSEG,IPRI)

Some of these routines are direct alternates to GKS routines and the GKS routine can be called instead. In the above routines, LSTAT is the character string 'ON' or 'OFF', and IDSEG, IDOLD, and IDNEW are integer segment numbers. IPRI is the segment priority. These routines allow you to set or change segment visibility (VIS), segment highlighting (HI), segment detectability (DET), segment priority (PRI), and segment pick identifier number (PID). They also allow you to rename (REN) a segment. The routines which begin with "SET" affect segments created after the call to that routine. The routines that begin with "CH" (change) affect a previously created segment, IDSEG. There are also routines for inquiring about the current setting for these segment parameters. These routines are called as follows:

IEXIST = ISGEX (IDSEG)

IFVIS = ISGVIS (IDSEG)

IFDET = ISGDET (IDSEG)

IFHI = ISGHI (IDSEG)

IPRI = ISGPRI (IDSEG)

5.3.5 End All Plotting

When all plotting is finished, the following call should be made:

CALL GSTOP

GPLOT will also accept the DISSPLA "CALL DONEPL" statement, but there GPLOT will not automatically send the plot to a print queue.

5.4 Clipping the Plot

Points lying outside the page window will always be clipped. To clip all points lying outside the subplot area defined by AREA2D, include the statement

CALL GRACE(0.0)

The default clipping can be restored by calling GRACE with an argument greater than 0.5. An alternative is to use the GKS routine GSCLIP to turn clipping on and off.

5.5 Framing the Plot/Subplot Areas

By default, GPLOT will draw a border around the plot area. To eliminate this plot border, make the call:

CALL NOBRDR

GPLOT also normally draws a frame around the subplot area with tick marks on both sides and on the top and bottom (unlike DISSPLA). The frame around the subplot area will normally be at the default line width, but can be made thicker with the following calls:

CALL FRAME

CALL THKFRM (THKNSS)

where THKNSS is approximately the thickness in inches. In calculating this, GPLOT assumes a nominal thickness of 0.01". An example of the use of FRAME and THKFRM is shown in the second subplot of Example 5-1. A value of THKNSS=0.04 means that the frame is approximately four times the normal thickness. If THKNSS is greater than or equal to 1.0 it is taken as a factor to multiply the nominal thickness, so THKNSS=4.0 yields a frame four times the nominal thickness.

5.6 Data Transforms

GPLOT defines three GKS transforms. They transform from the page or subplot area in inches or data units to the workstation viewport. The GKS Workstation viewport is always set to be the same as the GKS normalization window. The page transform attempts to produce the largest plot possible on the display surface without changing the aspect ratio of the plot. The subplot area within the page is defined by the x and y axes in the call to area2d. The origin of the subplot area with respect to the page area (in inches) is referred to as the physical origin. GPLOT provides routines to transform between data units and position with respect to the subplot area. These routines are function routines patterned after those in DISSPLA. To obtain the position of a coordinate point (XVAL,YVAL) in inches relative to the origin, use the calls:

XINCH = XPOSN (XVAL,YVAL)

YINCH = YPOSN (XVAL,YVAL)

To find the data value of a point at a given position on the plot, use the transforms:

XVAL = XINVRS (XINCH,YINCH)

YVAL = YINVRS (XINCH,YINCH)

These routines are particularly useful when doing input from a graphics workstation or when you need to label data points.

6. Writing Text on the Plot

It is often desirable to write additional text on the plot. This can be done with the GKS routine GTX or with one of the GPLOT routines. The GPLOT routines for writing text are the following:

CALL CHRMSG (XPOS,YPOS,CSTRING)

CALL MESSAG (STRING,LSTRING,XPOS,YPOS)

CALL RLMESS (STRING,LSTRING,XDATA,YDATA)

CALL HEADIN (STRING,LSTRING,HTMULT,NLINES)

CALL REALNO (ANUM,IPLACE,XPOS,YPOS)

CALL RLREAL (ANUM,IPLACE,XDATA,YDATA)

CALL INTNO (INUM,XPOS,YPOS)

CALL RLINT (INUM,XDATA,YDATA)

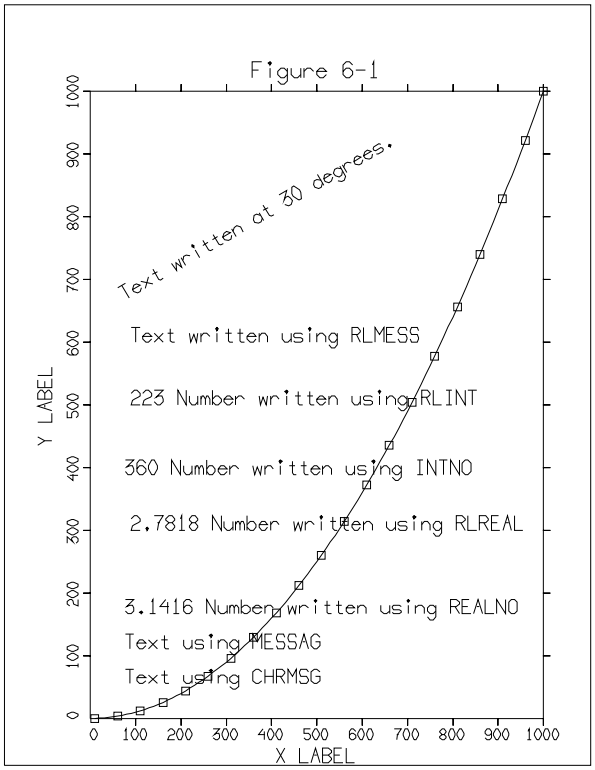
CHRMSG is the basic GPLOT routine for text output and all the other GPLOT text routines call CHRMSG. CHRMSG has the same arguments as the GKS text output routine GTX, but CHRMSG

includes the capability of switching fonts within the text string or changing character size or position with embedded instructions. This makes it easier to do subscripts, superscripts, special symbols, etc. The other routines are included to emulate DISSPLA. If you are using routines with text strings and do not wish to count the length, you can end the string with a dollar sign ($) and set LSTRING=100. For GPLOT you can also specify LSTRING as 0. The parameter CSTRING used in CHRMSG is a character string so no string length needs to be passed if CHRMSG is used.

In the above routines the text position XPOS and YPOS are in terms of axes units (inches or centimeters), and XDATA and YDATA are in terms of data units.

HEADIN is used to center lines of text above the subplot area, so no positioning is needed, but you must pass the total number of heading lines to be used in the variable NLINES. You can also expand the height of the heading characters above the default by setting HTMULT > 1.0. The last four routines are used for writing real numbers and integers on the plot. IPLACE is used to indicate the number of decimal places to use in writing real numbers on the plot. If IPLACE is negative, the number is written in exponent form. Examples of each of these text routine calls are shown in Example 6-1

```
PROGRAM GPE3                ! Example 6-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/

CALL GSTART                 ! Nominate Device
CALL PAGE(8.5,11.0)         ! Specify page size
CALL BLOWUP(0.4)
CALL ORIENT('PORTRAIT')
CALL AREA2D(6.5,9.0)        ! Define Subplot area
CALL XNAME('X LABEL$',100)  ! Define X Axis label
CALL YNAME('Y LABEL$',100)  ! Define Y Axis label
CALL HEADIN('Figure 6-1$',100,2.,1)
CALL HEIGHT(0.25)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*10.0
  Y(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)
CALL CHRMSG(0.5,0.5,'Text using CHRMSG')
CALL MESSAG('Text using MESSAG',100,0.5,1.0)

CALL REALNO(3.1416,4,0.5,1.5)
CALL CHRMSG('ABUT','ABUT',' Number written using REALNO')

CALL RLREAL(2.7818,4,90.,300.)
CALL CHRMSG('ABUT','ABUT',' Number written using RLREAL')

CALL INTNO(360,0.5,3.5)
CALL CHRMSG('ABUT','ABUT',' Number written using INTNO')

CALL RLINT(223,90.,500.)
CALL CHRMSG('ABUT','ABUT',' Number written using RLINT')

CALL RLMESS('Text written using RLMESS$',100,90.,600.)
CALL ANGLE(30.)
CALL CHRMSG(0.5,6.0,'Text written at 30 degrees.')

CALL ENDPL(0)
CALL GSTOP
CALL EXIT
END
```



It is sometimes useful to know the length of a text string written on a plot for text positioning, etc. The function calls XMESS, XINT, and XREAL can be used to get the length of a text string output by MESSAG, INTNO, RLINT, REALNO, or RLREAL with the current font and character height. They are called as follows:

XLENGTH = XMESS(LMESS,IMESS)

XLENGTH = XINT(INUM)

XLENGTH = XREAL(ANUM,IPLACE)

6.1 Text Size and Orientation

The Height and angle of the text may be altered by calling the routines:

CALL HEIGHT (HITE)

CALL ANGLE (ANG)

These routines affect the height and angle of all text written to the screen after they are called. Example 6-1 shows the effect of a call to ANGLE. HEIGHT was called in this example to make the scaled down plot used in this manual still readable.[3] If calls to the GKS routines GSCHH (HITE) and/or GSCHUP (XVEC,YVEC) are mixed with these calls, they may produce unpredictable results unless you are sure what world coordinate transformation is being used at the time.

6.2 Text Fonts

DISSPLA provides a large number of software fonts and some of the DISSPLA font names have been included as valid calls in GPLOT for ease in conversion of graphics programs, but there may be little similarity in the appearance of the two fonts. The font names from DISSPLA which can be used include: **CARTOG, COMPLX, DUPLX, FASHON, FUTURA, SCMPLX, SERIF, SIMPLX, SWISSB, SWISSL, SWISSM, TRIPLX, GOTHIC, GREEK, MATHEMATICAL, RUSSIAN, SCRIPT, SPECIAL,** and **STANDARD**. Some of these select an alphabet instead of a font style and thus must be generated in software. Others may have the desired effect on a Postscript printer, but have no effect on other devices.

The actual fonts available are both device and GKS implementation specific. For complete control over the font style and precision, use the GKS call:

CALL GSTXFP (FONTID,PRECIS)

where FONTID is an integer indicating the font to be selected, and PRECIS is an integer indicating the precision required. For highest precision, specify PRECIS = 2 (stroke precision). For medium precision, specify PRECIS = 1 (character precision), and for low precision, specify PRECIS = 0 (String precision). These precision constants are a part of the GKS standard, but the FONTID numbers are device and implementation specific.

DEC has used negative numbers for all their fonts except the default font, FONTID = 1, to avoid conflict with future GKS standard font numbers. VAX GKS provides a number of software fonts which are available on any device. To draw text using software fonts requires that the processor calculate all the strokes to be made to draw each character and sends these vectors to the device. Hardware fonts are created by sending a font selection command to the device and then sending the ASCII character codes to the device. Using software fonts can increase the time taken to generate a plot by a factor of ten or more. For speed and efficiency it is always preferable to use hardware fonts. Software fonts will provide higher quality and placement precision unless you can design your graph for the hardware fonts available on a specific plotter or you are using a Postscript plotting device.

---

3. The current version of DEC GKS produces software characters which are two-thirds of the requested height. The Postscript driver produces characters of the correct height, so mixing hardware and software characters on a postscript printer produces characters of differing heights.

Postscript plotters are able to generate a variety of fonts in any size and orientation. Postscript is recommended when you want to have higher quality plots, or any time you are going to need a lot of large or heavy text on the plot. Postscript plots can also be printed on a typesetter at even higher quality. DEC VAX-GKS supports a number of software fonts which are described in Appendix G of the DEC GKS Reference Manual. As mentioned, Postscript fonts can be drawn in any size and orientation. Since it is difficult to remember the font numbers, a routine has been included in GPLOT to allow selecting the fonts by name and attribute, using the call:

CALL FONT ('font_name','font_attribute'),

where the font attributes are 'NORMAL', 'BOLD', 'SLANT' or 'ITALIC', and 'BOLD ITALIC'. The font names for the Apple Laserwriter Plus are shown in Table 6.1 with the font name written in the indicated style.

Font	Attribute	Sample Text
Helvetica	Normal	ABCDEFGabcdefg
Helvetica	**Bold**	**ABCDEFGabcdefg**
Times Roman	Normal	ABCDEFGabcdefg
Times Roman	**Bold**	**ABCDEFGabcdefg**
Times Roman	*Slant*	*ABCDEFGabcdefg*
Times Roman	***Bold Italic***	***ABCDEFGabcdefg***
Palatino	Normal	ABCDEFGabcdefg
Bookman	Normal	ABCDEFGabcdefg
Helvetica Narrow	Normal	ABCDEFGabcdefg
Chancery	*Normal*	*ABCDEFGabcdefg*
Symbol	Normal	∝∞÷ΔΨΦΓαβχδεφγ
Dingbats	Normal	✡✢✂✌✣✦✧❀✲❀❄❄❄✳

Table 6.1 POSTSCRIPT Font Samples

It should be noted that the Laserwriter Plus has more fonts than the regular Laserwriter, and other Postscript printers may have some differences in the fonts available.

6.3 Instruction Font

The advantage of using the GPLOT text routines instead of the GKS routine GTX for text output is that some capability for special formatting and font-switching within the string have been included. The routine **MXiALF (LALPHA,LCHAR),** where "**i**" can be 1-6, can be used to define up to six special font-switching or instruction characters. In these routines LALPHA is a character or byte string giving the name of the font or character set to be used and LCHAR is a character which is used as a flag to activate the font change or instruction interpretation. Two special variations of **MXiALF** (**BASALF** and **MIXALF**) have been defined which use the left and right parenthesis as their font switch flag characters, so they only require the parameter LALPHA. It is also possible to map one character to another by using the KBMAP routine as shown in the next example.

If LALPHA = 'INSTRUCTION' the text following the occurrence of the corresponding flag character will be interpreted as instructions until another flag character is found. The instructions which have been implemented are:

Ex.....Elevate text baseline a fraction "x" of the character height.

Lx.....Lower text baseline a fraction "x" of the character height.

Bi.....Backspace "i" characters.

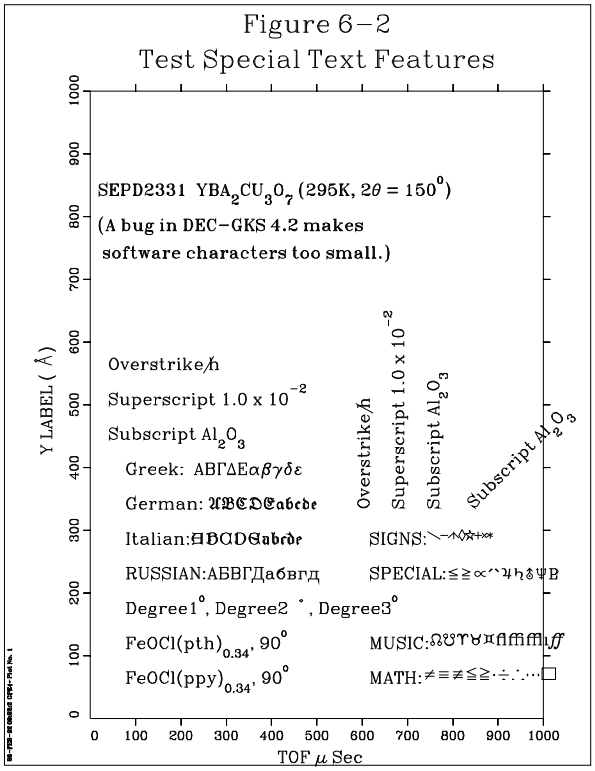
Hx.....Multiply character height by factor "x".

Ax.....Move by a fraction of character height "x".

Zn.....Insert Macro Instruction "n" (See below).

The "x" above is either a real number or the letter "X". If the letter "X" is used the corresponding value will reset the variable to the default value. These instructions are interpreted by the subroutine ANLZIN which is called by CHRMSG. All the GPLOT text routines go through CHRMSG so may use these special features for font switching,etc. Example 6-2 demonstrates these instructions.

```
PROGRAM GPE4      ! Example 6-2
CHARACTER TITLE*200
TITLE(1:) = 'SEPD2331  YB!A#LH.8!2#EXHX!C!U#LH.8!3#EXHX!O#LH.8!7'//
1'#EXHX! (295K, 2*Z! = 150#EH.6!O#EXHX!)  '
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/
CALL GSTART             ! Select plotting device
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL PAGE(8.5,11.0)     ! Specify page size
LL AREA2D(6.5,9.0)      ! Define Subplot area
CALL HEIGHT(0.25)       ! Set text size
CALL DUPLX              ! Select base font
CALL MX1ALF('STANDARD','!') ! "!" switch to STANDARD character set
CALL MX2ALF('L/CSTD','?')     ! "?" switches to "Lower-case"
characters
CALL MX3ALF('GREEK','*')      ! "*" switches to "GREEK" characters
CALL MX4ALF('INSTRUCTION','#')  ! Define instruction switch
CALL MX5ALF('GERMAN','&') ! "&" switches to German characters
CALL MX6ALF('ITALIAN','~')  ! "~" switches to Italian characters
CALL KBMAP ('^',140)    ! map "^" to character 140 (Angstrom)
CALL KBMAP ('\',176)    ! map "\" to character 176 (degree)
CALL ZUSE('LH.7')
CALL Z1USE('EH.7')
CALL Z2USE('EXHX')
CALL Z3USE('EH.5!O#EXHX')
! Set up the plot with axis labels and title using special features
CALL XNAME('TOF *l#MX! Sec$',100)   ! Define X Axis label
CALL YNAME('Y LABEL (^)$',100)       ! Define Y Axis label
CALL HEADIN('Figure 6-2$',100,1.5,2)  ! Output heading lines
CALL HEADIN('Test Special Text Features$',100,1.5,2)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
! Add more text to the plot
CALL CHRMSG(0.5,0.5,'FeOCl(ppy)#z!0.34#z2!, 90#Z3')
CALL CHRMSG(0.5,1.0,'FeOCl(pth)#z!0.34#z2!, 90#Z3')
CALL CHRMSG(0.5,1.5,'Degree1#EH.5!O#EXHX!, Degree2\, Degree3#Z3')
CALL CHRMSG(0.5,2.5,'Italian:~ABCDEabcde!')
CALL CHRMSG(0.5,3.0,'German: &ABCDEabcde!')
CALL CHRMSG(0.5,3.5,'Greek:  *ABCDEabcde!')
CALL MX6ALF('RUSSIAN','~')  ! Define instruction switch
CALL CHRMSG(0.5,2.0,'RUSSIAN:~ABCDEabcde!')
CALL MX6ALF('MATH','~') ! Define instruction switch
CALL CHRMSG(4.0,0.5,'MATH:~ABCDEabcde!')
CALL MX6ALF('MUSIC','~')    ! Define instruction switch
CALL CHRMSG(4.0,1.0,'MUSIC:~ABCDEabcde!')
CALL MX6ALF('SPECIAL','~')  ! Define instruction switch
CALL CHRMSG(4.0,2.0,'SPECIAL:~ABCDEabcde!')
CALL MX6ALF('SIGNS','~')    ! Define instruction switch
CALL CHRMSG(4.0,2.5,'SIGNS:~ABCDEabcde!')
CALL CHRMSG(0.25,5.0,'Overstrike h#B1!/')
CALL CHRMSG(0.25,4.5,'Superscript 1.0 x 10#EH.7!-2#EXHX!')
CALL ANGLE(90.)
CALL CHRMSG(4.0,3.0,'Overstrike h#B1!/')
CALL CHRMSG(4.5,3.0,'Superscript 1.0 x 10#EH.7!-2#EXHX!')
CALL CHRMSG(5.0,3.0,'Subscript Al#LH.7!2#EXHX!O#LH.7!3')
CALL ANGLE(45.0)
CALL CHRMSG(5.5,3.0,'Subscript Al#LH.7!2#EXHX!O#LH.7!3')
CALL ANGLE(0.0)
CALL CHRMSG(0.25,4.0,'Subscript Al#LH.7!2#EXHX!O#LH.7!3')
CALL SCMPLX
CALL CHRMSG(0.1,7.0,'(A bug in DEC-GKS 4.2 makes')
CALL CHRMSG(0.1,6.6,' software characters too small.)')
CALL MESSAG(TITLE,100,0.1,7.5)
CALL ENDPL(1)
CALL GSTOP
CALL EXIT
END
```



6.3.1 Macro Instruction Z

It is possible to define Macro Instructions through one of the following calls to ZUSE or ZnUSE:

CALL ZUSE (macro,lmacro)
CALL ZnUSE (macro,lmacro)

where "n" in ZnUSE is the number of the macro string (valid numbers are 1-6) to be defined, "macro" is the string to be substituted, and "lmacro" is the length of the macro string. The call to ZnUSE must precede the call to the text output routine which will use the macro string. This allows a two or three character sequence to substitute for a much longer sequence when the output routine encounters a switch character followed by "Z" plus (optionally) the macro number.

7. Drawing Axes

A number of GPLOT routines are available for drawing axes. There are two routines for labelling the axes which are called as follows:

CALL XNAME (XNAME,LNAME)

CALL YNAME (XNAME,LNAME)

where XNAME and YNAME are the labels for the axes and LNAME is the number of characters in the name. If you don't want to count characters, you can add a '$' at the end of BNAME and set LNAME = 100. If XNAME and YNAME are specified as variables instead of text constants in single quotes, they should be CHARACTER variables. The axes which are generated will have labels and numbers on the bottom and left axes, and will have axes with tic marks on the right and top. This differs from the DISSPLA convention which places the tic marks on the outside by default. Additional axes may be drawn by the calls

CALL XGRAXS (XORIG,XSTEP,XMAX,XAXIS,XNAME,LXNAME,XPOS,YPOS)

CALL YGRAXS (YORIG,YSTEP,YMAX,YAXIS,YNAME,LYNAME,XPOS,YPOS)

The arguments have the same meanings here as described for GRAF, XNAME, and YNAME, except that if LXNAME or LYNAME are negative the numbers and labels will be drawn on the opposite side of the axes. Example Plot 6 shows how YGRAXS can be used together with OREL or PHYSOR to produce multiple subplots on a page. The values at the axis tic marks can be set to integers by the calls

CALL XINTAX

CALL YINTAX

CALL INTAXS

It should be noted that the values XSTEP and YSTEP can be replaced by the string 'SCALE' to have the program calculate appropriate step sizes. If you want the tic marks to go to the outside, you can reverse them with the following calls:

CALL XREVTK

CALL YREVTK

The axes drawn will join at the XORIG and YORIG values, but the axes can be made to cross at x = 0.0, y = 0.0 by calling **CROSS**. This is sometimes useful when one or both axes start at negative values and go to positive.

By default the axis labels will be horizontal on the x-axis and vertical on the y-axis. The angle of these labels can be changed with the routines:

CALL XAXANG (ANG)

CALL YAXANG (ANG)

where ANG is the angle from the horizontal in degrees. Example 7-1 shows how these routines can be used to modify the tic marks and axis labels.

Another option is to eliminate the labels at the beginning or the end of the axes, or both. This can be done with the calls:

CALL XAXEND (option)

CALL YAXEND (option)

where "option" is one of the following values: 'NOFIRST','NOLAST', or 'NOENDS'.

Another option is to move the first and/or last tick label inward on the axis to prevent overlap with other graphic elements. The subroutines for doing this are:

CALL XNMADJ (option)

CALL YNMADJ (option)

WHERE "option" is one of the strings 'FIRST', 'LAST', 'ENDS', or 'NONE'. The default is 'FIRST' to move the numbers at the origin to prevent overlap. To turn this off you must call these routines with option 'NONE'.

7.1 Linear Axes

The basic graph set up routine is GRAF as described previously. GRAF uses data units to define the origin, step size and ending values for the axes. There are two other routines for setting up linear axes which use inches instead of data units for one of the axes. They are:

CALL XGRAF (XORIG,XSTP,XMAX,YORIG,YSTEP)

CALL YGRAF (XORIG,XSTEP,YORIG,YSTP,YMAX)

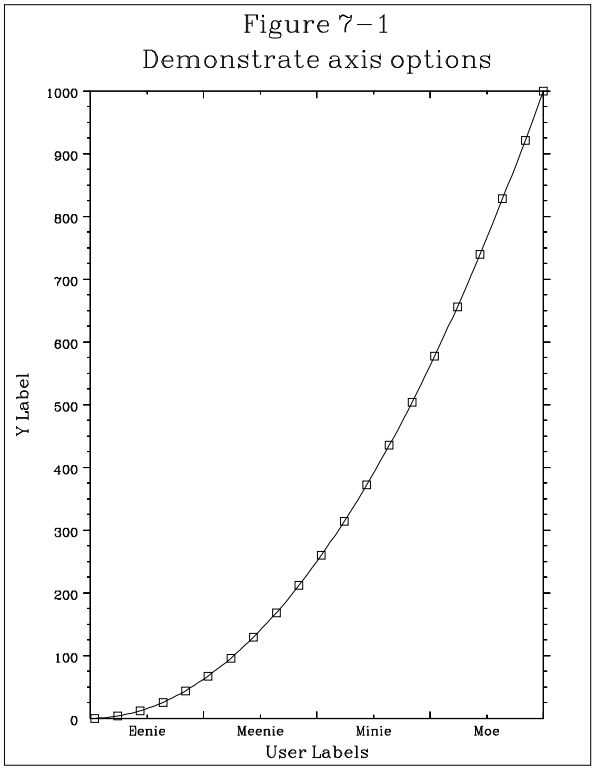
where XSTEP and YSTEP are given in inches and the other arguments are the same as for the GRAF routine. There is also a routine for setting up a plot with axes labelled with a user specified text string instead of with numbers. These routines are:

CALL XLABGR (LXARAY,IXWORD,IXTOT,YORIG,YSTP,YMAX)

CALL YLABGR (XORIG,XSTP,XMAX,LYARAY,IYWORD,IYTOT)

In XLABGR and YLABGR, IXTOT and IYTOT are the number of labels for the axis. IXWORD and IYWORD are intended to be the length of the labels, but these are no longer used. An example of the use of XLABGR and some of the other routines in this section is shown in Example 7-1.

```
PROGRAM GPE5      ! Example 7-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/
CHARACTER*6 XLABEL(4)/'Eenie ','Meenie', 'Minie ','Moe   '/

CALL GSTART               ! Nominate Device
CALL PAGE(8.5,11.0)       ! Specify page size
CALL ORIENT('PORTRAIT') ! Force portrait orientation
CALL BLOWUP(0.4)        ! Scale down for manual
CALL AREA2D(6.5,9.0)    ! Define Subplot area
CALL DUPLX              ! Choose default font
CALL HEIGHT(0.25)       ! Use larger characters for small plot
CALL XNAME('User Labels$',100)        ! Define X Axis label
CALL YNAME('Y Label$',100)   ! Define Y Axis label
CALL HEADIN('Figure 7-1$',100,1.5,2)
CALL HEADIN('Demonstrate axis options$',100,1.5,2)
CALL XAXCTR       ! Center labels on X axis
CALL XREVTK       ! Reverse X Axis tick marks
CALL XTICKS(2)    ! Two tick marks per X division
CALL YNMADJ('NONE')
CALL YAXANG(0.0)  ! Use horizontal Y Axis labels
CALL YINTAX       ! Use Integers to label Y Axis
CALL YTICKS(4)    ! Four ticks per y division
CALL THKFRM(2.0)  ! Make frame twice normal thickness
CALL FRAME
CALL XLABGR(XLABEL,100,4,YORIG,DY,YMAX)
!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*0.04
  Y(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)
CALL ENDPL(0)
CALL GSTOP

CALL EXIT
END
```



7.2 Producing Grids and/or Tick Marks

Tick marks are normally generated at each major axis division. To produce additional tick marks, make the following calls:

CALL XTICKS (ITICKS)

CALL YTICKS (ITICKS)

where ITICKS is the total number of tick marks per axis step. These routines are demonstrated in Example Plot 2 in the Appendix.

To produce a grid on the plot, use one of the following subroutine calls:

CALL GRID (NXGRID,NYGRID)

CALL DOTGRID (NXGRID,NYGRID)

In these grid routines NXGRID and NYGRID are the numbers of grid lines per division on the x and y axes. DOTGRID is a special grid which produces the dots each one-fifth of an axis division. This is slow, but can be used for interpolation. A dotted grid can also be produced by GRID if the line type is changed to dotted before calling GRID, but the dots will not be synchronous with the axis divisions.

7.3 Logarithmic Axes

Logarithmic axes can be used instead of linear axes by replacing the call to GRAF with one of the following calls:

CALL XLOG (XORIG,XCYCLE,YORIG,YSTEP)

CALL YLOG (XORIG,XSTEP,YORIG,YCYCLE)

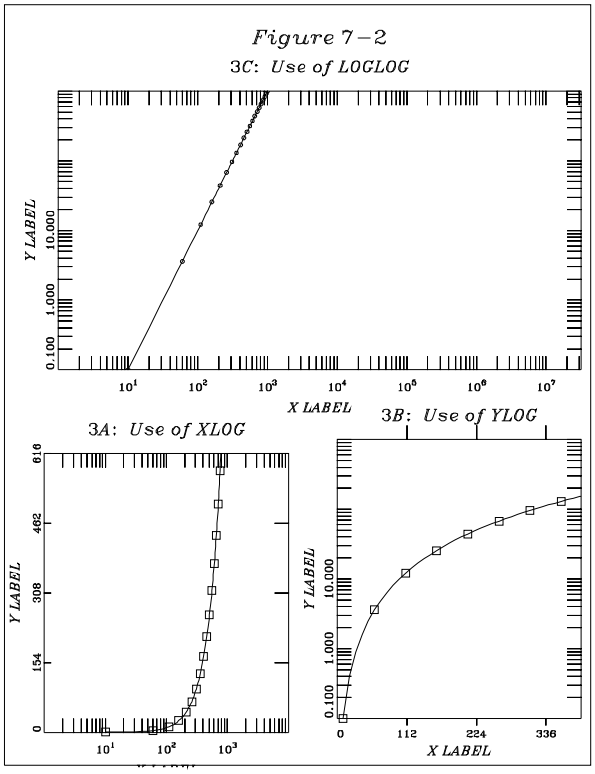
CALL LOGLOG (XORIG,XCYCLE,YORIG,YCYCLE)

CALL XLGAXS (XORIG,XCYCLE,XAXIS,LXNAME,IXNAME,XPOS,YPOS)

CALL YLGAXS (YORIG,YCYCLE,YAXIS,LYNAME,IYNAME,XPOS,YPOS)

XLOG and YLOG give semilog plots and LOGLOG does loglog plots. XLGAXS and YLGAXS are used to draw secondary log axes. In these routines XCYCLE and YCYCLE are the cycle lengths in axis units (Inches/cycle), and the other parameters are as in GRAF. Example 7-2 demonstrates the different logarithmic axes.

```
PROGRAM GPE6      ! Example 7-2
                  ! Demonstrate Log graphics routine usage
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/

!  Compute a quadratic curve
DO I=1,100
  X(I) = I*10.0
  Y(I) = 0.1*I*I
END DO
IDX = (XMAX-XORIG)/9.0 + 1
DX = IDX
!                 Draw First Plot using XLOG
CALL GSTART
CALL PAGE(8.5,11.0)          ! Specify page size
CALL ORIENT('PORTRAIT') ! Use Portrait orientation
CALL BLOWUP(0.4)  ! Scale down for manual
CALL PHYSOR(0.6,0.5)       ! PHYSOR must be called before AREA2D
CALL AREA2D(3.5,4.0)         ! Define Subplot area
CALL FONT('HELVETICA','SLANT')
CALL HEIGHT(0.25)  !Scale up text for manual
CALL XNAME('X LABEL$',100)   ! Define X Axis label
CALL YNAME('Y LABEL$',100)   ! Define Y Axis label
CALL HEADIN('3A:  Use of XLOG$',100,1.5,1)
XORIG = 1.0
XCYCLE = 1.5      !Inches/cycle
IDY = (YMAX-YORIG)/6.5 + 1
DY = IDY
CALL YINTAX
CALL YREVTK
CALL XLOG(XORIG,XCYCLE,YORIG,DY)
CALL CURVE(X,Y,100,5)
CALL ENDGR

!                 Draw Second Plot using YLOG
CALL OREL(4.2,0.2)
CALL AREA2D(3.5,4.0)         ! Define Subplot area
CALL HEADIN('3B:  Use of YLOG$',100,1.5,1)
YORIG = 0.1
YCYCLE = 1.0 ! Inches/cycle
XORIG = 0.0
CALL XINTAX
CALL YLOG(XORIG,DX,YORIG,YCYCLE)
CALL CURVE(X,Y,100,5)

!                 Draw Third Plot using LOGLOG
CALL OREL(-4.0,5.0)
CALL AREA2D(7.5,4.0)         ! Define Subplot area
CALL HEADIN('Figure 7-2$',100,2.0,2)
CALL HEADIN('3C:  Use of LOGLOG$',100,1.5,2)
YORIG = 0.1
YCYCLE = 1.0 ! Inches/cycle
XORIG = 1.0
XCYCLE = 1.0 ! Inches/cycle
CALL LOGLOG(XORIG,XCYCLE,YORIG,YCYCLE)
CALL CURVE(X,Y,100,5)
CALL ENDPL(0)
CALL GSTOP
CALL EXIT
END
```



8. Drawing Curves

The basic routine for drawing curves in GPLOT is CURVE, as indicated in the basic plot routines section. This routine will draw curves and/or place markers at data points.

8.1 Error Bars

It is often desirable to put error bars on the data points being plotted. Three calls have been provided in GPLOT which will draw error bars for the x or y uncertainty or both. Use one of the following calls in place of the call to CURVE and supply the array(s) of error values to be used. The calls are:

CALL EYCRV (XARRAY,YARRAY,NPTS,IMARK,EYARY)

CALL EXCRV (XARRAY,YARRAY,NPTS,IMARK,EXARY)

CALL EXYCRV (XARRAY,YARRAY,NPTS,IMARK,EXARY,EYARY)

The default error bar will be a split line which is not drawn over the marker, but this can be changed to a split line with bars on the end by the call:

CALL EMARK (2)

To use unsplit lines, CALL EMARK(3), or unsplit lines with bars on the end, CALL EMARK(4). To change back to the default error bar marker, CALL EMARK(1). Example 8-1 shows a sample plot which uses the error bar routines. It also demonstrates the creation of a legend which is discussed later.

8.2 Line Types

GPLOT supports the following calls to set the line-type to be used in generating curves:

CALL SOLID

CALL DOT

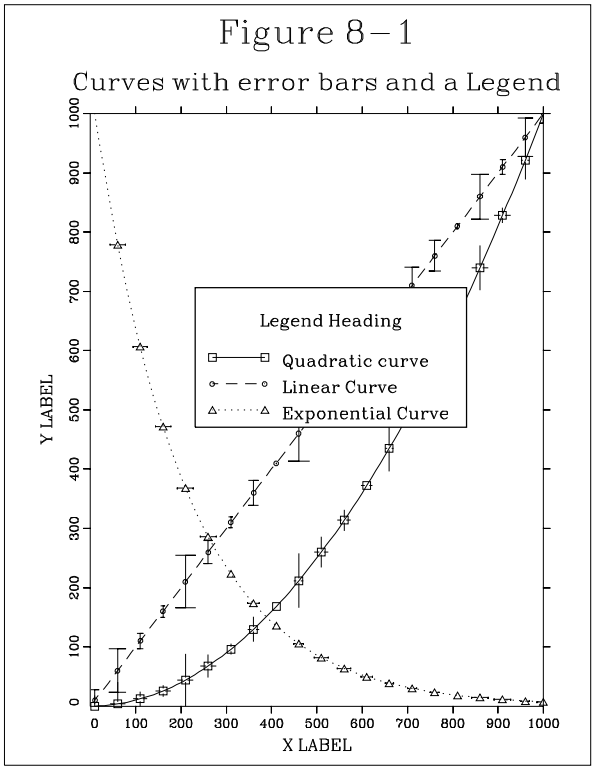
CALL DASH

CALL CHNDOT

CALL CHNDSH (DEC GKS extension not supported prior to DEC GKS 4.0)

DOT and DASH are also demonstrated in Example 8-1.

```
PROGRAM GPE7      ! Example 8-1
DIMENSION X(100),Y(100),Y2(100),Y3(100),EY(100),EX(100)
INTEGER IPKRAY(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/
CALL GSTART                   ! Nominate graphics device
CALL ORIENT('PORTRAIT')       ! Draw in portrait orientation
CALL BLOWUP(0.4)
CALL PAGE(8.5,11.0)           ! Define Page size/aspect ratio
CALL FONT('AvantGarde','NORMAL')! Select Text Font
CALL HEIGHT(0.25)
CALL AREA2D(6.5,8.5)          ! Define Plot Area
CALL XNAME('X LABEL$',100)    ! Define X Axis label
CALL YNAME('Y LABEL$',100)    ! Define Y Axis label
CALL HEADIN('Figure 8-1$',100,2.,2)
CALL HEADIN('Curves with error bars and a Legend$',100,1.5,2)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
ISEED = 65535     ! Initialize seed to large odd number
DO I=1,100                    ! Compute three curves
  X(I) = I*10.0
  Y(I) = 0.1*I*I
  Y2(I) = X(I)
  ARG = 0.05*(I-1)
  Y3(I) = 1000.*EXP(-ARG)
  EX(I) = 0.02*XMAX*RAN(ISEED)
  EY(I) = 0.05*YMAX*RAN(ISEED)
END DO
CALL LEGNAM ('Legend Heading$',100)     ! Define legend heading
CALL LINES('Quadratic curve$',IPKRAY,1 ! Define first curve label
CALL EMARK(1)
CALL EXYCRV(X,Y,100,5,EX,EY)        ! Draw first curve
CALL LINES('Linear Curve$',IPKRAY,2) ! Define second curve label
CALL DASH                     ! Use dashed line for second curve
CALL NEWCLR('BLUE')           ! Make second  curve blue
CALL EMARK(4)
CALL EYCRV(X,Y2,100,5,EY)       ! Draw second curve with error bars
CALL LINES('Exponential Curve$',IPKRAY,3) ! Define third curve label
CALL DOT                      ! Use dotted line for third curve
CALL NEWCLR('RED')            ! Make third curve red
CALL EMARK(2)
CALL EXCRV(X,Y3,100,5,EX)       ! Draw third curve
CALL LEGEND(IPKRAY,3,1.5,4.0)
CALL ENDPL(0)                 ! Terminate this plot
CALL GSTOP
CALL EXIT
END
```



The thickness of the lines drawn with the curve routines will be the default value for the graphics device unless they have been thickened by a call to THKCRV. The call is:

CALL THKCRV (THKNSS)

where if THKNSS > 1.0 it is a factor which multiplies the nominal line width. If THKNSS < 1.0 it is approximately the line thickness in inches, although this is somewhat device dependent. THKCRV does not affect the line thickness of lines drawn by other routines.

8.3 Colors

When plotting on a color device, it is possible to choose colors by name using the call:

CALL NEWCLR ('color')

Color names which are recognized include FOREGROUND, BACKGROUND, RED, GREEN, BLUE, CYAN, MAGENTA, YELLOW, BLACK, and WHITE. It will also recognize other colors such as BROWN, GREY, GRAY, PURPLE, AQUA, and PINK for certain devices. The color set by a call to NEWCLR will affect text, markers, and filled areas as well as lines. Since colors are device dependent they will appear differently on different devices, and some trial and error will probably be necessary.

Examples 8-1 and 8-2 include calls to NEWCLR to change the color of some curves. These have no effect on a monochrome device, unless an argument of 'BACKGROUND' has been used. Some devices, such as Postscript printers and VAXstations, permit writing in the background color as well as

the foreground color, allowing you to erase an area with the background color before writing over it in another color.

For more control over colors, you can use the GKS call to set the color representation (GSCR) together with the calls to set color index for the text, polyline, polymarker, and fill area (GSTXCI, GSPLCI, GSPMCI, GSFACI). See the GKS documentation for more details of these calls. You should be aware, however that some GPLOT routines call these GKS color routines with a color index stored in common.

8.4 Data Markers

Thirty two different data markers are supported by GPLOT for marking data points (1-24, 51-54, 61-64), with most of the first 18 being the same as those produced by DISSPLA. The exceptions are symbols 11, 14, and 17. Symbols 11 and 14 appear slightly different because triangles are centered on the center of mass in GPLOT instead of at the half height of the triangle as in DISSPLA. GPLOT uses a different symbol for marker 17 because DISSPLA duplicates markers 17 and 0. Markers 51-54 are used for vertical error bars and markers 61-64 are used for horizontal error bars.

The symbols automatically increment when multiple curves are drawn on the same plot, but specific markers may be selected by the call

CALL MARKER (n)

The symbols are plotted at a default size unless a marker scaling factor is applied by the call:

CALL SCLPIC (PICFAC)

where PICFAC is the marker height scale factor. Markers 51-54 and 61-64 have their size controlled by variable EHITE. When used as error bars, the marker height is set by the CURVE routine to the error value of each data point. These error bar markers can also be used as data values by calling MARKER to select that marker type. For instance, to use vertical bars as data markers, precede the call to CURVE by the calls:

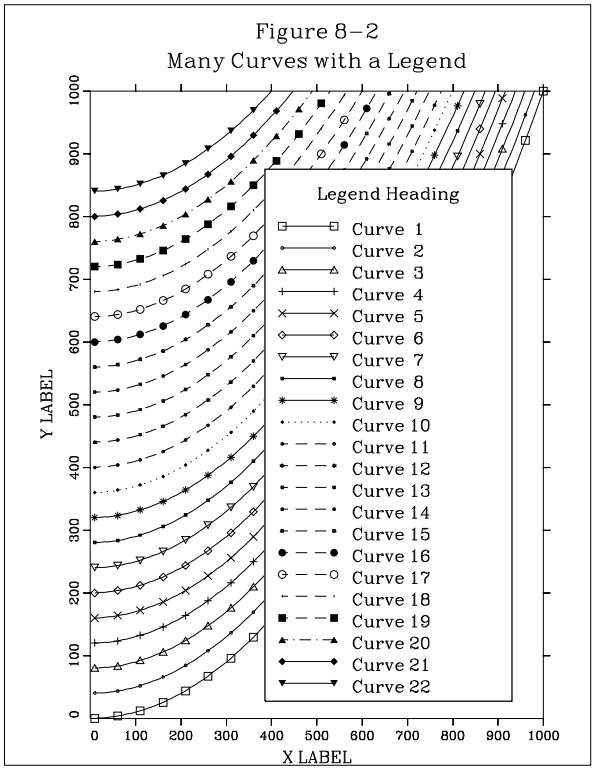
CALL MARKER(53)
CALL EHEIGHT(EHITE)

where EHITE is the desired height of the marker. When using error bar type markers, the marker height is controlled by calling EHEIGHT(EHITE).

In some cases different markers will be different sizes because some are drawn by GPLOT subroutines and others are furnished by DEC VAX GKS. Markers furnished by DEC VAX GKS will also be different sizes on different devices because marker size scaling is in terms of the default marker size for the device. If this causes a problem, you can select the markers drawn by GPLOT subroutines. Markers 2,8,10-15, and 22-24 are generated by GPLOT. The other markers are generated by DEC VAX GKS and the size will vary from device to device. There is also a bug in DEC VAX GKS in marker clipping for some devices, so errors may result when you attempt to draw one of those markers outside the clipping rectangle.

The first 22 available markers are illustrated in Example 8-2.

```
PROGRAM GPE8      ! Example 8-2
DIMENSION X(100),Y(100)
INTEGER IPKRAY(500)
CHARACTER LABEL*20
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/
CALL GSTART
CALL PAGE(8.5,11.0)           ! Specify page size
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL AREA2D(6.5,9.0)          ! Define Subplot area
CALL FONT('TIMES','NORMAL')
CALL HEIGHT(0.20)
CALL XNAME('X LABEL$',100)    ! Define X Axis label
CALL YNAME('Y LABEL$',100)    ! Define Y Axis label
CALL HEADIN('Figure 8-2$',100,1.25,2)
CALL HEADIN('Many Curves with a Legend$',100,1.25,2)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
CALL LEGNAM ('Legend Heading$',100)     ! Define legend heading
!  Compute and plot a quadratic curve
NLINES=22
DO IC = 1,NLINES
  OFFSET = YMAX*0.04*(IC-1)
  WRITE(LABEL,FMT='(''Curve'', I3,''$'')') IC
  CALL LINES(LABEL,IPKRAY,IC)        ! Define curve label
  DO I=1,100
    X(I) = I*10.0
    Y(I) = 0.1*I*I + OFFSET
  END DO
  IF(IC .EQ. 10) THEN
    CALL DOT
    CALL NEWCLR('RED')
  ELSE IF (IC .EQ. 11) THEN
    CALL DASH
    CALL NEWCLR('BLUE')
  ELSE IF (IC .EQ. 20) THEN
    CALL CHNDOT
    CALL NEWCLR('GREEN')
  ELSE IF (IC .EQ. 21) THEN
    CALL SOLID
    CALL NEWCLR('FOREGROUND')
  END IF
  CALL CURVE(X,Y,100,5)
END DO
CALL LINESP(1.3)  ! Reduce spacing between lines
CALL LEGEND(IPKRAY,NLINES,4.5,0.01)
CALL ENDPL(0)     ! If argument = 0, no plot summary is written on
plot
CALL GSTOP

CALL EXIT
END
```



8.5 Interpolation Routines

A number of routines have been included which allow interpolation before drawing the curve through the data points. The default is to draw straight lines connecting the points supplied to the curve routine. If the call to the curve routine is preceded by a call to an interpolation routine, a different curve will be drawn for each of the other methods. The interpolation routines which can be called include:

CALL LINEAR

CALL BARS (BARWTH)

CALL PARA3

CALL PARA5

CALL POLY3

CALL POLY5

CALL PSPLIN

CALL RASPLN (TENSN)

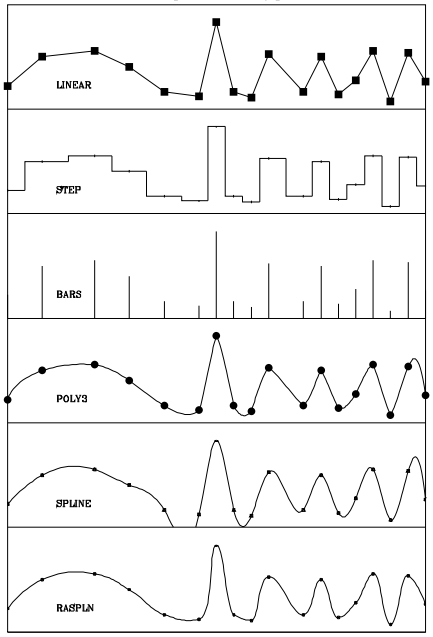
CALL SPLINE

CALL STEP

Here BARWTH is the ratio of the bar width to data interval, and TENSN is a parameter that determines the degree of smoothing by the RASPLN routine. A sample of the effects of some of these interpolation routines is shown in Example 8-3.

```
!     Program to test interpolation routines
PROGRAM GPE9  ! Example 8-3
DIMENSION X(18),Y(18)
!  Initialize data for plots
DATA X /0.00,0.50,1.25,1.75,2.25,2.75,3.00,3.25,3.50,
1      3.75,4.25,4.50,4.75,5.00,5.25,5.50,5.75,6.00/
DATA Y /0.40,0.90,1.00,0.73,0.30,0.22,1.50,0.30,0.20,
1      0.95,0.30,0.90,0.25,0.50,1.00,0.13,0.97,0.48/

CALL GSTART
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL FONT('HELVETICA','NORMAL')
!  Establish plot area (Use default 8.5" X 11" page size)
CALL AREA2D (6.,9.)
CALL HEIGHT (.25)
XL=XMESS ('Interpolation Types$',100)
CALL MESSAG ('Interpolation Types$',100,3.-XL/2.,9.125)
XL = XMESS('GPLOT Example 6$',100)
CALL MESSAG('GPLOT Example 6$',100,3.-XL/2.,9.5)
CALL RESET ('HEIGHT')
!  Turn on clipping at boundary of subplot area
CALL GRACE (0.0)
CALL GRAF (0.,1.,6.,0.,1.,1.5)
!  Draw a single line around plot area
CALL GRID (0,0)
CALL ENDGR(0)
!  Loop to draw all 6 interpolations
DO 50 I=1,6
!  Shift up for next plot
  IF(I.NE.1) CALL OREL (0.,1.5)
  CALL AREA2D (6.,9.)            ! Redefine subplot area
  CALL GRAF (0.,1.,6.,0.,1.,1.5) ! Redefine axes data limits
!  Draw a secondary y axis (This restricts height of plot also)
  CALL YGRAXS (0.,.5,1.5,1.25,' ',0,0.,0.)
!  Draw horizontal line on each graph
  CALL RLVEC (0.,0.,6.,0.,0)
!  Set interpolation mode for each plot
  IF (I.EQ.1) CALL RASPLN(3.)
  IF (I.EQ.2) CALL SPLINE
  IF (I.EQ.3) CALL POLY3
  IF (I.EQ.4) CALL BARS(.25)
  IF (I.EQ.5) CALL STEP
  IF (I.EQ.6) CALL LINEAR
!  Draw interpolated curve
  CALL MARKER (I+12)     ! Set marker symbol
  IF (I.NE.4) CALL CURVE (X,Y,18,1)
  IF (I.EQ.4) CALL CURVE (X,Y,18,0)    ! (No markers on bar chart)
!  Write interpolation type
  IF (I.EQ.1) CALL MESSAG ('RASPLN',,.7,.3)
  IF (I.EQ.2) CALL MESSAG ('SPLINE$',100,.7,.3)
  IF (I.EQ.3) CALL MESSAG ('POLY3$',100,.7,.3)
  IF (I.EQ.4) CALL MESSAG ('BARS$',100,.7,.3)
  IF (I.EQ.5) CALL MESSAG ('STEP$',100,.7,.3)
  IF (I.EQ.6) CALL MESSAG ('LINEAR$',100,.7,.3)
  CALL ENDGR(0)
50   CONTINUE
CALL ENDPL (0)

CALL GSTOP
CALL EXIT
END
```

Figure 8–3
Interpolation Types



9. Legends

Graphs may be labelled with legend routines similar to those used by DISSPLA except that the GPLOT routines keep track of line color and type as well as symbol type for use in the legend. A legend may be produced as follows:

```
PARAMETER NWORDS=100
INTEGER IPKRAY
      .
      .
      .

CALL LEGNAM ('Legend Heading$',100)
      .
      .
      .
CALL LINES ('Curve 1 Label$',IPKRAY,1)
      .
      .
      .
CALL LINES ('Curve 2 Label$',IPKRAY,2)
      .
      .
      .
CALL LEGEND (IPKRAY,NCURVE,XPOS,YPOS)
```

Here LMAX is the maximum number of characters in a legend line, NCURVE is the total number of curves, and the last argument in the call to LINES is ICURVE, the curve index. You may want to position the legend where it will not interfere with other curves and text because GPLOT does not always blank the plot as DISSPLA does and the graph can overwrite the legend. For devices that can write in the background color the legend area will be cleared before the legend is written. Examples showing how to produce a legend on a plot were included in Example Plots 8-1 and 8-2. There are several routines which can be used to modify the legend. they have the following calls:

```
CALL ALNLEG(XALIGN,YALIGN)

CALL DELLEG(IENTRY)

CALL LINESP(YRATIO)
```

In the routine ALNLEG, XALIGN and YALIGN represent the alignment position with respect to the legend origin. If XALIGN, YALIGN are 0.0 the legend will be aligned with respect to the lower left corner of the legend. If they are both 1.0 it will be aligned with respect to the upper right corner of the legend.

In the subroutine DELLEG, IENTRY is the entry number of a legend line that is to be omitted from the legend.

In the routine LINESP, YRATIO is the vertical line spacing ratio for legend lines. The default value is 1.5.

10. Vectors With and Without Arrowheads

If you wish to draw vectors without using the CURVE calls, GPLOT provides routines to draw vectors in either axes units or data units. They may also include various types of arrow heads if desired. The calls for these routines are:

CALL VECTOR (XFROM,YFROM,XTO,YTO,IVEC)

CALL RLVEC (XFROM,YFROM,XTO,YTO,IVEC)

IVEC is a four digit integer specifying the arrowhead to be used. The first digit gives the ratio of width to length and may have values from 0 to 5. The second digit gives the arrowhead size from 0 to 6. The third digit specifies the type of arrowhead:

0 -> Solid
1 -> White
2 -> Open
3 -> Closed

The last digit specifies the location of the arrowheads:

0 -> None
1 -> at (XTO,YTO)
2 -> at both ends, pointing out
3 -> at both ends, pointing to (XTO,YTO)

Vectors can also be drawn using the following routines to define a starting point and subsequent points to be connected.

CALL STRTPT (X0,Y0)

CALL CONNPT (XI,YI)

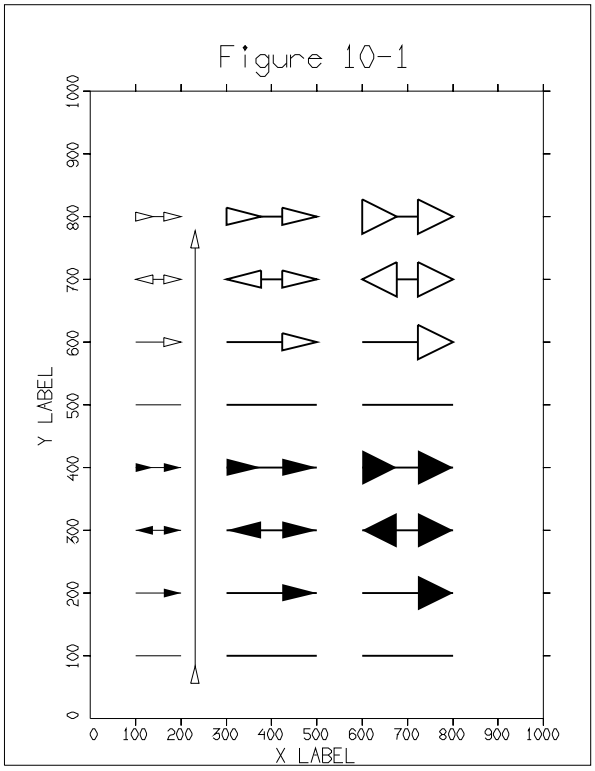
Vectors will be drawn using the nominal line width for the graphics device unless THKVEC has been called. To change the line width of lines drawn with vector routines, use the call:

CALL THKVEC (THKNSS)

where the line width used to draw the vector will be THKNSS times the nominal linewidth if THKNSS > 1.0 or will be approximately THKNSS inches thick if THKNSS < 1.0.

Example 10-1 shows how to use VECTOR, RLVEC, and THKVEC. Several different values of IVEC are used to illustrate the differences in arrowheads produced.

```
PROGRAM GPE10     ! Example 10-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/

CALL GSTART       ! Nominate Device
CALL PAGE(8.5,11.0)          ! Specify page size
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL AREA2D(6.5,9.0)         ! Define Subplot area
CALL XNAME('X LABEL$',100)   ! Define X Axis label
CALL YNAME('Y LABEL$',100)   ! Define Y Axis label
CALL HEIGHT(0.25)
CALL HEADIN('Figure 10-1$',100,1.5,1)
CALL GRAF (XORIG,DX,XMAX,YORIG,DY,YMAX)
!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*10.0
  Y(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)
CALL RLVEC(100.,100.,200.,100.,0)
CALL RLVEC(100.,200.,200.,200.,1)
CALL RLVEC(100.,300.,200.,300.,2)
CALL RLVEC(100.,400.,200.,400.,3)
CALL RLVEC(100.,500.,200.,500.,10)
CALL RLVEC(100.,600.,200.,600.,11)
CALL RLVEC(100.,700.,200.,700.,12)
CALL RLVEC(100.,800.,200.,800.,13)

CALL VECTOR(1.5,0.5,1.5,7.0,13)

CALL THKVEC(3.0)
CALL RLVEC(300.,100.,500.,100.,100)
CALL RLVEC(300.,200.,500.,200.,101)
CALL RLVEC(300.,300.,500.,300.,102)
CALL RLVEC(300.,400.,500.,400.,103)
CALL RLVEC(300.,500.,500.,500.,110)
CALL RLVEC(300.,600.,500.,600.,111)
CALL RLVEC(300.,700.,500.,700.,112)
CALL RLVEC(300.,800.,500.,800.,113)
CALL RLVEC(600.,100.,800.,100.,1100)
CALL RLVEC(600.,200.,800.,200.,1101)
CALL RLVEC(600.,300.,800.,300.,1102)
CALL RLVEC(600.,400.,800.,400.,1103)
CALL RLVEC(600.,500.,800.,500.,1110)
CALL RLVEC(600.,600.,800.,600.,1111)
CALL RLVEC(600.,700.,800.,700.,1112)
CALL RLVEC(600.,800.,800.,800.,1113)

CALL ENDPL(0)
CALL GSTOP

CALL EXIT
      END
```



11. Graphic Primitives

DEC VAX-GKS provides graphic primitives for drawing arcs, circles, ellipses, and rectangles. It also provides primitives for filling these figures. These have been used by GPLOT to produce easy to use routines for drawing graphics primitives. Some of these correspond in name and calling parameters with DISSPLA routines for blanking areas, however the GPLOT routines only draw the figures, and don't do blanking in most cases. The routine CLRREC can be used to erase a rectangular area on devices which provide this capability (i.e. the HDS graphics terminal, VAXstation, DECWindows terminals, Postscript printers). The filling of these primitives is dependent on the state of the GKS Fill Area Constants at the time of the call. For some devices an area can be erased by setting the Fill Area Interior Style to solid and the Fill Area Color Index to the background color. See Example 11-1 shows how the filling is controlled by these variables.

CALL BLREC (XORG,YORG,WIDE,HIGH,FRM)

CALL BLTREC (XORG,YORG,WIDE,HIGH,ANG,FRM)

CALL CIRCLE (XORG,YORG,RADIUS,FRM)

CALL ELLIPS (XORG,YORG,RADX,RADY,ANG,FRM)

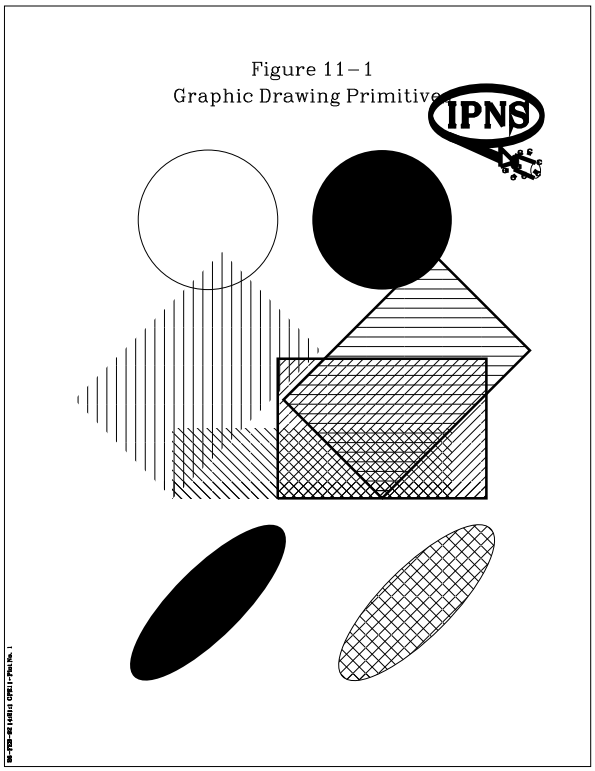
CALL BLPOLY (XARAY,YARAY,NPTS,FRM)

CALL ARC (XORG,YORG,RADIUS,THETA1,THETA2,ICLS,FRM)

CALL ELPARC (XORG,YORG,RAD1,RAD2,ANG,THETA1,THETA2,ICLS,FRM)

For rectangles, XORG and YORG are the coordinates of the lower left corner of the rectangle. For circles and arcs, XORG and YORG are the coordinates of the center. ICLS is one of the following character strings: 'NONE', 'SLICE', or 'CHORD'. Example plots 11-1 and 11-2 demonstrate the use of these graphic primitive routines. [4]

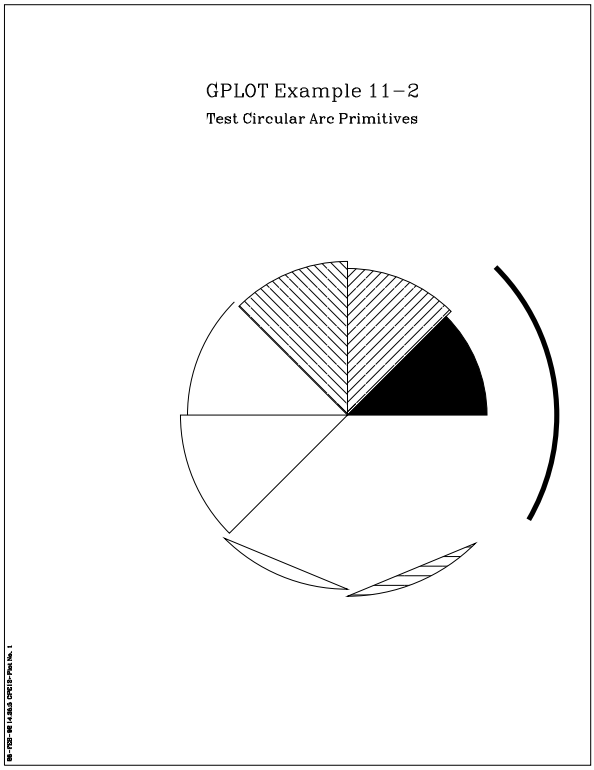
```
!
! Example program to demonstrate Graphic Drawing Primitives
!
PROGRAM GPE11  ! Example 11-1
INCLUDE 'SYS$LIBRARY:GKSDEFS.BND'
CALL GSTART
CALL ORIENT('PORTRAIT') ! This must come before AREA2D
CALL BLOWUP(0.4)
CALL AREA2D(7.0,8.5)
CALL DUPLX

CALL HEADIN('Figure 11-1$',100,2.0,2)
CALL HEADIN('Graphic Drawing Primitives$',100,2.0,2)

CALL LOGO(6.0,8.5,2.0)                ! Draw IPNS Logo

CALL GSFAIS(GSOLID)                   ! Set filling solid
CALL CIRCLE (4.5,7.0,1.0,0.0)          ! Draw filled circle
CALL ELLIPS(2.0,1.5,1.5,0.5,45.,2.0)  ! Draw ellipse at 45 deg

CALL GSFAIS(GHATCH)                  ! Set filling hatched
CALL ELLIPS(5.0,1.5,1.5,0.5,45.,0.0)  ! Draw filled ellipse
CALL NEWCLR('BLUE')
CALL BLREC (3.0,3.0,3.0,2.0,4.0)      ! Draw rectangle
CALL BLREC (1.5,3.0,4.0,1.0,0.0)      ! Draw rectangle w/o border
!
! (Note that fill pattern increments automatically)
!
CALL BLTREC(4.5,3.0,3.0,2.0,45.0,4.0) ! Draw tilted rectangle
CALL BLTREC(1.5,3.0,3.0,2.0,45.0,0.0) ! Same, w/o border
CALL NEWCLR('GREEN')                  ! Change colors
CALL BLCIR (2.0,7.0,1.0,1.0)          ! Draw open circle
CALL GSFAIS(GSOLID)                   ! Set filling solid again

CALL ENDPL(1)
CALL GSTOP

CALL EXIT
END
```



---

4. There is a bug in the DEC GKS 4.1 Postscript driver which causes the ARC and ELPARC routines to work incorrectly on Postscript printers. This has been fixed in DEC GKS 4.2, so be sure you are not running an older version if you want to use these routines.

```
PROGRAM GPE12     ! Example 11-2, test circular arcs
INCLUDE 'SYS$LIBRARY:GKSDEFS.BND'
REAL*4 XORG,YORG,RADIUS,THETA1,THETA2,FRM
INTEGER*4 WKTYPE,MASK
DATA MASK/'FF'X/
CALL GSTART       ! NOMINATE DEVICE
CALL PAGE(8.5,11.0)
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL AREA2D(7.0,8.0)
CALL DUPLX
CALL HEADIN('GPLOT Example 11-2$',100,2.0,2)
CALL HEADIN('Test Circular Arc Primitives$',100,1.5,2)
XORG = 4.0
YORG = 4.0
RADIUS = 2.0
THETA1 = 0.0
THETA2 = 45.0
FRM = 0.0   ! FRM = 0 for filled arcs
CALL GSFAIS(GSOLID)     ! Set to solid fill
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'SLICE')
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
CALL GSFAIS(GHATCH)     ! Change to hatch fill
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'SLICE')
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
!  Note that hatch pattern increments automatically
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'SLICE')
FRM = 1.0               ! FRM > 0.0 for unfilled arc
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'NONE')
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
CALL GSFAIS(GHOLLO)
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'SLICE')
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'CHORD')
THETA1 = THETA2
THETA2 = THETA1 + 45.0
RADIUS = RADIUS + 0.1
FRM = 0.0
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'CHORD')
THETA1 = -30.
THETA2 = THETA1 + 75.0
RADIUS = 3.0
FRM = 8.0                  ! Line width times 8
CALL ARC(XORG,YORG,RADIUS,THETA1,THETA2,FRM,'NONE')
CALL ENDPL(1)
CALL GSTOP
CALL EXIT
END
```



12. Bar Graphs

GPLOT contains routines to emulate the DISSPLA routines for drawing vertical and horizontal bars. These subroutine calls are:

CALL VBARS (XARAY,Y1ARAY,Y2ARAY,NPTS)

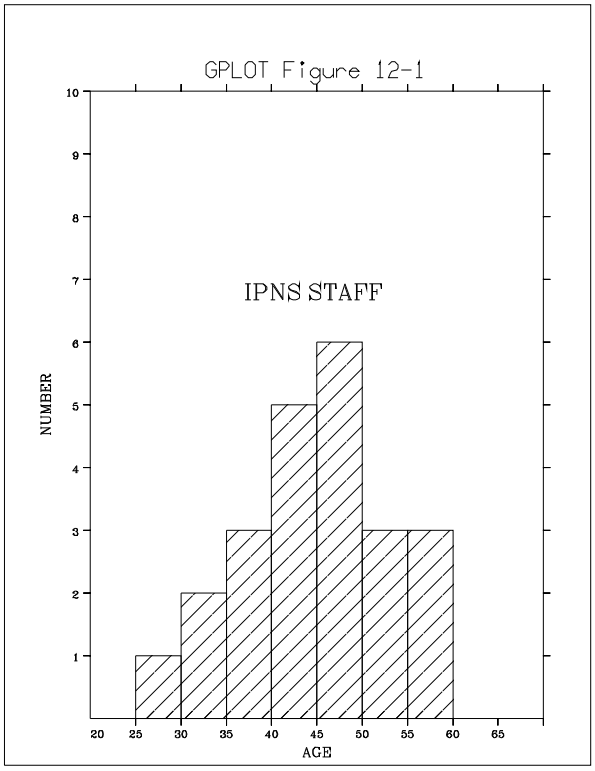
CALL HBARS (X1ARAY,X2ARAY,YARAY,NPTS)

CALL BARPAT (ISTYLE)

CALL BARWID (BARWTH)

These work as the DISSPLA routines except GPLOT patterns are based on the GKS standard patterns and there are only nine different patterns. Example 12-1 demonstrates the use of these routines.

```
PROGRAM GPE13  ! Example 12-1
!               Program to plot age bar graph
graphs
REAL*4 NAGE(10),AGE(10)
DATA NAGE/0.,1.,2.,3.,5.,6.,3.,3.,0.,0./
DATA AGE/20.,25.,30.,35.,40.,45.,50.,55.,60.,65./

CALL GSTART
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL PAGE(8.5,11.0)
CALL AREA2D(6.5,9.)

CALL HEADIN('GPLOT Example 12-1$',100,1.5,1)
CALL DUPLX
CALL HEIGHT(0.21)
CALL YAXEND('NOFIRST')
CALL XAXEND('NOENDS')
CALL XNAME('AGE$',100)
CALL YNAME('NUMBER$',100)
CALL YAXANG(0.0)
CALL YINTAX
CALL XINTAX

CALL GRAF(20.,5.,70.,0.0,1.,10.)
CALL BARWID(-1.0)
DO I=1,10
  AGE(I) = AGE(I) + 2.5
END DO
CALL VBARS(AGE,'BASE',NAGE,10)
CALL HEIGHT(0.35)
CALL MESSAG('IPNS STAFF$',100,3.5,5.0)
CALL ENDPL(0)
CALL GSTOP

CALL EXIT
END
```



13. Contouring Routines

GPLOT contains several routines for generating contour plots. The relevant routines are **CONMAK**, **CONLIN**, **CONTUR**, **CONMIN, CONANG,** and **CONDIG**. CONMAK makes the contour lines and saves them in its internal data structures. CONLIN is used to set up various line styles for the contour lines of different levels. CONTUR actually plots the contour lines made by CONMAK according to the line styles established by CONLIN. CONMIN and CONDIG are used to modify the minimum distance between contour labels and to specify the format for contour labels. CONANG specifies the maximum angle in degrees for contour line curvature if the contour is to contain a label. The call

CALL CONMAK (w, nx, ny, zIncrement)

builds contour lines for a rectangular array of data values, w(,). It is assumed that w is dimensioned as w(nx,ny). w(i,j) should contain the data value corresponding to the point (xi, yj) of a rectangular grid of points (xi,yj), i = 1,...,nx, j = 1,...,ny. The argument zIncrement gives the increment between contour levels. Alternatively, zIncrement can also be the character string constant 'SCALE', in which case CONMAK will choose rounded levels between the largest and smallest values in w(,) at which to draw contour lines. In either case there will be a constant increment between contour lines, call it INCR, and contour lines will be calculated for all Z levels of the form

K*INCR, K = ...,-1, 0, 1, ...

which are between the largest and smallest values stored in w(,). The call

CALL CONLIN (LineNum, MODE, 'NOLABELS', IThick, 1)

sets up the characteristics to be used when drawing the contour lines. LineNum is the "number" of the contour line whose characteristics are to be set. The correspondence between the contour level K*INCR and LineNum is

LineNum = K MOD nLines

where nLines is the total number of line characteristics set by calls to CONLIN.
MODE may be one of

'SOLID'	--	use a solid line
'DOT'	--	use a dotted line
'DASH'	--	use a dashed line
'CHNDOT'	--	use a chain dotted line
'CHNDSH'	--	use a chain dashed line
'MYCON'	--	use line characteristics as computed by the user supplied subroutine MYCON

The first five of these values select the specified line styles. If MODE is 'MYCON', the user must provide a subroutine by that name which will set the line characteristics at run time. The use of MYCON is explained more fully below.

The argument IThick is an integer specifying a thickness multiplier for line LineNo. Specifically, if IThick = 3, a line three times thicker than the standard line width will be used.

The remaining two arguments 'NOLABELS' and 1 must be passed in as shown, but have no effect on the resulting plot. The routine

CALL CONTUR (nLines, 'NOLABELS', DrawFlag)

draws the contour lines. The argument nLines must be the total number of line characteristics set by calls to CONLIN.

The argument DrawFlag must be either of the character strings 'DRAW' or 'NODRAW'. This determines whether each contour line will be drawn or not drawn by default. 'NODRAW' is only useful in conjunction with the user supplied routine MYCON. MYCON can selectively permit lines at various levels to be drawn in case the default is set to 'NODRAW'.

The remaining argument is 'LABELS' or 'NOLABELS' to select or deselect contour labels.

If the MODE 'MYCON' was selected for a particular line characteristic, then the user MUST supply a subroutine MYCON (RArray,IArray) which will be called by CONTUR before a contour is plotted using that characteristic. In this way MYCON can change the appearance of particular contour lines. The two arguments of MYCON are arrays:

```
REAL    RArray(2)
INTEGER IArray(9)
```

Currently, the only positions in the arrays that are used are:

RArray(1) --	Contains the Z-level of the current contour line being drawn.
IArray(1) --	Contains the value of current curve thickness multiplier. This may be changed depending on the Z-level to highlight specific levels if desired.
IArray(9) --	Contains the default curve drawing flag value. 0 = 'NODRAW' 1 = 'DRAW' This may be changed to selectively omit or include various Z-levels.

When contour labelling is specified, the labels are written by default every 3.0 inches along the contour line. To change this to CLMIN, include the call

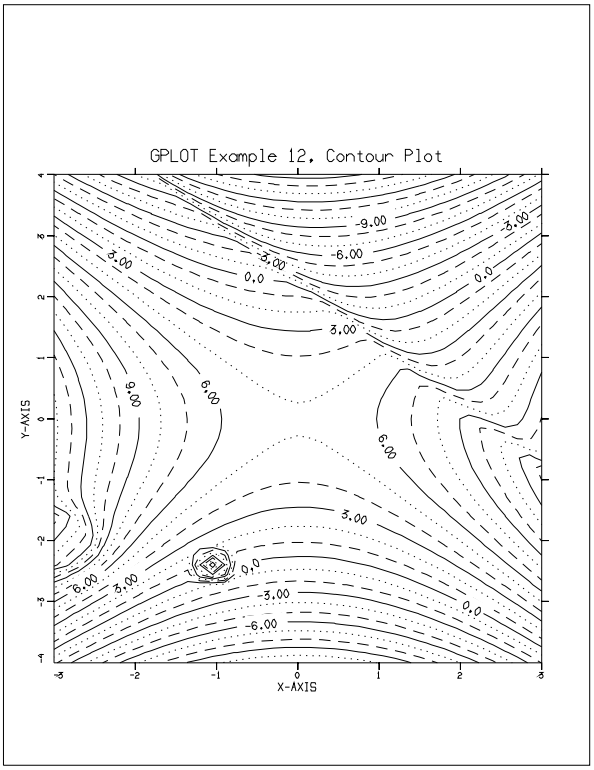
CALL CONMIN(CLMIN)

To change the number of decimal places for the labels, issue the call:

CALL CONDIG(IPLACE)

where IPLACE has the same meaning as it does in REALNO. Example 13-1 illustrates the use of the contour routines.

```
PROGRAM GP14  ! Example 13-1
Example 13-1
PARAMETER Nx = 41        ! Number of X data values (#rows)
PARAMETER Ny = 51        ! Number of Y data values (#columns)
PARAMETER xRange = 3.0   ! x Varies between -xRange and +xRange
PARAMETER yRange = 4.0   ! y Varies between -yRange and +yRange

DIMENSION DataArray(Nx,Ny) ! Array of values to contour
CALL BuildArray( DataArray, Nx, Ny, xRange, yRange )
call GSTART
CALL PAGE(11.0,8.5)
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL PHYSOR(2.0,0.5)
call area2d(7.0,7.0)
call xname('X-AXIS$',100)
call yname('Y-AXIS$',100)
call graf(-xRange, 'SCALE', xRange, -yRange, 'SCALE', yRange)
call headin('GPLOT Example 12, Contour Plot$', 100,1.5,1)
zIncrement = 1.
call CONMAK(DataArray, Nx, Ny, zIncrement)
iThick = 1
call CONLIN(0, 'SOLID' , 'LABELS', iThick, 1)
call CONLIN(1, 'DASH'  , 'NOLABELS', iThick, 1)
call CONLIN(2, 'DOT'   , 'NOLABELS', iThick, 1)
nLineTypes = 3
call conmin(3.0)
call condig(-4)   ! For scientific notation with four decimal places
call condig(2)    ! For floating point with two decimal places
call condig(0)    ! For integer
call CONTUR(nLineTypes, 'LABELS', 'DRAW')
call endpl(0)
call GSTOP

end
!    Subroutine MYCON
!    An example of dynamic modification of contour lines
!    It is called by CONTUR routine before drawing a contour line
SUBROUTINE MYCON( RArray, IArray )
DIMENSION RArray(2), IArray(9)
call reset('dot')
if ( ABS(RArray(1)) .lt. .05) then       ! level 0, so make it three
times
IArray(1) = 3                          ! as thick
endif
end

SUBROUTINE BuildArray( DataArray, Nx, Ny, xRange, yRange )
REAL DataArray( Nx, Ny )
xstep = 2.0 * xRange/(Nx-1)
ystep = 2.0 * yRange/(Ny-1)
xi = -xRange
do 10 i = 1,Nx
yj = -yRange
do 5 j = 1,Ny
  temp1 = max(xi+yj-2 , .2)
  temp2 = max( (xi+3.)**2 + (yj+2.)**2, .2)
  DataArray(i,j) = xi**2 - yj**2 +   1./temp1 + 1./temp2
  yj = yj + ystep
5       continue
xi = xi + xstep
10    continue
!  Now put in a "peak"
iPeak = Nx/3 + 1
jPeak = Ny/5 + 1

DataArray( iPeak-1 , jPeak-1 ) =  4.0
DataArray( iPeak-1 , jPeak   ) =  6.0
DataArray( iPeak-1 , jPeak+1 ) =  4.0
DataArray( iPeak   , jPeak-1 ) =  6.0
DataArray( iPeak   , jPeak   ) = 10.0
DataArray( iPeak   , jPeak+1 ) =  6.0
DataArray( iPeak+1 , jPeak-1 ) =  4.0
DataArray( iPeak+1 , jPeak   ) =  6.0
DataArray( iPeak+1 , jPeak+1 ) =  4.0

END
```



14. 3 Dimensional Plotting

GPLOT includes routines for drawing 3-Dimensional vectors, curves and surfaces. The 3-Dimensional objects are drawn in a 3D region called the "workbox" which can be viewed from various positions outside the workbox.

To use the 3-Dimensional plotting features in GPLOT, your program must first set up the page and subplot area as for 2-Dimensional plots. The 3D workbox will be projected into the subplot area specified in the call to AREA2D. The projection used is determined from the 3D workbox and various viewing parameters as described in the next two subsections. Meaningful defaults for the workbox and viewing parameters are provided and the subroutines in the next two sections may be omitted if the defaults are acceptable.

14.1 Setting the "workbox"

The 3D workbox can be specified using the routines Volm3D or Axes3D. Specifically, the call

CALL Volm3D (xSize, ySize, zSize)

would set the 3D workbox to extend from zero to xSize in the x direction, from zero to ySize in the y direction, and from zero to zSize in the z direction. These values adjust the 3D aspect ratios of the resulting plots. If all three parameters are equal, the workbox will appear cubical. If zSize is much smaller than xSize and ySize, the resulting workbox will be quite flat. The workbox establishes a coordinate system called the "absolute 3D" coordinate system.

Names can be applied to the axis using the calls

CALL X3NAME (LxName, IxName)
CALL Y3NAME (LyName, IyName)
CALL Z3NAME (LzName, IzName)

in which the parameters have the same effect as in the corresponding 2D routines XNAME, YNAME.

The above operations can be combined into one call to the routine Axes3D as shown:

```
 CALL Axes3D ( LxName, IxName, LyName, IyName, LzName,
1         IzName, xSize, ySize, zSize )
```

The default workbox size is 1.0 X 1.0 X 1.0 with blank labels. If these defaults are satisfactory, calls to the above routines may be omitted. However, if these routines are used, they must be used BEFORE Graf3D is called.

14.2 Setting the Viewing Parameters.

Once the workbox has been set up, the observer's position and the point at which the observer is looking can be specified in one of several ways. The observer's position is called the Center Of Projection or COP. The point at which the obsever is looking is called the View Reference Point or VRP.

To set the COP, use one of the calls:

CALL VuAbs (xAbs, yAbs, zAbs)
CALL VuAngl (Phi, Theta, Radius)

CALL View (x3, y3, z3)

The parameters xAbs, yAbs and zAbs specify the COP in terms of the "absolute 3D" coordinate system determined by the workbox. The parameters x3, y3 and z3 specify the COP in terms of the "relative 3D" coordinate system that will be placed on the workbox when Graf3D is called.

The parameters to VuAngl specify the COP relative to the VRP as follows:

Phi - specifies the angle in the xy-plane between the positive x axis and the projection of the ray from VRP to COP

Theta - specifies the angle between the ray from VRP to COP and the xy-plane.

Radius - specifies the distance from the VRP to the COP in absolute 3D coordinates.

NOTE: The default COP is at (-1.5, -1.5, 1.5) in absolute 3D coordinates. If a program specifies the COP, it should be outside of the 3D workbox. IF the specified COP is too close to the workbox, GPlot will move it so that the distance between it and the center of the work box is at least 1.5 times the length of the diagonal of the workbox. If the specified COP is more than 20 times the workbox diagonal away, GPLOT will use 20 times the workbox diagonal.

To set the VRP, use one of the calls:

CALL FclAbs (xAbs, yAbs, zAbs)
CALL FclAng (Phi, Theta, Radius)
CALL FclPt (x3, y3, z3)

The parameters to FclAbs and FclPt specify the VRP in terms of the "absolute 3D" and "relative 3D" coordinates respectively.

The parameters to FclAng specifiy the VRP relative to the COP as follows:

Radius - specifies the distance from the COP to the VRP in absolute 3D coordinates.

Theta - specifies the angle between the ray from COP to VRP and the xy-plane.

Phi - specifies the angle in the xy-plane between the positive x axis and the projection of the ray from COP to VRP.

FclAng should be called after VuAbs, or View have set the position of the COP as desired.

NOTE: The default VRP is at (0.5, 0.5, 0.5) in absolute 3D coordinates. If a program specifies the VRP, it should lie in the 3D workbox. IF the specified VRP lies outside of the workbox, GPlot will move it into the 3D workbox.

Ordinarily, GPLOT will adjust the projection of the workbox onto the screen so that the entire workbox is visible. If desired, a program may adjust the projection so that less of the workbox is visible in order to "zoom" in on one part of a 3D object. This can be done using the call

CALL VisAng (Angle)

Angle specifies the angle at the COP that is subtended by the virtual screen on which the 3D world is projected. If Angle is made quite small, the effect is to "zoom" in on a portion of the workbox.

NOTE: The GPLOT implementation of VisAng differs from the same named routine in DISSPLA. GPLOT always assumes a square virtual screen and so only one is specified whereas DISSPLA allows a rectangular screen specified by two angles.

14.3 Establishing a "relative" coordinate system

The 3D workbox will be given a relative 3D coordinate system by a call to Graf3D. Graf3D actually builds the projection as determined by the workbox and other viewing parameters and draws the axis system. Consequently, GRAF3D MUST be called AFTER the routines that establish the workbox and viewing parameters mentioned above, but BEFORE any routines to draw 3D objects given in the next section. Graf3D is called as follows:

```
CALL Graf3D (     x3Orig, x3Step, x3Max,
1                 y3Orig, y3Step, y3Max,
2                 z3Orig, z3Step, z3Max  )
```

The parameters are interpreted in the same way as the parameters to the 2D routine GRAF.

As with the 2D graf routines, many variations on the appearance of the axes are allowed. This is accomplished using the calls

```
CALL zIntAx
CALL zTicks ( iTicks )
CALL zRevTk
CALL zAxCtr
CALL zAxEnd ( Flag )
CALL zNoNum
CALL zAxAng ( Angle )
CALL zLabGr ( LabelList, LabelLength, NLabels )
```

The corresponding 2D calls such as xIntAx, yNoNum, etc., will also affect the appearance of the 3D axis system.

The effect of these calls is generally similar to the corresponding 2D GPLOT and DISSPLA calls. Note however that there are small differences in the interpretation of some of the parameters between the 2D and 3D routines. For example, the default direction for tick marks in 3D is directed outward from the workbox. Also, the rotation Angle specified in xAxAng, yAxAng and zAxAng will be interpreted as a rotation from a "normal" orientation which is parallel to the axis with the top of the letters pointed towards the axis. However, as the viewing parameters are changed, this would occasionally leave some of the lettering oriented generally downward. GPLOT calculates the orientation of the lettering and if it is "upside down" the lettering is rotated by 180 degrees.

14.4 3D Output

GPLOT provides 3 dimensional vectors, curves, and surfaces using the calls:

```
CALL Vectr3 (      xAbsFr, yAbsFr, zAbsFr,
1                  xAbsTo, yAbsTo, zAbsTo,  IVEC )

CALL RlVec3 (      xFrom,  yFrom,  zFrom,
1                  xTo,    yTo,    zTo,     IVEC )

CALL Curv3D ( xAray, yAray, zAray, NPts, IMark )

CALL Surmat ( ZMat, IXPts, IXDim, IYPts, IYDim, WkArea )
```

The vector and curve routines are analogous to the corresponding 2D routines. The parameters to Vectr3 specify 3D positions in "absolute 3D" coordinates whereas the parameters to RlVec3 and Curv3D specify the coordinates in terms of the "relative" coordinate system set up by the call to Graf3D.

3D surfaces are drawn using the routine Surmat. The parameters to Surmat are:

ZMat -- Real array with IxDim rows and IyDim columns, containing the z values of a grid of points defining the surface. The z values are specified in terms of "relative" coordinates.

IXPts -- The increment between successive rows of ZMat that are to be used in drawing the surface. For example, if IXPts = 2, Surmat will only use the data from alternate rows of ZMat.

IYPts -- The increment between successive columns of ZMat that are to be used in drawing the surface. For example, if IYPts = 3, Surmat will only use the data from every third column of ZMat.

IxDim, IyDim -- The dimensions of the array ZMat

WkArea -- Not used, but retained for compatability with DISSPLA.

Note that the x and y coordinates of the points on the surface are not specified directly in the call to Surmat. It is assumed that ZMat contains the Z-values of the surface evaluated on a grid of points (x,y) that are evenly spaced in the x and y directions. By default the ranges of x and y values of the data points are assumed to be the same as the ranges specified in the call to Graf3D. These default ranges can be changed with the call

```
CALL SurSze ( XMin, XMax, YMin, YMax )
```

which specifies the new range of x and y values to be associated with the z values in ZMat. This will NOT change the calibrations on the axes, or alter the "relative" coordinate system specified in Graf3D, but will adjust the position and shape of the surface relative to the workbox.

If a surface is to be constructed from data values that correspond to points (x,y) that do not form a regular grid of points, some preprocessing of the surface is necessary. GPlot provides the routines BGNMAT, MDFMAT, GETMAT and ENDMAT to approximate the z-values of a surface on a regular grid of points given points (x,y,z) which have x, y coordinates distributed randomly in the rectangle [X3Orig, X3Max] K [Y3Orig, Y3Max]. These routines may be used to construct a matrix ZMat, with up to 101 rows and 101 columns from an arbitrary number of scattered points. These routines are used as follows. The size of the matrix ZMat that is to be constructed first must be specified using BGNMAT:

CALL BGNMAT (IXDim, IYDim)

Then GETMAT would be called one or more times to approximate the effect on the surface due to a collection of randomly scattered points (x,y,z). The call

CALL GETMAT (xAray, yAray, zAray, nPoints, junk)

accepts three real arrays containing nPoints values specifying the coordinates of scattered data points. (The parameter "junk" is retained for compatibility with earlier versions of DISSPLA.)

It is assumed that each of the scattered points should add to the value of the surface at nearby regularly spaced grid points. For each scattered data point, (x,y,z), GETMAT accumulates the effect on each nearby grid point.

Finally, the call

CALL ENDMAT (ZMat, junk)

produces the array of z values needed by Surmat from the information accumulated during previous calls to GETMAT. The value returned in ZMat(i,j) is a weighted average of the z-values of scattered data points (x,y,z) that were "close" to the grid point (Xi, Yi). Each of the z values is weighted by

Weight = 1/(Distance + Offset) ** Power

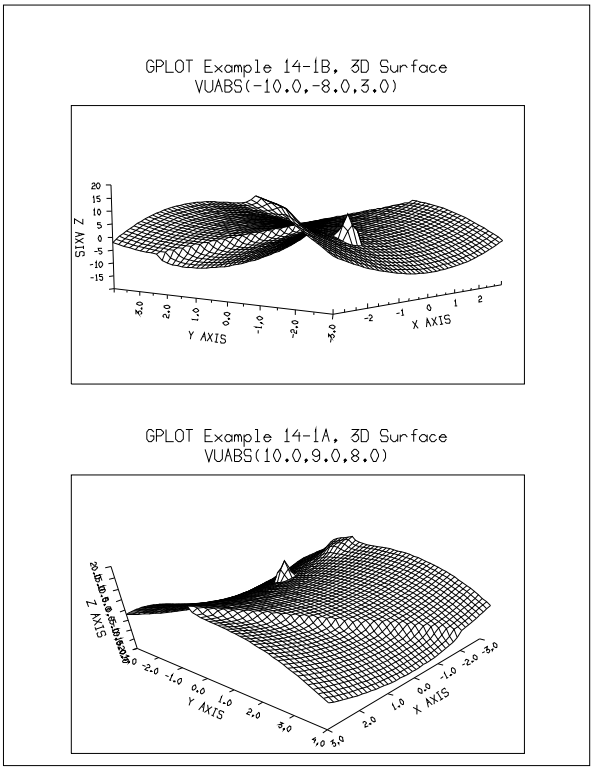
where Distance is the distance from (x,y) to (Xi, Yi) and Offset is a small positive value to avoid dividing by 0. Some of the parameters used in this process are set by the call

CALL MDFMAT (Ix, Iy, Power)

Ix specifies the number of rows of ZMat above and below the data point that are to be affected by the data point. Similarly, Iy specifies the number of columns of ZMat left and right of the data point that are to be affected. Power specifies the value of Power to be used in the calculation of the weight. The default value for each of these quantities is 2. MDFMAT need not be called if these defaults are acceptable.

Example 14-1 shows a 3D surface plot seen from two different viewpoints.

```
PROGRAM GPE15  ! Example 14-1
!++
!
! ---------------------- DEMO OF GPLOT SURFACE ROUTINES ---------
! This program constructs a 2-dimensional array of values and uses
! the GPlot surface routines to produce views of the three-D
! surface represented by those values.
!-------------------------------------------------------------------------
PROGRAM GPE15            ! GPLOT Example 14-1
PARAMETER Nx = 41        ! Number of data values in x direction
(#rows)
PARAMETER Ny = 51        ! Number of data values in y direction
(#columns)

PARAMETER xRange =  3.0  ! x Varies between -xRange and +xRange
PARAMETER yRange =  4.0  ! y Varies between -yRange and +yRange
PARAMETER zRange = 20.0  ! z Varies between -zRange and +zRange

DIMENSION DataArray(Nx,Ny)              call BuildArray( DataArray,
nx, ny, xRange, yRange )
!
!  Start GPlot and select the output device
!
call gstart
!
CALL PAGE(8.5,11.0)
CALL ORIENT('PORTRAIT')
CALL BLOWUP(0.4)
CALL PHYSOR(1.0,0.2)
CALL AREA2D(6.5,4.0)
call headin('GPLOT Example 14-1A, 3D Surface$', 100, 1.5, 2)
call headin('VUABS(10.0,9.0,8.0)$', 100, 1.5, 2)
call frame
!
call Axes3d( 'X AXIS$',100,'Y AXIS$',100,'Z AXIS$',100,
:               5.0, 5.0, 3.0 )
Vx = 10.0
Vy =  9.0
Vz =  8.0
call VuAbs( Vx,Vy,Vz)
call Graf3D( -xRange, 'SCALE', xRange,
:            -xRange, 'SCALE', yRange,
:            -zRange, 'SCALE', zRange )
CALL ENDGR(0)
!
!  Now repeat the process from a different viewpoint,
! using some of the axis attributes.
!
CALL xIntAx
CALL zIntAx ! Integer valued labels on x, z

CALL XTICKS( 3 ) ! Intermediate tick marks on x, y
CALL YTICKS( 2 )
CALL xRevTk
CALL zRevTk ! Reversed tick marks on x, z
CALL xAxEnd('NOENDS') ! Suppressed end labels on x, y and z
CALL yAxEnd('NOLAST')
CALL zAxEnd('NOFIRST')

CALL yaxang( 90.0)             ! Angled labels on y and z
CALL zaxang(-90.0)
!
! Now construct the surface again from a different viewpoint
!
CALL OREL(0.0,5.3)
CALL AREA2D(6.5,4.0)
call headin('GPLOT Example 14-1B, 3D Surface$', 100, 1.5, 2)
call headin('VUABS(-10.0,-8.0,3.0)$', 100, 1.5, 2)
call frame
call Axes3d( 'X AXIS$',100,'Y AXIS$',100,'Z AXIS$',100,
:             5.0, 5.0, 3.0 )
Vx = -10.0
Vy =  -8.0
Vz =   3.0

call VuAbs( Vx,Vy,Vz)
call Graf3D( -xRange, 'SCALE', xRange,
:            -xRange, 'SCALE', yRange,
:            -zRange, 'SCALE', zRange )
call SurMat( DataArray, 1, nx, 1, ny, 0 )
call endpl(0)
call GSTOP
end
```



15. Graphics Input

Graphics input can be obtained from input/output devices such as DECWindows terminals, DECwindows-Motif terminals, VAXstations, the VT240/241, and Tektronix 4014 emulators like the HDS 2200GX. In order for the keyboard input described in this section to work correctly on windowing terminals, the input focus must be on the window which was used to run your graphics program, not the graphics window itself. If input focus gets shifted to the graphics window, reselect the DECterm window which is running the graphics program. On Motif terminals, you may then need to put the cursor on the screen background and use the pop up Workspace menu to "shuffle down" the DECterm window to expose the graphics window again.

15.1 Reading the Cursor Position

To turn on the cursor and read the cursor position when input is terminated, use the call:

CALL HWINPT(XPOS,YPOS)

Input is terminated on the VAXstation by clicking the left-hand button on the mouse. Input is terminated on the other devices by typing a RETURN. When positioning the cursor on a Tektronix emulator, holding down the shift key before pressing the arrow keys will speed up cursor movement. HWINPT will return the cursor position in inches relative to the physical origin specified by the call to AREA2D. To obtain the position in data units, use the call:

CALL HWDATA (XVAL,YVAL)

If these routines are included when plotting on an output only device, an error message will be returned.

15.2 Using the Cursor to Modify the Plot

The subroutine CURSON was written to allow interaction with the plot on some devices (currently only the VAXstation, DECwindows devices, and the HDS terminals). It allows displaying the cursor position, optionally adding cursor position or text information to the plot and selecting a rectangular subplot area to zoom in on. Use the following syntax to call CURSON:

CALL CURSON (KEY,X0,X1,Y0,Y1)

where KEY is a CHARACTER*1 variable and X0,X1,Y0,Y1 are REAL*4 variables. When CURSON is called the cursor cross hairs are turned on. On the VAXstation, a continuous display of cursor position in data units will be displayed in the upper right corner of the plot. On the HDS the cursor coordinates will be displayed each time a return is typed. CURSON also allows input from the keyboard to add text to or delete text from the plot.

If you think you might want a copy of the screen plot, you should call your screen device plus either WISS or a hardcopy device at the beginning of the program. The only Tektronix 4014 emulator which currently supports all these features is the HDS graphics terminal, because a proprietary erase function has been used. For more consistent results between the screen and hardcopy when you use the HDS terminal as the screen device, software fonts are used on postscript printers when both devices are nominated.

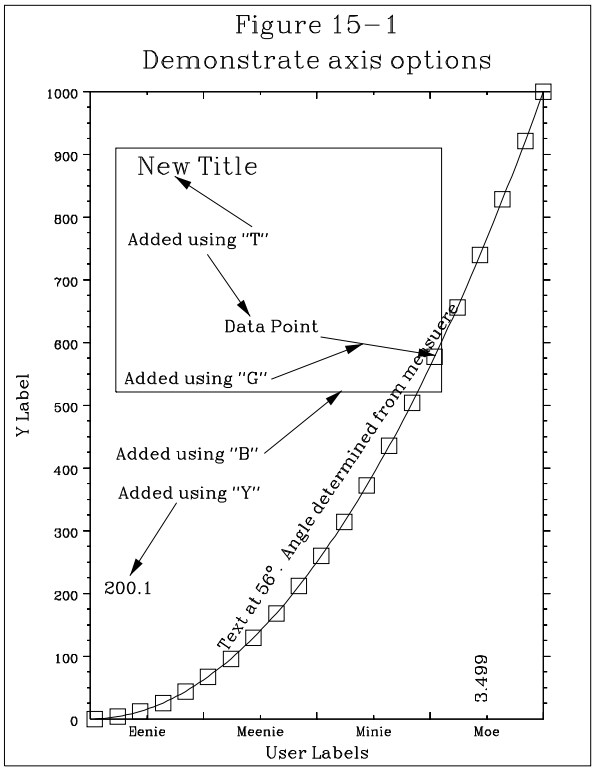
The following keys are recognized by CURSON:

H	(D)	Set text direction horizontal
V		Set text direction vertical
-		Set text direction upside down and backward
A		Allow input of text angle (default is 0.0R)
X		Write x-coordinate on plot
Y		Write y-coordinate on plot
P		Write both x and y coordinates on plot
G		Enter beginning/ending position of vector/arrow
W		Enter arrowhead type to be used in **G** command. (Default is IVEC=0, no arrowhead)
M		Measure the angle and length of a vector. (DECwindows only) When "M" is first typed, the prompt changes to a rubber-band line and the coordinate display changes to line length and angle. The second "M" changes the prompt back to a cross hair and the display back to the x and y coordinates.
T		Write Text on plot. The text will be in the current default size. An **up-arrow** will cause text to go into superscript mode and a **down-arrow** will cause it to go into subscript mode. The **right-arrow** will cause text size to be 1.5 times the default size and the **left-arrow** will cause it to be 0.67 times the default size. Use **PF2** to end superscript mode, subscript mode, or to return to the default size. **PF3-G** will cause text to go into Greek mode. **PF3-N** will set text back to normal. While in text mode, the **DELETE** key can be used to delete the previously typed character. The area erased on the screen may be larger than it should be if you are in superscript mode subscript mode, or at some text angles.
B		defines the Lower left and Upper right corner of a box which will be drawn on the screen. On DECwindows terminal the pointer will become a rubber-band box with origin at the position when the first "B" was typed. The pointer will return to a cross-hair when the box is finished with another "B" command. Once a box has been drawn the Zoom command may be used to zoom in on the selected region.
Z		Zoom in to selected box region, or Zoom back out to original region.
^W		Type CTRL-W to redraw the plot. This may be necessary after zooming in and out, or any other operation which corrupts the plot on the screen.
D		Delete previous segment added to plot.
CTRL-D		Dump current screen to plot device.

Q	Quit editing plot without saving additions (plot will be redrawn on the screen without additions).
E	Exit from CURSON and save additions to plot.

An example using Curson is shown in Figure 15-1. Compare the source code to the plot to see elements added through cursor interaction.

```
PROGRAM GPE16     ! Figure 15-1
DIMENSION X(100),Y(100)
DATA XORIG,DX,XMAX/0.0,100.,1000./,YORIG,DY,YMAX/0.0,100.0,1000.0/
CHARACTER*6 XLABEL(4)/'Eenie ','Meenie', 'Minie ','Moe   '/

CHARACTER*1 KEY
CALL WISS                   ! Turn on Workstation Segment Storage

CALL GSTART               ! Nominate Device
CALL PAGE(8.5,11.0)       ! Specify page size
CALL ORIENT('PORTRAIT') ! Force portrait orientation
CALL BLOWUP(0.4)        ! Scale down for manual
CALL AREA2D(6.5,9.0)    ! Define Subplot area
CALL DUPLX              ! Choose default font
CALL HEIGHT(0.25)       ! Use larger characters for small plot
CALL XNAME('User Labels$',100)        ! Define X Axis label
CALL YNAME('Y Label$',100)   ! Define Y Axis label
CALL HEADIN('Figure 15-1$',100,1.5,2)
CALL HEADIN('Demonstrate axis options$',100,1.5,2)
CALL XAXCTR       ! Center labels on X axis
CALL XREVTK       ! Reverse X Axis tick marks
CALL XTICKS(2)    ! Two tick marks per X division
CALL YNMADJ('NONE')
CALL YAXANG(0.0)  ! Use horizontal Y Axis labels
CALL YINTAX       ! Use Integers to label Y Axis
CALL YTICKS(4)    ! Four ticks per y division
CALL THKFRM(2.0)  ! Make frame twice normal thickness
CALL FRAME
CALL XLABGR(XLABEL,100,4,YORIG,DY,YMAX)
!  Compute and plot a quadratic curve
DO I=1,100
  X(I) = I*0.04
  Y(I) = 0.1*I*I
END DO
CALL CURVE(X,Y,100,5)

CALL CURSON(KEY,X0,X1,Y0,Y1)

CALL ENDPL(0)
CALL GSTOP

CALL EXIT
END
```



15.3 Printing the Screen

When your program is set up to do plot interaction, it is sometimes useful to be able to print the screen contents. To simplify doing this from your program, GPLOT includes a routine to select a screen dump device, PRINTER_SETUP, and a PRINT_SCREEN subroutine. The arguments to PRINTER_SETUP are:

NAME	Device name
TYPE	Device type number (GKS WS type)
WKCAT	GKS kstation category number
QUE	Queue name
FORM	Name of form needed to print plot.

The PRINT_SCREEN routine does not take any arguments. It will automatically copy all segments stored in Workstation Independent Segment Storage to the device specified through the call to PRINTER_SETUP.

Index

A
ALNLEG, 26, 44
ANGLE, 12, 16, 17, 31, 36, 37, 42, 44
ARC, 29, 30, 44
AREA2D, 1, 4, 5, 6, 9, 10, 20, 25, 29, 35, 41, 44

B
BARPAT, 30, 44
BARS, i, 1, 20, 21, 22, 23, 24, 30, 44
BARWID, 30, 44
BASALF, 14, 44
BGNMAT, 39, 44
BLCIR, 29, 44
BLELIP, 44
BLOWUP, 5, 44
BLPOLY, 29, 44
BLREC, 28, 29, 44
BLSLIC, 44
BLTREC, 28, 44

C
CARTOG, 12, 44
CGMCO, 3, 44
CGMTO, 3, 44
CHDET, 8, 44
CHHI, 8, 44
CHNDOT, 21, 24, 44
CHNDSH, 21, 44
CHPRI, 8, 44
CHRMSG, 10, 11, 14, 44
CHVIS, 8, 44
CIRCLE, 29, 44
COMPLX, 12, 44
COMPRS, 3, 44
CONANG, 31, 44
CONDIG, 31, 44
CONLIN, 31, 32, 44
CONMAK, 31, 44
CONMIN, 31, 44
CONNPT, 27, 44
CONTUR, 31, 32, 34, 44
CROSS, 16, 41, 42, 44
CURSON, 41, 42, 43, 44
CURV3D, 38, 44
CURVE, 1, 4, 5, 7, 11, 18, 20, 22, 23, 24, 25, 26, 27, 28, 33, 38, 43, 44

D

DASH, 21, 22, 24, 45
define plotter, 2, 3, 45
define screen, 3, 45
DELLEG, 26, 45
DOT, 21, 22, 24, 45
DOTGRID, 18, 45
DUPLX, 12, 15, 18, 29, 30, 31, 43, 45

E
EHITE, 23
ELARC, 45
ELLIPS, 29, 45
ELPARC, 29, 45
EMARK, 21, 45
ENDDEV, 7, 45
ENDGR, 6, 7, 20, 45
ENDMAT, 39, 45
ENDPL, 4, 6, 7, 25, 45
EXCRV, 21, 45
EXYCRV, 21, 45
EYCRV, 20, 45

F
FASHON, 12, 45
FCLABS, 36, 45
FCLANG, 36, 45
FCLPT, 36, 45
FONT, i, 11, 12, 13, 14, 15, 18, 22, 43, 45
fontid, 12, 45
FRAME, 7, 9, 18, 40, 43, 45
FUTURA, 12, 45

G
GETMAT, 39, 45
GKSM, 3, 45
GMENU, 2, 3, 45
gothic, 12, 45
GRACE, 9, 25, 45
GRAF, 4, 5, 7, 11, 15, 16, 17, 19, 22, 24, 25, 28, 37, 45
GRAF3D, 35, 36, 37, 38, 39, 45
greek, 12, 42, 45
GRID, 18, 25, 31, 38, 39, 45
GSCHH, 12, 45
GSCHUP, 12, 45
GSTART, 2, 3, 4, 5, 7, 11, 15, 18, 20, 22, 24, 25, 28, 29, 30, 31, 34, 40,
43, 45
GSTOP, 4, 5, 6, 7, 9, 11, 15, 18, 20, 22, 24, 25, 28, 29, 30, 31, 34, 40,
43, 45
GSTXFP, 12, 45

GTX, 10, 14, 45

H
HBARS, 30, 46
HEADIN, 10, 11, 46
HEIGHT, 11, 12, 14, 23, 25, 46
HWDATA, 41, 46
HWINPT, 41, 46

I
instruction, i, 14, 15, 46
INTAXS, 16, 46
INTNO, 10, 11, 46
ISGDET, 8, 46
ISGEX, 8, 46
ISGHI, 8, 46
ISGPRI, 9, 46
ISGVIS, 8, 46

K
KBMAP, 14, 15, 46

L
LA50, 46
LA75, 46
LEGEND, 1, 21, 22, 24, 26, 27, 46
LEGNAM, 22, 24, 26, 46
LINEAR, i, 1, 17, 19, 24, 25, 46
LINES, 11, 15, 18, 21, 22, 24, 26, 27, 31, 32, 34, 46
LINESP, 27, 46
LN03TK, 3, 46
LOGLOG, 19, 20, 46
LW, 2, 3, 46

M
MAKESEG, 7, 8, 46
MARKER, 4, 21, 23, 25, 46
mathematical, 12, 46
MDFMAT, 39, 46
MESSAG, 10, 11, 25, 46
MIXALF, 14, 46
MXIALF, 14, 46
MYLEGN, 46

N
NEWCLR, 22, 46
NOBRDR, 9, 46

O

OREL, 6, 7, 16, 25, 46

P
PAGE, i, 4, 5, 6, 7, 9, 10, 11, 15, 16, 18, 20, 22, 24, 25, 28, 35, 43, 47
pagex, 4, 5, 47
pagey, 4, 5, 47
PARA3, 24, 47
PARA5, 24, 47
PHYSOR, 6, 16, 20, 47
picfac, 23, 47
POLY3, 24, 25, 47
POLY5, 24, 47
POSTSCRIPT, 2, 3, 12, 13, 22, 28, 29, 41, 47
precis, 12, 47
PRINTER_SETUP, 43, 47
PRINT_SCREEN, 43, 47
PSPLIN, 25, 47

R
REALNO, 10, 11, 33, 47
RESET, 14, 25, 47
RLINT, 10, 11, 47
RLMESS, 10, 47
RLREAL, 10, 11, 47
RLVEC, 25, 27, 47
RLVEC3, 38, 47
russian, 12, 47

S
SCLPIC, 23, 47
SCMPLX, 12, 15, 47
script, 12, 47
SEGBEG, 8, 47
SEGDEL, 8, 47
SEGDLA, 8, 47
SEGEND, 8, 47
SEGREN, 8, 47
SERIF, 12, 47
SETDET, 8, 47
SETHI, 8, 47
SETPID, 8, 47
SETPRI, 8, 47
SETVIS, 8, 47
SIMPLX, 12, 47
SOLID, 21, 24, 27, 28, 29, 30, 32, 47
special, 1, 5, 11, 12, 14, 15, 18, 47
SPLINE, 25, 47
standard, 1, 12, 15, 30, 32, 47
STEP, 16, 17, 18, 25, 47

STRTPT, 27, 47
SURMAT, 38, 39, 48
SURSZE, 38, 48
SWISSB, 12, 48
SWISSL, 12, 48
SWISSM, 12, 48
symbols, 1, 4, 11, 23, 48

T
THKCRV, 22, 48
THKFRM, 9, 48
THKVEC, 27, 48
TK4014, 2, 3, 48
TK4014F, 3, 48
TRIPLX, 12, 48

V
VAXSTATION, 2, 3, 28, 41, 48
VBARS, 30, 48
VECTOR, 27, 38, 42, 48
VECTR3, 38, 48
VIEW, 4, 35, 36, 48
VISANG, 37, 48
VOLM3D, 35, 48
VT330, 3, 48
VT340, 3, 48
VUABS, 35, 36, 48
VUANGL, 35, 36, 48

X
X3NAME, 35, 48
XAXANG, 16, 37, 48
XAXCTR, 18, 43, 48
XAXEND, 17, 48
xaxis, 16, 19, 48
xcycle, 19, 20, 48
XGRAF, 17, 48
XGRAXS, 5, 16, 48
XINT, 11, 48
XINTAX, 16, 20, 31, 37, 40, 48
XINVRS, 10, 48
XLABGR, 17, 48
XLGAXS, 19, 48
XLOG, 19, 20, 48
xmax, 4, 5, 7, 11, 15, 16, 17, 22, 24, 28, 38, 48
XMESS, 11, 48
XNAME, 4, 16, 35, 48
XNMADJ, 17, 48
xorig, 5, 7, 11, 15, 16, 18, 20, 22, 24, 28, 43, 48

XPOSN, 10, 48
XREAL, 11, 49
XREVTK, 16, 18, 40, 43, 49
xstep, 4, 16, 17, 19, 34, 49
XTICKS, 18, 49

Y

Y3NAME, 35, 49
YAXANG, 17, 37, 49
YAXEND, 17, 49
yaxis, 4, 16, 19, 49
ycycle, 19, 20, 49
YGRAF, 17, 49
YGRAXS, 5, 6, 16, 25, 49
YINTAX, 16, 18, 20, 31, 43, 49
YINVRS, 10, 49
YLABGR, 17, 49
YLGAXS, 19, 49
YLOG, 19, 20, 49
ymax, 16, 38, 49
YNAME, 4, 16, 35, 49
YNMADJ, 17, 49
YNONUM, 37, 49
yorig, 4, 5, 7, 11, 15, 16, 17, 18, 19, 20, 22, 24, 28, 43, 49
YPOSN, 10, 49
YREVTK, 16, 20, 49
ystep, 4, 16, 17, 34, 49
YTICKS, 18, 49

Z

Z3NAME, 35, 49
ZAXANG, 37, 49
ZAXCTR, 37, 49
ZAXEND, 37, 49
ZINTAX, 37, 40, 49
ZNONUM, 37, 49
ZnUSE, 15, 49
ZREVTK, 37, 40, 49
ZTICKS, 37, 49
ZUSE, 15, 49