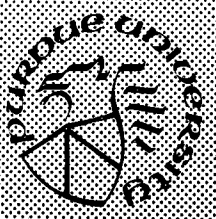
10-1-1984

The CRC Plotting Package

Carl Crawford
Purdue University

Mani Azimi
Purdue University

Malcolm Slaney
Purdue University



The CRC Plotting Package

Carl Crawford
Mani Azimi
Malcolm Slaney

TR-EE 84-43
October 1984

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

NAME

Introduction to the CRC Graphics Package

DESCRIPTION

The CRC Plotting Package is a device independent graphics system. Subroutines for generating graphics exist for programs written in FORTRAN or C. A program called Qplot exists to plot binary vectors generated as the output of any program.

The following list indicates the devices currently supported by this package:

DEVICE	MACHINE(S)	RESOLUTION (bits)	DISPLAY SIZE (in plot units)
Plot(5)		32768 x 32768	Device Dependent
Comtal	ARPA	512 x 512	10 x 10
Grinnel	ARPA	512 x 512	10 x 10
Printronix	P, EC, PP, MA	512 x 512	10 x 10
Retro-Graphic		512 x 256	10 x 10
HP 7221A	EG	10240 x 7800	13.1 x 10
Tektronix 4014	EG	1024 x 780	13.1 x 10
Tektronix 4010	ARPA	1024 x 780	13.1 x 10

The MACHINE field indicates which network computers currently have one of the specified devices. The DISPLAY SIZE is the x by y size of the plotting area for each device. The units are arbitrary and vary from device to device. All distances are measured in plot units.

The CRC Graphics package is currently in a state of transition. The original package contained all the support for each device and this accounted for a large number of lines in the package. With the addition of a number of new graphics devices to the network this approach has become unworkable so now the package supports *plot(5)* mode as the preferred output format. The *plot(1)* program is used to translate plot format into the actual commands needed to drive each graphics device.

In addition to the *plot* format, the CRC package supports two types of output formats. It is anticipated that support for these two formats will end sometime in the future and users will use the *plot(1g)* program to obtain all output. The CRC package can talk directly to Tektronix 4010 (or 4014) displays and HP 7221A flat-bed plotters. The Retro-Graphic works the same way as a Tektronix terminal. Finally, a number of devices can display graphics on a 512 by 512 bit plane. This bit map can then be copied to one of the following devices: Comtal graphics overlay, Comtal image display, Grinnell graphics overlay, Grinnell image display, or a Printronix line printer.

The bit plane can also be written out to a file. The programs Gplp (1) can be used to obtain overlayed outputs on the Printronix.

The rest of this document is subdivided into four sections. They are:

1) Documentation of Qplot, Strip7, Gplp, Hpd, and Gd.
2) Documentation of user callable subroutines.
3) Character font information.
4) Examples.

Online, all of this information is available with:

$help graphics/crc

are used. The Comtal is connected to the ARPA machine.

-G Display the plot on the Grinnel connected to the ARPA machine. The default action is to display the plot on graphics overlay 0 unless the -0, -1, -2, -3 or -i flags are used. A Grinnell is connected to the ARPA and the PB machines.

-i Display the plot on an image plane instead of a graphics overlay. This flag is the opposite of the *-g* flag. If this flag is used then the Comtal (*-c*) is the default device.

-g Display the plot on a graphic overlay instead of an image plane. This flag is set by default if ARPA is the host machine. This flag is the opposite of the *-i* flag. If this flag is used then the Comtal (*-c*) is the default device.

-n The plot is displayed on the graphic overlay or image plane *n*. The number *n* can be either 0,1, 2, 3. If this flag is used then the Comtal (*-c*) is the default device.

**** AXIS OPTIONS ****

-a No axes will be plotted.

axis=[xyz] An axis will be plotted only for axis listed after the equals sign. The default action is *axis=xyz* which labels all three axis of the plot. The option *axis=* is equivalent to the *-a* option.

-f A border will be drawn around the plot.

-F Do not draw a border around the plot. This is the default.

zaxis=r The x-y axis of the plot is drawn at *z=r*. The default value is -.05.

len=r Set the length of all axes to *r*. The default value is 8.

xlen=r The length of the x axis is changed from eight plot units to *r* units. This parameter is relative to the default value of 8. The actual length of the x axis in the 2-d plot coordinate system is a function of the angle of view and the *scfac*.

ylen=r The length of the y axis is changed from eight plot units to *r* units. This parameter is relative to the default value of 8. The actual length of the y axis in the 2-d plot coordinate system is a function of the angle of view and the *scfac*.

zlen=r The length of the z axis is changed from eight plot units to r units. This parameter is relative to the default value of 8. The actual length of the y axis in the 2-d plot coordinate system is a function of the angle of view and the *scfac*.

raxis=xyz Floating point (real) numbers will be used to label the axis. If just 'x' is specified then only the x-axis will be forced to floating point notation. If just 'y' is specified then only the y-axis will be forced to floating point notation. If just 'z' is specified then only the z-axis will be forced to floating point notation. If 'xyz' is used then all axis will be forced to floating point notation. If none of the axis are listed then all axis will be labelled with integers. Any combination of 'xyz' can be specified. The default notation is determined by the type of data used as input to the program. If integer data is used as input then the axis will be labelled with integers. Otherwise floating point notation is used.

dig=n The number of significant digits used in the annotation of the axes will be set to 'n'. The default value is six significant digits.

xdig=n Set the number of significant digits in the x axis to n.

ydig=n Set the number of significant digits in the y axis to n.

zdig=n Set the number of significant digits in the z axis to n.

tic=r	The distance between tic marks for both the x, y and z axes will be changed to r. The default value of r is one.
xtic=r	Set only the x tic mark distance.
ytic=r	Set only the y tic mark distance.
ztic=r	Set only the z tic mark distance.

**** LABEL OPTIONS ****

tl=str	The string will be used as a label at the top of the graph.
bl=str	The string will be used as a label at the bottom of the graph.
xl=str	The character string is used as a label along the x axis.
yl=str	The character string is used as a label along the y axis.
zl=str	The character string is used as a label next the z axis.
-l	The user will be prompted for labels not entered with the 'xl', 'yl' and 'zl' options.
-L	Don't prompt for labels. This option is the default.

**** GRAPH OPTIONS ****

direction=[xy]	Plot lines along the axis (or axes) specified by this parameter. The default is to plot the lines along the y axes only or *direction=y*. The options *direct=* and *dir=* are synonyms for this parameter. Use of the *direction=x* or *direction=y* parameters is encouraged since half the CPU time is used to produce the plot. In addition if a large number of lines are to be drawn, the graph will often appear simpler if lines are only drawn in one direction.
resolution=x	This parameter controls the resolution of the "hide" subroutine. The default value is 1.0 and larger values can be used to produce a more accurate estimate of the line intersections at the expense of more computer time. Values larger than one are generally needed only for publication quality plots of functions with large number of discontinuities. Conversely a value smaller than one will save computer time at the expense of small errors in the intersections.

The parse routines internal to **plot3d** allow for two default mechanisms to specify options. The first method is to create a file '.plot3drc' in your HOME directory (see environ(5)). In it, one can place options and flags that **plot3d** will use before it parses your command line. The syntax is one command or flag string per line.

As an example, consider the case where the user wishes to always obtain output on the Versatec and the plot be framed. Also the length of the axes are desired to be 6 units. The correct format for the file '.plot3drc' would be:

```
-vf
len=6
```

The second method to set default options is to set the environment variable 'PLOT3DARGS'. The format of 'PLOT3DARGS' is the same as the input command line.

The options set in the previous example can be set using the following procedure:

For /bin/sh:

```
$ PLOT3DARGS='-vf len=6'
$ export PLOT3DARGS
```

For /bin/csh

```
$ setenv PLOT3DARGS '-vf len=6'
```

If you need to set up labels with the 'bl=', or 'tl=' options, surround the respective option with double quotes. An example:

```
$PLOT3DARGS='-vf "tl=this is the title"'
```

There is no way to get a double quote into the label field.

Plot3d will parse the file '.plot3drc' first if it exists. Then it will parse 'PLOT3DARGS' if it exists. Finally, **plot3d** will parse the command line.

FILES

/etc/cpu Contains name of host

SEE ALSO

graphics/crc and **qplot(1)** documentation

AUTHOR

Mani Azimi assembled the **plot3d** program from software written by a couple of people. Carl Crawford did some of the work when he wrote **qplot.** The Fortran program that actually does the plotting was rewritten by Mani Azimi based on code that was available in the EE department for several years.

DIAGNOSTICS

Bad command line requests are flagged. Occasionally an 'out of bounds' warning will result from looking at a surface from too high an angle.

BUGS

Array size is limited on the PDP 11's, especially when using the bit mapped devices (Grinnel, Comtal, Versatec, Printronix.)

No check is made to insure that the number of points to read from the input file for each line (count*(skip+1)+begin) is consistent with the 'size' parameters.

The resolution parameter really shouldn't be necessary.

FUTURE

Eventually all devices except for plot format will be phased out. It is not practical for *plot3d* to know about every possible output device. Instead it is better that *plot3d* output a generic format with infinite resolution and let each device filter produce the best plot possible.

SUPPORT

Funding for the development of this software was provided by Prof. A.C.Kak of Electrical Engineering. Bug fixes and enhancements will be made on a time available basis by the author.

NAME

Qplot - Quick Plot

SYNOPSIS

qplot [arguments]

DESCRIPTION

Qplot is used to display one vector versus another on various graphic devices. By using **Qplot** with no arguments, a graph will be generated on the default Printronix line printer as defined in **gplp** (1). If ARPA is the host machine, the output will appear on the Comtal graphics overlay bit-plane zero.

A set of numbers will be read from a file called 'y' and used as the y vector. This vector will be displayed against a program generated x vector containing integers from zero to the number of points in the y vector minus one.

The input for **qplot** can be either a binary or an ASCII file. In a binary file the data is coded in the machine's internal format. The file can be generated using the **write** statement in C or an unformatted **write** in Fortran or Pascal. This is the most efficient form since it saves on both file space and machine time.

For simple applications it is also possible to give qplot an ASCII file with the numbers in a readable format. The user must be careful to edit out any titles or other non-numeric information before **qplot** is called. The numbers are read from the file "free format." In other words, spaces, tabs and newlines can all be used to separate the numbers. **Qplot** will read as many lines as necessary to get enough data for the plot as specified by the count, skip and begin parameters. The numbers that are read from the file must not contain any spaces.

Superimposing plots is done differently depending on the type of output format. For the HP plotter it is sufficient to use the same piece of paper for a number of graphs. For other devices it is necessary to use the *-b* option to turn off blanking. If the output is going to be sent to a bit mapped device (the Printronix printers, image processing devices or the Versatec on ARPA) then it is necessary to use several *plot3d* commands and to put the output in a file with the *g=graph* option. The resulting bit plane will be a logical-or of each picture and can be sent to the output device using either the *gplp(1g)*, *gd(1g)* or *gp(1g)* commands. When using *plot(5)* output format it is necessary to use the *-P* option and then redirect the output of *plot3d* into a file. Superimposing several graphs is accomplished with a string of commands like

```
qplot y=file1,f -P > graph.output
qplot y=file2,f -b -P >> graph.output
plot -Tver graph.output
```

The following options are available to modify the graph:

**** FILE SPECIFICATION OPTIONS ****

y=file,n — The y vector will be read from 'file' instead of 'y'. The number n is the byte declaration field and can be one of the following:

- c — Single byte, unsigned, fixed precision data.
- cs — Single byte, signed, fixed precision data.
- s — Two bytes, signed, fixed precision data.
- i — Two bytes (PDP 11's), signed, fixed precision data.
- i — Four bytes (Vax's), signed, fixed precision data.
- l — Four bytes, signed, fixed precision data.
- f — Four bytes, floating point data.

d Eight bytes, floating point data.

a ASCII numbers, free format data, file is readable, spaces, tabs and newlines are used to separate input points.

If *n* isn't specified, a comma is not needed after the file name. The default for n is 'f'.

y=,n The y vector is read from the default file 'y' but the byte declaration field is set to *n*.

-x The x vector will be read from a file called 'x' instead of having **qplot** generating it for you. The default file type is 'i'.

-X Turn off the -x flag. This option is set by default.

x=file,n The x vector is read from 'file' instead of 'x'. The byte declaration for this vector is set to n. The default type is 'i'. This option invokes the -x flag.

x=,n The x vector is read from the default input file 'x' but the byte declaration is set to n.

count=n Up to n points will be read from the input file. N can have a maximum value of 512. The default value of n is 512.

begin=n The first n points in the x and y files will be ignored. If 'begin' is negative, then the value actually used will be given by the product of 'count' and the absolute value of 'begin'. The starting point on the program generated x vector will be set to 'begin'. The default value of n is zero.

skip=n Only every (n+1)'th point will be read from the input file. The program generated x vector incremental value is set to 'n' + 1. The default value of n is zero.

**** SCALING OPTIONS ****

s=file **Qplot** generates the scale values for the x and y vectors by finding the minimum and maximum values of each vector. If 'file' exists in the current directory, the minimum and maximum values used by the scaling routines will be read from it. The format of the file is four numbers on a single line in the following format: 'xmin xmax ymin ymax'. The default name for 'file' is 'xybnd'.

ymin=r The ymin value read from the scale file or obtained from the y vector is replaced by r.

ymax=r The ymax value read from the scale file or obtained from the y vector is replaced by r.

xmin=r Works the same as the 'ymin=' option when the *-x* flag is used. Otherwise the program will create the x vector with a starting value of *r*.

xmax=r Works the same as the 'ymax=' option when the -x flag is used. If the program generates the x vector and the 'xmin=r1' and 'xmax=r2' options are also used, then the x vector will be a set of equally spaced point integers between r1 and r2.

-s The xmin, xmax, ymin, ymax values used to make the graph are written out to the file specified by 's=file'. This is useful in plotting multiple graphs with the same scale. The file has the same format as the input scale file.

-S Turn off the -s flag. This option is set by default.

**** POSITIONING AND REDUCING OPTIONS ****

xp=r — The x starting coordinate of the graph is moved to position 'r'. The default value of r is zero.

yp=r — Same as the 'xp=r' flag but the y origin is moved to 'r'.

scfac=r — The graph is expanded or reduced in size by 'r'. The default value is 1.0.

**** DEVICE SELECTION OPTIONS ****

-P — Output the graph to standard output in *plot (5)* format. This is a device independent format that can be piped through the *plot(1)* program.

plot=xx — Output the graph in plot format (See section 5 of the UNIX manual) piping all plotting commands to the plot program. The *xx* argument is used as a type argument to the plot command. Thus the use of *qplot plot=4014* is equivalent to the command string *qplot -P | plot -T4014*. For example the following are other valid devices: adm (to send to a terminal), ver (to send to the versatec), 4014 (to send to a Tek 4014 terminal). See the *plot(1)* manual for additional devices.

-p — The plot is sent to the Printronix line printer through **gplp** (1). This is the default output option for all host machines except for ARPA. See **gplp** (1) to determine to which network line printer the output will be sent.

site=XX — The output is sent to the network line printer specified by XX. See **opr** (1) for more information about the allowable printer names. This option invokes the '-p'

-t — Display the plot on a Tektronix 4010 or 4014 display. Standard output must end up on the specified terminal.

-T — Display the plot on a Retro-Graphics RG-512 in conjunction with an ADM-3A terminal.

-h — Display the plot on an HP plotter.

pen=n — When using the HP plotter, the pen in bin n is used to plot the graph. The default bin is 1. This invokes the '-h' flag.

speed=n — The HP plotter's pen movement velocity will be changed to 'n' cm/sec. The valid range for n is [1,36]. The default value is 36. This option will invoke the '-h' flag. flag.

-v — The plot is sent to the Versatec printer/plotter through a pipe to gp (1). NOTE: This option is obsolete on all machines except for the DIPL machine. Users should use the *plot=ver* option to obtain versatec output elsewhere on the network.

-o — The graph is written out to standard output.

g=file — The graph is written out to a file named 'file'. This file can be sent to the Versatec or Printronix with **gp** (1) and **gplp** (1), respectively.

op=str — The string is appended to the call to **gplp, gp, hpd,** and **gd.** This provides a method to change any of the options to the driver programs.

-b — The display device is not blanked before plotting. This option has no effect when used with the 'h', 'v', 'V', and 'p' flags. If the 'g=file' option is used and 'file' exists in the local directory before plotting begins, then the new graphics will overlay the graphics contained in 'file'.

-B — Turn off the -b flag. This option is set by default.

-c — The plot is displayed on the Comtal. The default action is to display the plot on

graphics overlay 0 unless the -0, -1, -2, -3 or -i flags are used. The Comtal is the default device if ARPA is the host machine, or the *-i, -g, -0, -1, -2* and *-3* flags are used. The Comtal is connected to the ARPA machine.

-G Display the plot on the Grinnel connected to the ARPA machine. The default action is to display the plot on graphics overlay 0 unless the -0, -1, -2, -3 or -i flags are used. A Grinnell is connected to the ARPA and the PB machines.

-i Display the plot on an image plane instead of a graphics overlay. This flag is the opposite of the *-g* flag. If this flag is used then the Comtal (*-c*) is the default device.

-g Display the plot on a graphic overlay instead of an image plane. This flag is set by default if ARPA is the host machine. This flag is the opposite of the *-i* flag. If this flag is used then the Comtal (*-c*) is the default device.

-n The plot is displayed on the graphic overlay or image plane *n*. The number *n* can be either 0,1, 2, 3. If this flag is used then the Comtal (*-c*) is the default device.

**** AXIS OPTIONS ****

-a No axes will be plotted.

-A Plot axes. This option is set by default.

xlen=r The length of the x axis will be changed from eight plot units to r units.

ylen=r The length of the y axis will be changed from eight plot units to r units.

len=r Set the length of both axes to r.

logx=n The 'x' axis will be logarithmic instead of linear. Valid values for n are -3, -2, -1, 1, 2, and 3. The logarithm base ten of the 'x' vector will be taken if n is negative. No tic marks will be plotted if n is -1 or 1. A few tic marks will be plotted if n is -2 or 2. All possible tic marks will be plotted if n is -3 or 3.

logy=n Same as the 'logx=' option except that it works for the 'y' axis.

raxis=xy Floating point (real) numbers will be used to label the axis. If just 'x' is specified then only the x-axis will be forced to floating point notation. If just 'y' is specified then only the y-axis will be forced to floating point notation. If 'xy' is used then both the x and the y-axis will be forced to floating point notation. The default notation is determined by the type of data used as input to the program. If integer data is used as input then the axis will be labelled with integers. Otherwise floating point notation is used.

dig=n The number of significant digits used in the annotation of the axes will be set to 'n'. The default value is six significant digits.

xdig=n Set the number of significant digits in the x axis to n.

ydig=n Set the number of significant digits in the y axis to n.

tic=r The distance between tic marks for both the x and the y axes will be changed to r. The default value of r is one.

xtic=r Set only the x tic mark distance.

ytic=r Set only the y tic mark distance.

-f A border will be drawn around the plot.

-F Don't frame the plot. This option is set by default.

**** LABEL OPTIONS ****

el=str	The string will be plotted just to the right of the last point in the line.
tl=str	The string will be used as a label at the top of the graph.
bl=str	The string will be used as a label at the bottom of the graph.
xl=str	The character string is used as a label below the x axis.
yl=str	The character string is used as a label next to the y axis.
-l	The user will be prompted for labels not entered with the 'xl' and 'yl' options.
-L	Don't prompt for labels. This option is the default.

**** LINE OPTIONS ****

-m	Mark every point in the line with an on-center symbol.
-M	Don't mark the line. This option is set by default.
j=n	Only every |j|'th point will have an on-center symbol on it. If j is negative, then the line connecting points will not be drawn. This option invokes the -m flag.
sym=n	One of 13 different on-center symbols can be selected. This option invokes the -m flag. The following on-center symbols can be used.

sym	*on-center symbol*
0	no on-center symbol
i	$'i-1'i=1,2,...,10
11	$*
12	$+
13	$,

-d	A dashed line is used instead of a solid line when drawing the graph.
-D	Don't draw dashed line. This option is set by default.
dash=r	The length of the visible portion of the dashed line is set to 'r'. The default value is 0.1. This option invokes the '-d' flag.
gap=r	Same as the 'dash=' option but affects the invisible portion of the dashed line.
-z	A bar graph is made instead of connecting vector elements with a solid line.
-Z	Don't draw bar graph. This option is set by default.

The parse routines internal to **qplot** allow for two default mechanisms to specify options. The first method is to create a file '.qplotrc' in your HOME directory (see environ (5)). In it, one can place options and flags that **qplot** will use before it parses your command line. The syntax is one command or flag string per line.

As an example, consider the case where the user wishes to always obtain output on the Versatec and the plot be framed. Also the length of the axes are desired to be 6 units and the number of digits in the axes annotation set to 4. The correct format for the file '.qplotrc' would be:

```
-vf
len=6
dig=4
```

The second method to set default options is to set the environment variable 'QPLOTARGS'. The format of 'QPLOTARGS' is the same as the input command line.

The options set in the previous example can be set using the following procedure:

For /bin/sh:

```
$QPLOTARGS='-vf len=6 dig=4'
$export QPLOTARGS
```

For /bin/csh
$setenv QPLOTARGS '-vf len=6 dig=4'

If you need to set up labels with the 'xl=', 'yl=', 'bl=', 'tl=', or 'el=' options, surround the respective option with double quotes. An example:

$QPLOTARGS='-vf "xl=this is the x axis label"'

There is no way to get a double quote into the label field.

Qplot will parse the file '.qplotrc' first, if it exists. Then it will parse 'QPLOTARGS', if it exists. Finally, **qplot** will parse the command line.

SEE ALSO:
graphics/crc

FUTURE
Eventually all devices except for plot format will be phased out. It is not practical for *qplot* to know about every possible output device. Instead it is better that *qplot* output a generic format with infinite resolution and let each device filter produce the best plot possible.

AUTHOR
Carl Crawford

FILES
/etc/cpu Contains name of host

APPENDIX: Input file conventions
Qplot users should write the data for the 'x' and 'y' files in binary format to make the most efficient use of the machine. This format will be explained in detail in this appendix for the benefit of users who have had little or no experience with binary files.

The normal output mode from programs is what will be called an ASCII or displayable format. This type of output can be directed to the terminal for inspection by the user. The output is generated with a **formatted** write. This type of data can be read by **qplot** using the ',a' modifier to the file name.

Internally a program assigns a certain number of bytes to each variable used. When a formatted write is initiated, the number is converted from the internal binary representation to a string containing displayable ASCII characters. When sending data between programs, it is a waste of time to convert data from binary to ASCII and then from ASCII back into binary. This is the reason why **qplot** encourages binary data as input.

Each programming language's internal variable types map into a **qplot** byte declaration field. The following list gives the correspondence between the internal variable types and the **qplot** byte declaration for some of the commonly used languages on the UNIX operating system.

C: (11's)

TYPE	DECLARATION
short	i or s
int	i or s
long	l
float	f
char	c or cs
double	d

C: (Vax)

TYPE	DECLARATION
short	s
int	i or l
long	i or l
float	f
char	c or cs
double	d

FORTRAN:(F77)

TYPE	DECLARATION
integer	l
integer*2	s
real	f
double	d
character*1	c or cs

FORTRAN: (F4P)

TYPE	DECLARATION
byte	c or cs
real	f
double	d
integer	s or i
integer*4	l

APL: (APL)

TYPE	DECLARATION
numeric	d
character	c or cs

APL: (APL2)

TYPE	DECLARATION
numeric	f
character	c or cs

PASCAL (11's)

TYPE	DECLARATION
integer	s or i
real	f

PASCAL (Vax)

TYPE	DECLARATION
integer	l or i
real	f

Unformatted writes in C can be obtained using the **fwrite** subroutine. See stdio(3) for more information.

In FORTRAN using the F77 compiler there are two ways to obtain unformated writes. The first is to use the **ucreat** and **uwrite** subroutines. They are the analog to C's **write** and **creat** subroutines. See the documentation for the C versions for more information on the syntax. When compiling your program add '-lU77' to the end of the command line.

The easiest way to get unformated writes from FORTRAN is to just write the data without a 'format' statement. An example would be:

```
real x(100),y(100)

write(2)x
write(3)y
```

In this example two files would be generated named 'fort.2' and 'fort.3'. The F77 compiler also generates code to produce record counts in its binary files. The program strip7(1) will remove the record counts from binary files generated with F77. The output file names can be changed using the FORTRAN subroutine **open.** See the compiler manual for more information.

NAME

gd - Image Processing Graphics Driver

SYNOPSIS

gd [-bcGio0123] [-mHOST] [if=stdin]

DESCRIPTION

Gd is an interface program between the output of the plotting subroutines and an image display device. Currently, *gd* supports Comtal and Grinnell image displays or graphics overlays.

The file:

/usr/lib/graphics/gd.site

contains information about which network machine has these devices. The file contains 'local' if there are devices connected locally or the name of the HOST that does have the display devices. If *gd* can not run locally, then the graphics information is sent to:

HOST!/usr/lib/graphics/gd [-Gcio0123]

where HOST comes from either the file 'gd.site' or from the option '-mHOST'. The mode is determined by the options the local Gd is invoked with.

If the *-b* flag is used the device is not cleared (blanked) before the graphical data is written on the screen. The *if=file* parameter specifies that the graphics information should be read from 'file'. The default input mode is standard input.

The device to write on is determined by a combination of the default action and the flags specified on the command line. The default action is to write the image on the Comtal Graphics Overlay number 0. In addition on the Comtal it is possible to determine the current image number being displayed so when writing to a Comtal image plane (-ci) the default action is to write to that image. These actions can be modified by using the following options.

-c	Write the data on the Comtal instead of the Grinnell.
-G	Write the data on the Grinnell instead of the Comtal.
-g	Write the data to a graphics overlay instead of an image plane.
-i	Write the data to an image plane instead of a graphics overlay.
-[n]	Write on either the image plane or graphics overlay labelled *n*. The digit *n* can be either 0, 1, 2 or 3. (Note that not all of the image processing hardware is configured with all 4 planes.)

FILES

/usr/lib/graphics/gd.site	contains default site name
/etc/cpu	contains local host name
/dev/Gr/[0123]	Grinnell Image Planes
/dev/Gr/gov[0123]	Grinnell Graphics Overlays
/dev/ct[012]	Comtal images
/dev/ct/tc	Comtal status
/dev/ct/gov[012]	Comtal graphic overlays

AUTHOR

Carl Crawford and Malcolm Slaney

SEE ALSO

graphics/crc

NAME

gp - graphics plot to versatec

SYNOPSIS

gp [-ahfi] [-mHOST] file1 file2 ... filen

DESCRIPTION

This command is obsolete on all ECN machines except for ARPA. Users desiring output on other ECN versatec machines should use *plot(5g)* format and send the plot to the versatec using the *plot(1g)* command.

Gp copies graphic files to a Versatec plotter. The input files are 'or'ed together to form one picture. The -i flag will force the first file to be read from standard input.

Gp can be initiated from any machine on the network. The output will come out on the Versatec connected to the ARPA machine. Also file names '0', '1', and '2' refer to the Comtal graphics overlays 0, 1, and 2, respectively.

The option -f produces 4:1 output only on the ARPA Versatec. The option -h, which is a default flag, produces 2:1 output.

The flag -s causes *gp* not to output the message given to the user when *gp* finishes.

The -m flag is used to change the name of the host where the output is sent to. The characters immediately after 'm' form the name of the new site. As an example, the command "gp -marpa file" will always send the output to the Versatec on the ARPA machine.

FILES

/usr/lib/graphics/gp.site	default site specifications
/etc/cpu	contains local host name /dev/vc Versatec

AUTHOR

Carl Crawford

SEE ALSO

gplp(1), qplot(1), graphics/crc

NOTES

The file 'gp.site' contains the name of the machine where the output is routed. If the machine has a Versatec then the name should be 'local #', where # is the length of the Versatec line buffer. For the ARPA Versatec, # is 132.

NAME

gplp - graphics print on a line printer

SYNOPSIS

gplp [-XX] [-i] file1 ... filen

DESCRIPTION

Gplp copies graphic files to a Printronix line printer. The input files are 'or'ed together to form one picture. The -i flag will force the first file to be read from standard input.

Depending on which host machine 'gplp' is run from, the output will go to various line printers on the network. As of this date the following table indicates which host's Printronix will be used for each originating computer:

ORIGINATING MACHINE	OPR SITE -XX
aa	ep
ca	cp
cb	cp
ea	ep
ec	ep
ed	ep
ee	ep
ef	ep
eg	em
ga	ep
gb	ep
ka	ep
ma	mp
mb	mp
pa	pp
pb	pp
pc	pp
eeg	ep
arpa	ep

The default site name can be changed using the -XX option. See *opr(1)* for more information about the valid printer sites.

FILES

/usr/lib/graphics/gplp.site	contains default site name

AUTHOR

Carl Crawford

SEE ALSO

qplot(1), gp(1), graphics/crc

NOTES

The file 'gplp.site' has the form '-XX<cr>', where XX is the name of the default line printer.

NAME

hpd - HP plotter driver

SYNOPSIS

hpd [-mHOST] [device]

DESCRIPTION

Hpd is an interface program between the output of the plotting subroutines and the HP plotter. It passes standard input to the plotter specified by the 'device' field. Specification of a host with the '-m' flag will cause:

HOST!/usr/lib/graphics/hpd [device]

to be executed. Thus, any plotter on the network can be reached.

The default HOST is contained in the file:

/usr/lib/graphics/hpd.site.

The file contains 'local' if there is a plotter on the machine or the name of a HOST that has a plotter.

The program will time out after 3 minutes after the plotter hangs.

This program unfortunately only knows about one plotter per machine. The *plot(1g)* routine in conjunction with the *-P* option to *qplot* or *plot3d* can be used to send output to the other plotter(s).

FILES

/usr/lib/graphics/hpd.site	contains default site name
/dev/plt0	default path to plotter
/etc/cpu	contains local host name

AUTHOR

Carl Crawford

SEE ALSO

graphics/crc

NAME

strip7 - remove byte counts from f77 data files

SYNOPSIS

strip7 [-r] file1 ...

DESCRIPTION

Strip7 removes the byte counts that f77 puts into its binary output files. The 'stripped' files replace the original files. The '-r' option causes the 'stripped' files to be placed in a new files. The new file names are the original file names with a ".s7" suffix.

COMMENT

Plots will be obtained even if 'strip7' is not run before calling *qplot (1g)* and *plot3d(1g)*. The resulting graphics will be garbage because *qplot* and *plot3d* will interpret the byte count as a floating point number and it is usually much larger than any data in the user's file..

FILES

strip7.tmp temp file

AUTHOR

Carl Crawford

BUGS

The maximum record length is only 4096 bytes.

NAME

alpha - Return Tektronix to alpha-numeric mode

SYNOPSIS

From C:

```
alpha()
```

From F77:

```
call alpha
```

DESCRIPTION

The software leaves the Tektronix in the plot state after each subrotuine call. If the user writes to the terminal without a proceeding the write with a call to *alpha*, the Tektronix will think that the alpha-numeric data from the user is valid plot information. Garbage graphics will result.

The terminal remains in an alpha-numeric state until the next graphics call.

A call to *alpha* when plotting on the HP plotter will cause the software to flush its internal output buffers. This allows for pseudo interactive plotting on the HP plotter.

NAME

axis - Draw axis

SYNOPSIS

From C:

```
float  x,y,size,min,max;
int    xy,flag;
char   *label;

axis(x,y,label,xy,size,min,max,flag);
```

From F77:

```
real    x,y,size,min,max,
integer xy,flag
character label()

call axis(x,y,label,xy,size,min,max,flag)
```

DESCRIPTION

To draw a labeled coordinate axis with numerically annotated tic-marks at one unit intervals.

The following parameters are used to specify the details of the axis.

x, y	the coordinates (in plot units) of the starting point of the axis.
label	a pointer to a zero terminated string which is to be used as an axis label.
xy	One of the following four values indicate the direction and orientation of the axis. 0: plot axis at zero degrees, tic marks below the axis. 1: plot axis at ninety degrees, tic marks left of the axis. 2: plot axis at zero degrees, tic marks above the axis. 3: plot axis at ninety degrees, tic marks right of the axis.
size	the length (in plot units) of the coordinate axis.
min	the numerical value corresponding to the first tic-mark of the axis (at the point (x,y)), usually the same as the value computed by 'scale'.
max	The numerical value corresponding to the last tic-mark of the axis.
flag	if flag is positive, the numerical values calculated for the tic-marks will be truncated to form integers. If flag is zero, the annotations will be floating point. If flag is negative, no numerical annotation will be done.

REMARKS

The distance between tic-marks can be changed by setting the variable *ticdis* to the desired distance using subroutine *axisv*. Ticdis will be adjusted so that it is an integer divisor of size. If ticdis is too small, the annotations at the tic-marks will run over each other.

When integer annotations are desired, (*max* - *min*) should be divisible by (*size* / *ticdis*) to prevent truncation errors.

When the tic numerical values become very small or very large (in absolute value), the values will be converted into base ten mantissa and characteristic notation with the characteristic appended to the label.

The axis itself starts at (x,y) and all the labeling appears below, next to, or above it. Be sure to allow space for the labeling.

The number of significant digits used in the annotations can be changed by setting the variable *digits* to the number of digits desired using the *axisv* subroutine.

NAME

axisv - change parameters of axis

SYNOPSIS

From C:

```
float   ticdis;
int     digits;

axisv(ticdis,digits);
```

From F77:

```
real    ticdis
integer digits

call axisv(ticdis,digits)
```

DESCRIPTION

To change the default distance between axis tic marks and the number of significant digits in the numerical annotations. The parameters are as follows

ticdis — the distance (in plot units) between tic-marks on the axis.

digits — the number of significant digits in the numerical annotation

REMARKS

The default values for ticdis and digits are 1.0 and 6, respectively.

The values will be in effect until the next call to *axisv* for all subsequent calls to *axis* and *laxis*.

The *axis* subroutine will modify *ticdis* so that the length of the axis divided by *ticdis* is an integer.

SEE ALSO

axis(3g), laxis(3g)

NAME

dline - draw a parametric line

SYNOPSIS

From C:

```
float  *x,*y,*dsh,*gap,lx,ly;
int    n,m;

dline(x,y,n,dsh,gap,m,lx,ly);
```

From F77:

```
real    x(),y(),dsh(),gap(),lx,ly
integer n,m

call dline(x,y,n,dsh,gap,m,lx,ly)
```

DESCRIPTION

To draw a plot of a parametric relationship (x[i],y[i]), i=0,2,...,n-1, with a dashed line. The *i=n'th* and *i=n+1'th* elements of each array should represent the minimum and maximum values of the points in that vector. The user can call the routine *scale(3g)* to automatically find the minimum and maximum values. All data values are scaled to these values.

The following parameters are used to specify the line.

x,y — pointers to arrays containing the x and y coordinates of a parametric relationship (x[i],y[i]).

n — the number of points (x[i],y[i]) in the parametric relationship.

dsh — a pointer to an array of length m containing the lengths of the sequence of dashes to be used in plotting the relationship.

gap — a pointer to an array of length m containing the lengths of the sequence of spaces between dashes to be used in plotting the relationship.

m — the number of dashes and gaps in the dash-gap sequence.

lx — the length (in plot units) that (xmax - xmin) is mapped to in page coordinates. Should be the same value used in the axis subroutine.

ly — the length (in plot units) that (ymax - ymin) is mapped to in page coordinates. Should be the value used in the axis subroutine.

REMARKS

Arbitrarily complex dash-gap sequences may be specified. Dshline first plots a dash of length dsh[0], then a gap of length gap[0], then a dash of length dsh[1], etc. After plotting gap[m-1], the sequence is repeated starting again with dsh[0].

If *m = 0, dsh* and *gap* are ignored and the curve is plotted as a solid line curve with subroutine *line*.

The arrays pointed to by *x* and *y* must be dimensioned at least *n+2*, and the minimum and maximum values of each array must be stored in the *n+1st* and *n+2nd* array positions respectively.

An error will occur if xmin = xmax or ymin = ymax.

NAME

factor - change scale factor

SYNOPSIS

From C:

```
float   sf;

factor(sf);
```

From F77:

```
real    sf

call factor(sf)
```

DESCRIPTION

To provide for changing the scale factor, initially 1.0, for subsequent x and y coordinates. The *sf* parameters is the new scale factor to be used in scaling all subsequent x and y coordinates unless and until factor is called again. All plotting is scaled by this factor thus the user coordinates (x,y) are scaled to the point (sf*x,sf*y).

NAME

fname - change default file name

SYNOPSIS

From C:

```
char *name;

fname(name);
```

From F77:

```
character name()

call fname(name)
```

DESCRIPTION

To change the default file name when the plots (3g) device is 0. The parameter *name* is a pointer to a zero terminated string containing the name of the file that the final graphics output will be written. If the name is "-", then the graphics output will be written out to standard output.

REMARKS

Fname should be called before the call to *plots*.

NAME

laxis - draw a logarithmic axis

SYNOPSIS

From C:

```
float x,y,size;
int xy,logmin,logmax,flag;
char *label;

laxis(x,y,label,xy,size,logmin,logmax,flag);
```

From F77:

```
real x,y,size
integer xy,logmin,logmax,flag
character label()

call laxis(x,y,label,xy,size,logmin,logmax,flag)
```

DESCRIPTION

Laxis draws a labelled logarithmic coordinate axis with numerically annotated tic marks. The following parameters are used:

x,y	the coordinates of the starting point of the axis.
label	a pointer to a zero terminated string which is to be used as an axis label.
xy	0: plot the axis at zero degrees, tic marks below the axis. 1: plot the axis at ninety degrees, tic marks to the left of the axis. 2: plot the axis at zero degrees, tic marks above the axis. 3: plot the axis at ninety degrees, tic marks to the right of the axis.
size	the length of the coordinate axis in plot units.
logmin	the exponent corresponding to the first tic mark of the axis (at the point (x,y)), usually the floor of the min computed by 'scale'.
logmax	the exponent corresponding to the last tic mark of the axis, usually the ceiling of the max computed by 'scale'.
flag	0: default, all logarithmic tic marks plotted. +/− 1: no logarithmic tic marks plotted. +/− 2: only the logarithmic tic marks corresponding to 2 and 5 are plotted. +/− 3: all logarithmic tic marks are plotted. If flag is negative, no numerical annotation will be done.

REMARKS

The user must supply *scale, line, and dline* with the log base 10 of the data to be plotted. Also, the min and max supplied to *line* and *dline* must be logmin and logmax, respectively.

AUTHOR

Jeffery L. Gray

NAME

Library Information

DESCRIPTION

The graphics library (/u/lib/libG.a) contains user callable subroutines for generating graphics. They all are callable from a program written in C or F77. The only restriction on the calling sequence is that *plots* is called before any other plot calls and 'plot(0.0,0.0,999)' is the last call. It should be noted that after the plot is terminated, *plots(3g)* can be called again.

Here is a list of the available routines:

alpha	returns Tektronix to alpha-numeric mode
axis	plots numerically annotated axes
axisv	set environment variables for *axis* and *laxis*
dline	same as *line* but plots with a dashed line
factor	change the scale factor of the plot
fname	set output file name when graphics device is a file
laxis	plots numerically annotated logarithmic axes
line	plots a line through a set of coordinate pairs
newpen	changes the pen on the HP plotter
number	provides formatted numeric labelling
plot	moves pen and terminates plotting
plots	initializes the plotting devices
scale	computes scale values for line, *dline*, and *sline*
site	change the default line printer name in link to Gplp
sline	same as *line* but can put on-center symbols on the line
speed	changes the plotting speed for the HP plotter
symbol	puts strings on the graph
where	returns current pen position and scale values

Compile your program with
one of the following commands:

```
From F77 (11's):
$f77 prog.f -i -lG
From F77 (Vax):
$f77 prog.f -lG
From C (11's):
$cc prog.c -i -lG -lm
From C (Vax):
$cc prog.c -lG -lm
```

SEE ALSO

See the CRC Graphics Introduction for more information.

NAME

line - draw a parametric line

SYNOPSIS

From C:

```
float  *x,*y,lx,ly;
int    n,bar;

line(x,y,n,bar,lx,ly);
```

From F77:

```
real    x(),y(),lx,ly
integer n,bar

call line(x,y,n,bar,lx,ly)
```

DESCRIPTION

To draw a line of a parametric relationship (x[i],y[i]), i=0,2,...,n-1. The *i=n'th* and *i=n+1'th* elements of each array should represent the minimum and maximum values of the points in that vector. The user can call the routine *scale(3g)* to automatically find the minimum and maximum values. The following parameters are used:

x,y	pointers to arrays containing the x and y coordinates of the parametric relationship (x[i],y[i]).
n	the number of points in the parametric relationship.
bar	normally a straight line is drawn between two points but if bar is non-zero, zero order interpolation will be used between points.
lx	the length (in plot units) that (xmax - xmin) is mapped to in page coordinates.
ly	the length (in plot units) that (ymax - ymin) is mapped to in page coordinates.

REMARKS

The Xi's and Yi's need not represent plot coordinates in inches.

The minimum and maximum values for the arrays must be stored in the n+1st and n+2nd locations of the arrays respectively. This can be easily done using the routine *scale(3g)*.

The point (x[i],y[i]) will be plotted at the page coordinates:

```
x = ((x[i] - x[n]) / (x[n+1] - x[n])) * lx
y = ((y[i] - y[n]) / (y[n+1] - y[n])) * ly
```

relative to the current origin. Hence depending upon the location of the current origin, the adjusted minima need not actually be the minima of the data which is to be plotted so long as each point (x[i],y[i]) has page coordinates which lie within the plotting area.

An error will occur if xmin = xmax or ymin = ymax.

The values of *lx* and *ly* are normally the values passed to the axis subroutines.

NAME

newpen - change pens on the HP plotter

SYNOPSIS

From C:

```
int     n;

newpen(n);
```

From F77:

```
integer n

call newpen(n)
```

DESCRIPTION

To deselect the current pen and select one of the other pens in the HP plotter. The command is ignored if the HP plotter is not the selected output device. *N* should be set to the bin number of the new pen, n=1,2,3,4, which is to be selected.

REMARKS

Initially, pen 1 is selected. When a pen is selected, the current pen is raised and the newly selected pen is left in the up position.

If n is the number of the currently selected pen, it is raised and left in the up position.

NAME

number - provide numeric labelling on plots

SYNOPSIS

From C:

```
float  x,y,height,angle;
char   *format;
(num type must conform with format)

number(x,y,height,angle,format,num);
```

From F77:

```
real   x,y,height,angle
character format()
(num type must conform with format)

call number(x,y,height,angle,format,num)
```

DESCRIPTION

This routine plots a single number. It is usually used as part of plotting the axis and the following parameters are used.

x,y	the coordinates of the point where the lower left corner of the first character is to be plotted.
height	the height (in plot units) of the alphanumeric character string.
angle	the angle (in degrees) counterclockwise from the +x direction at which the string is to be plotted.
format	a pointer to a zero terminated string which contains the 'C format' by which num is to be converted for plotting.
num	the number whose value is to be plotted.

REMARKS

From f77 integers must be printed out using the 'c' format %ld, since fortran integers are two bytes.

The width of the characters is four-sevenths of height and they are spaced at intervals of six-sevenths of height. The space between characters is two-sevenths of height.

A second set of characters can be accessed by preceeding a valid character with a '$'. A '$' can be obtained using '$$'. The second set contains Greek letters and some special mathematical symbols. See *fontinfo(8g)* for more information about this alternate character font.

See the documentation for *printf(3)* for more information on the syntax of the format statement *number* expects.

NAME

plot - move the pen

SYNOPSIS

From C:

```
float   x,y;
int     i;

plot(x,y,i);
```

From F77:

```
real    x,y
integer i

call plot(x,y,i)
```

DESCRIPTION

To provide for moving the pen in a straight line from its current position to a new position, and for terminating the plotting subroutines. The following parameters are used.

x,y — the coordinates of the point (in plot units) to which the pen is to be moved relative to the current origin.

i

+/- 1: do not change vertical position of pen.
+/- 2: put pen into down position.
+/- 3: put pen into up position.
999: terminate plot subroutines

REMARKS

If *i* is +/- 1, +/- 2, or +/- 3, the pen is moved from its current position to the point (x,y) along a straight line with the vertical position of the pen as specified.

If *i* is -1, -2, or -3, the point (x,y) becomes the new origin for subsequent plotting. Unless and until the origin is again redefined, all future coordinates will specify positions with respect to this point.

If *i* is 999, the values of x and y are disregarded and may as well be '0.0'. An error will occur if subsequent plot calls are made prior to another call to *plots*.

NAME

plots - initialize CRC Graphics Package

SYNOPSIS

From C:

```
int     dev,overwrite;
char    *options;

plots(dev,overwrite,options);
```

From F77:

```
integer dev,overwrite
char options()

call plots(dev,overwrite,options)
```

DESCRIPTION

Plots is used to allocate buffers and initialize the devices before plotting can begin. Except for calls to *site(3g)* and *fname(3g)* this subroutine must be called before any other subroutines are called. The following parameters should be passed to *plots*.

dev The device the plot should be sent to. The following device numbers are currently supported (add 64 to the number in the table if the option string is used):

0	file or standard output
8	Versatec through gp(1g) (ARPA Machine Only!!!)
16	Printronix through gplp (1g) and opr (1g)
1	Comtal graphics overlay 0(*)
9	Comtal graphics overlay 1(*)
17	Comtal graphics overlay 2(*)
2	Comtal image image displayed(*)
10	Comtal image 0(*)
18	Comtal image 1(*)
26	Comtal image 2(*)
3	Grinnell graphics overlay 0(*)
11	Grinnell graphics overlay 1(*)
19	Grinnell graphics overlay 2(*)
27	Grinnell graphics overlay 3(*)
4	Grinnell Image being Displayed (*)
12	Grinnell Image Plane 0(*)
20	Grinnell Image Plane 1(*)
28	Grinnell Image Plane 2(*)
36	Grinnell Image Plane 3(*)
44	Grinnell Image Plane 4(*)
5	Plot Subroutines through plot(I)
6	Tektronix through standard output
14	Retro-Graphics through standard output
22	Tektronix 4113

7 HP through /u/lib/graphics/hpd

The output to those devices marked (*) is sent via the *gd(1g)* program.

overwrite If this parameters is zero then the specified device will be cleared before plotting. Otherwise (if overwrite is non zero) then the new plot will overwrite whatever is currently being displayed.

options A null terminated string that contains options for *gplp(1g), hpd(1g), gd(1g)* and *plot(1g)*. The value passed as *dev* must be 64 plus the number found in the table above for this string to be used. When *dev* is equal to 69 then the *plot(5)* output is automatically passed to *plot(1g)*. The *options* string is appened to the string "-T" and the result is passed as the first argument to *plot(1g)*. Thus use *plots(69,0,"ver")* to use the Versatec through the *plot(1g)* program, *plots(71,0,"eg")* to send a plot to the a HP plotter on the 'eg' machine. *plots(6,0)* is also a valid plots call when the option string isn't needed.

REMARKS

The default file name when *dev* equals zero can be changed using the subroutine *fname*. Also if the file exists in the local directory before plotting begins, then the new graphics will overlay the graphics contained in 'file'.

For dev=printronix(16) there exists a subroutine *site(3g)* to change the name of the default line printer.

Space limitations occur on the PDP 11 computers when the device selected is a bit mode device (all dev numbers except for 3, 4, 11). For these devices the software has to allocate an 64K byte buffer. If your program has a lot of text and data, then the graphics software can't get its space. An error message will printed in these situations. When you are on the 11's try compiling your program with the loader option '-i' to get more space for data.

The origin is set to (0,0) before plotting begins. This coordinate is the lower left hand corner of the plot device.

The *options* field is an optional parameter. It is intended for advanced applications of the package with special emphasis placed on the network interface and future expansion of device daemons.

NOTE

If the CRC-Graphics routines *fname(3g)* and *site(3g)* are needed then they must be called before the first call to *plots(3g)*.

NAME

scale - find the minimum and maximum values of a vector

SYNOPSIS

From C:

```
float  *a;
int    n;

scale(a,n);
```

From F77:

```
real    a()
integer n

call scale(a,n)
```

DESCRIPTION

This routine is most often used to find the minimum and maximum values of a vector. The result is placed at the end of the vector in locations *a[n]* and *a[n+1]* (here the array subscripting is assumed to start from zero as in C). The first argument *a* is a pointer to an array and *n* is the number of points in the array.

REMARKS

The array must be dimensioned at least n+2. The minimum value of the array is stored in a[n] and the maximum is stored in a[n+1].

In Fortran arrays are referenced starting from 1 so the minimum value is stored in a(n+1) and the maximum is stored in a(n+2).

The minimum and maximum values are NOT adjusted to pleasing values as they are with the CDC Calcomp routines. This is because it is difficult to define pleasing and the author prefers to see the maximum and minimum values on the axis.

If the integer flag is used with the axis subroutine, then the min and max values returned by scale should be adjusted so that (max - min) is divisable by (size / ticdis).

NAME

site - change default line printer site

SYNOPSIS

From C:

```
char *s;

site(s);
```

From F77:

```
character s()

call site(s)
```

DESCRIPTION

This routine is used to change the default line printer site for *gplp (1g)*. The parameter, *s*, is a null terminated string of the form: "-XX", where XX is the name of the printer.

REMARKS

site must be called before the call to *plots*.

See *opr(1)* for more information about the line printer names.

The line printer must be a Printronix.

NAME

sline - draw a parametric line with on-center symbols

SYNOPSIS

From C:

```
float *x,*y,lx,ly;
int n,j,sym;

sline(x,y,n,lx,ly,j,sym);
```

From F77:

```
real x(),y(),lx,ly
integer n,j,sym

call sline(x,y,n,lx,ly,j,sym)
```

DESCRIPTION

To draw a line of a parametric relationship (x[i],y[i]), i=0,2,...,n-1, with on-center symbols plotted every jth point. The *i=n'th* and *i=n+1'th* elements of each array should represent the minimum and maximum values of the points in that vector. The user can call the routine *scale(3g)* to automatically find the minimum and maximum values. The following parameters are used:

x,y — Pointers to arrays containing the x and y coordinates of the parametric relationship (x[i],y[i]).

n — The number of points in the parametric relationship.

lx — The length (in plot units) that (xmax - xmin) is mapped to in page coordinates.

ly — The length (in plot units) that (ymax - ymin) is mapped to in page coordinates.

j — Plot the on-center symbol specified by 'sym' at every |j|th point. if j is negative, no line is drawn between on-center symbols.

sym — Specifies which on-center symbol is to be used. The following on-center symbols can be used.

sym	*on-center symbol*
0	no on-center symbol
i	$'i-1'i=1,2,...,10
11	$*
12	$+
13	$,

REMARKS

The Xi's and Yi's need not represent plot coordinates.

The minimun and maximum values for the arrays must be stored in the n+1st and n+2nd locations of the arrays respectively.

The point (x[i],y[i]) will be plotted at the page coordinates:

$$x = (x[i] - x[n]) / (x[n+1] - x[n]) * lx$$
$$y = (y[i] - y[n]) / (y[n+1] - y[n]) * ly$$

relative to the current origin. Hence depending upon the location of the current origin, the adjusted minima need not actually be the minima of the data which is to be plotted so long as each point (x[i],y[i]) has page coordinates which lie within the plotting area.

An error will occur if xmin = xmax or ymin = ymax.

NAME

speed - change plotting speed of the HP plotter

SYNOPSIS

From C:

```
int vel

speed(vel);
```

From F77:

```
integer vel

call speed(vel)
```

DESCRIPTION

The quality of a plot on an HP plotter is dependent on the quality of the pen and the speed at which it is moved across the paper. Generally as a pen ages and dries out a slower velocity can be used to maintain the quality of the plot.

The single parameter *speed* is the velocity of the pen in cm/sec. The valid range is [1,36]. The default velocity is 36 cm/sec.

REMARKS

Higher quality output is obtained at lower velocities.

NAME

symbol - add alphanumeric labelling to a plot

SYNOPSIS

From C:

```
float x,y,height,angle;
char *string;

symbol(x,y,height,string,angle);
```

From F77:

```
real x,y,height,angle
character string()

call symbol(x,y,height,string,angle)
```

DESCRIPTION

The *symbol* routine adds alphanumeric notation to a plot. The location, size, angle and string to be plotted are specified with the following parameters.

x,y	the coordinates of the point where the lower left corner of the first character is to be plotted.
height	the height (in plot units) of the alphanumeric character string.
string	a pointer to a zero terminated string.
angle	the angle (in degrees) counterclockwise from the +x direction at which the string is to be plotted.

REMARKS

The width of the characters is four-sevenths of height and they are spaced at intervals of six-sevenths of height. The space between characters is two-sevenths of height.

A second set of characters can be accessed by preceeding a valid character with a '$'. (A '$' can be obtained using '$$'.) The second font contains Greek letters and some special mathematical symbols. See *fontinfo(8g)* for more information about this alternate character font.

NAME

where - return current pen position

SYNOPSIS

From C:

```
float  x,y,sf;

where(&x,&y,&sf);
```

From F77:

```
real   x,y,sf

call where(x,y,sf)
```

DESCRIPTION

This routine aids in the optimization of plotting by returning the current pen coordinates and scale factor to the calling program. The current state is returned in the following variables:

x,y	variables for the return of the current pen coordinates relative to the current origin.
sf	variable for the return of the current scale factor.

REMARKS

Values of *x, y,* and *sf* when this subroutine is called are disregarded.

Subroutine *where* can be used, for example, to determine which direction a line between the points (x1,y1) and (x2,y2) should be drawn in order to reduce pen movement when the pen position is unknown.

NAME

examples - several examples using the CRC Grapics Package

EXAMPLE 1

The first example will demonstrate the use of Qplot using the output from a FORTRAN program compiled with 'f77'. Consider the following program:

```
      real y1(100),y2(100),x(100),y3(100)
      open(unit=2,file='y1',status='new',form='unformatted')
      open(unit=3,file='y2',status='new',form='unformatted')
      open(unit=7,file='y3',status='new',form='unformatted')
      open(unit=4,file='x',status='new',form='unformatted')
      do 10 i=1,100
      x(i) = (i - 1) * 2.0 * 3.14159 / 99.
      y1(i) = sin(x(i))
      y2(i) = cos(x(i)) * 2.0
      if(i .lt. 50)then
            y3(i) = i
      else
            y3(i) = 100 - i
      end if
10    continue
      write(2)y1
      write(3)y2
      write(7)y3
      write(4)x
      end
```

If the program is in a file 'test.f', it should be compiled with:

```
$f77. test.f
```

Execute the program with:

```
$a.out
```

The program will generate four binary files in F77 binary format. The file 'y1' will contain a sine wave. 'y2' will have a cosine in it. The file 'y3' will have a ramp in it. The last file, 'x', will contain the points in which the sine and cosine were generated from. To convert the f77 binary format to standard UNIX binary, the following commands should be run:

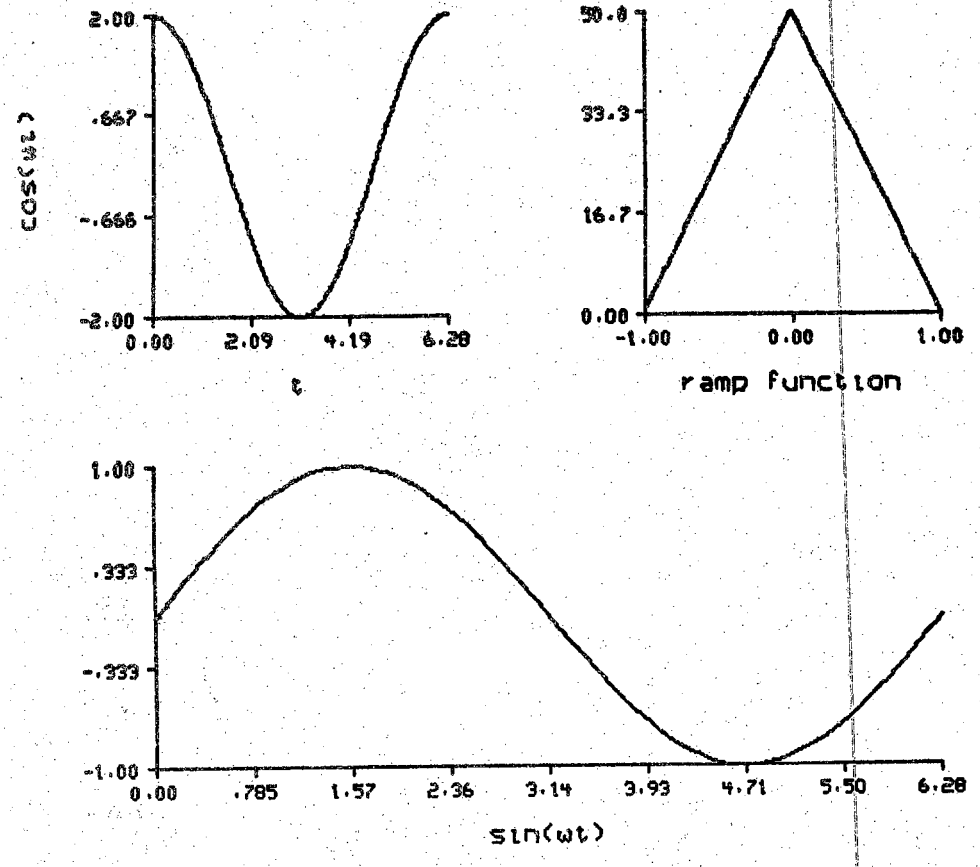
```
$strip7 y1 y2 y3 x
```

The following Qplot/Gplp sequence will plot the three curves on the Printronix line printer:

```
$qplot y=y1 x=x,4 digits=3 ylen=3.0  'xl=sin($wt)' g=g1
$qplot y=y2 x=,4 digits=3 ylen=3.0 xlen=3.0 yp=4.5 \
        'yl=cos($wt)' xl=t g=g2
$qplot y=y3 xmin=-1.0 xmax=1.0 len=3.0 yp=4.5 xp=5.0 digits=3 -r \
        xtic=1.5 "xl=ramp function" g=g3
$gplp g1 g2 g3
$rm g1 g2 g3
```

The output generated is listed on the following page.

Output from example 1



EXAMPLE 2

This example demonstrates a simple user generated graphics program in FORTRAN. Consider the following program which writes it output on the HP plotter:

```
call plots(7,0)
call plot(1.0,1.0,-3)
call plot(8.0,0.0,2)
call plot(8.0,8.0,1)
call plot(0.0,8.0,1)
call plot(0.0,0.0,1)
call symbol(.5,4.0,.3,"graphics",0.0)
call plot(0.0,0.0,999)
end
```

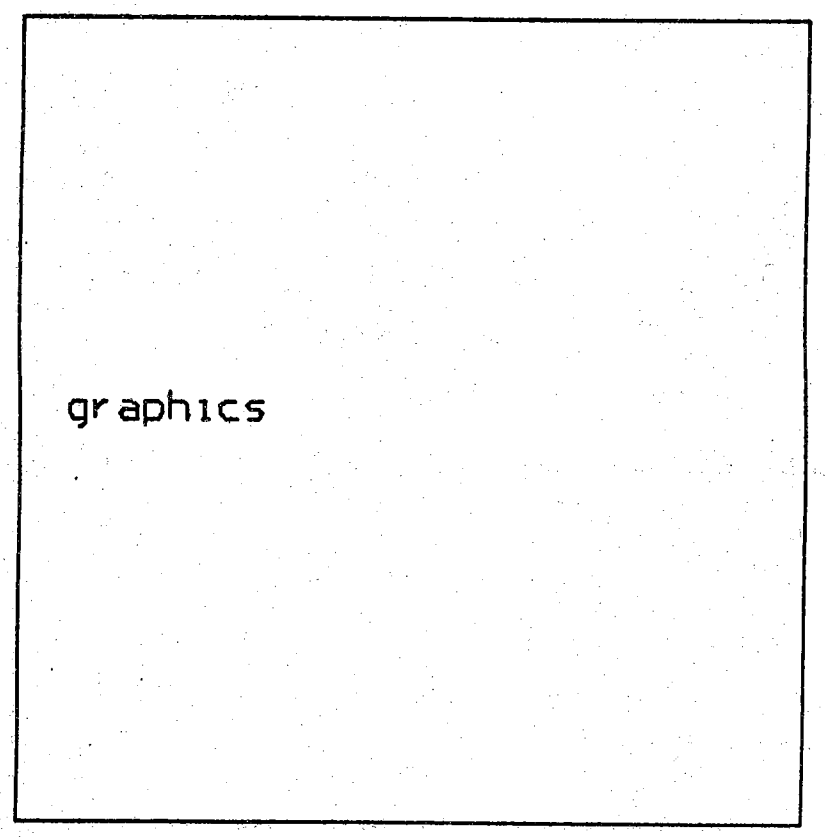
This program will plot a box with the word 'graphics' in the middle of it. It should be compiled with:

```
$f77 prog.f -i -lG
```

To obtain the output use:

```
$a.out
```

The output generated is:



EXAMPLE 3

This example will present a more complicated graphics program written in C. Consider the following program written in C:

```
#include        <math.h>

main(){
        int     i;
        float   x[631],y[631];

        plots(69,0,"ver");

        for(i=0;i<630;i++){   /* compute data */
                x[i] = cos(3.0 * i / 100) + 1.0;
                y[i] = sin(4.0 * i / 100);
        }

        scale(y,629);     /* get scaling information */
        scale(x,629);

        plot(1.5,1.0,-3); /* move origin away from corner */
        axisv(2.0,3); /* change default ticdis and digits */
        axis(0.,0.,"x axis",0,8.,x[629],x[630],0);
        axis(0.,0.,"y axis",1,8.,y[629],y[630],0);
        plot(0.0,8.0,3);  /* draw border */
        plot(8.0,8.0,2);
        plot(8.0,0.0,2);
        symbol(2.5,8.5,.2,"lissajous figure",0.0);
        line(x,y,629,0,8.0,8.0);

        plot(0.0,0.0,999);          /* terminate plotting */
}
```

This should be compiled with:

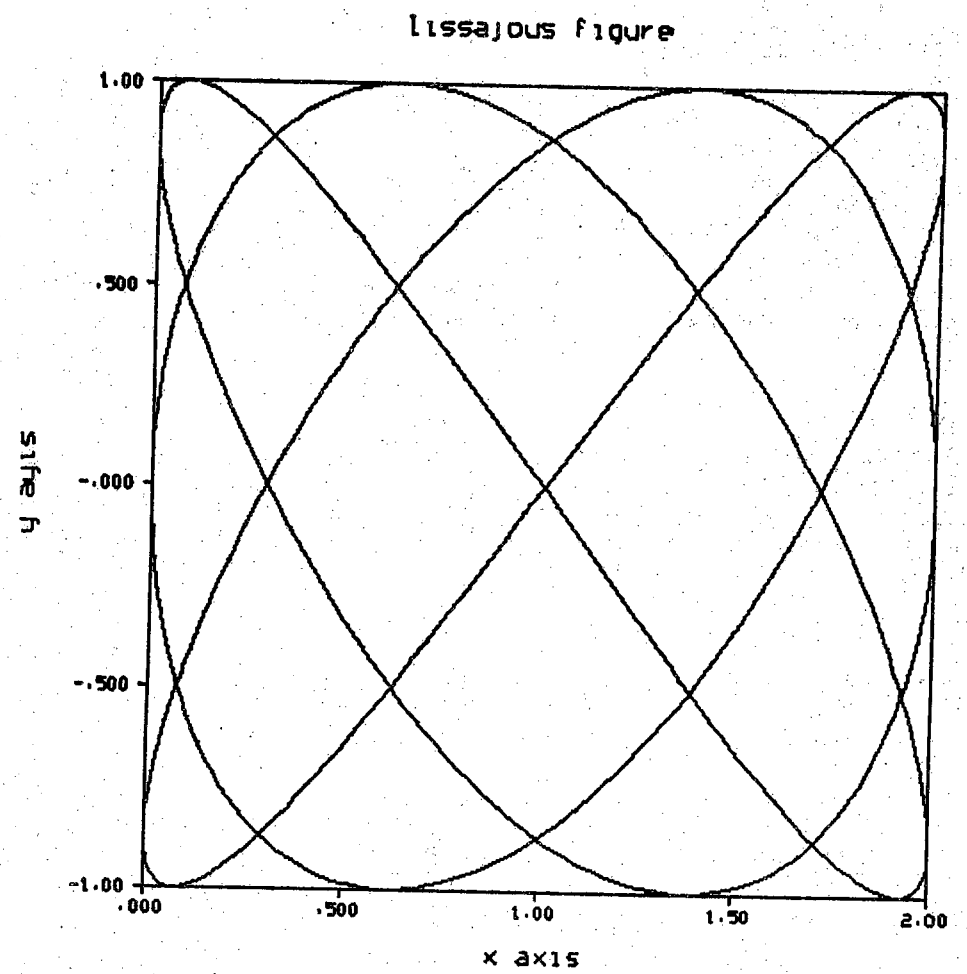
```
$cc. prog.c -i -lG -lm
```

To get the output on the Versatec, use:

```
$a.out
```

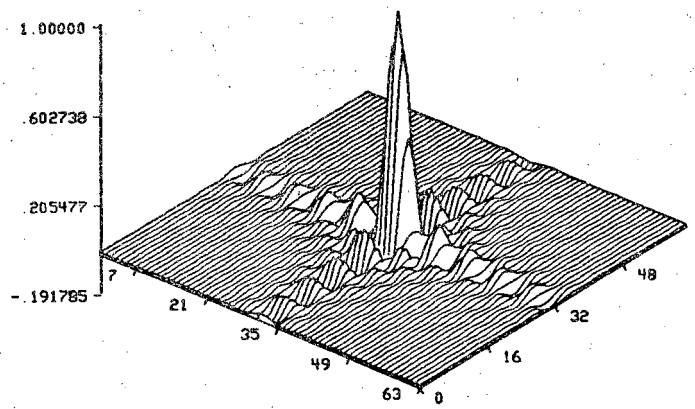
The graphics output is listed on the following page.

Output from example 3

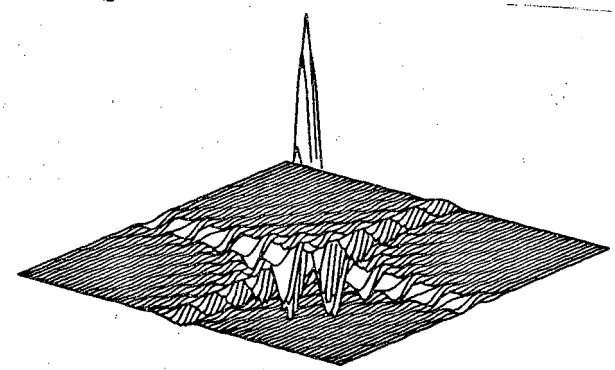


```
/*
 *      C Example 2 - Compute and plot a sinc(x)*sinc(y) function.
 *
 *      Compile with
 *              cc ex2.c -lm -o ex2
 *
 *      Run with
 *              ex2 > data
 */

#include        <stdio.h>
#include        <math.h>
#define N       64

main(){
        int     i, j;
        double  x, y, z, sinc();

        for (i=0;i<N;i++){                      /* For each x */
                x = i - N/2;
                for (j=0;j<N;j++){              /* and for each y */
                        y = j - N/2;
                        z = sinc(x)*sinc(y);
                        printf("%f\n",z);       /* Write out the value */
                }
        }

}

double sinc(x)                                  /* Compute the sinc(x) */
double  x;
{
        if (fabs(x) < .0001)
                return(1.0);
        else
                return(sin(x)/x);
}
```

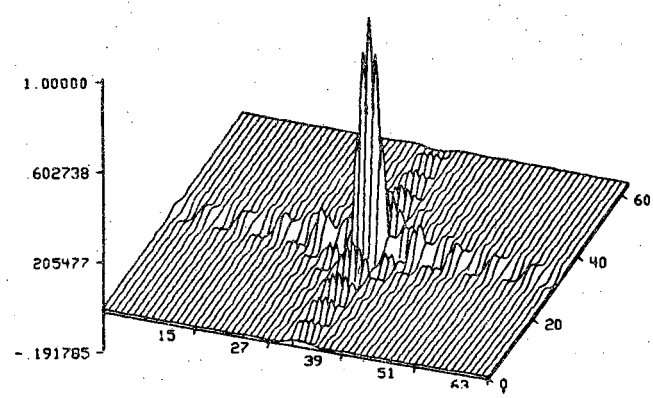
plot3d z=data,a



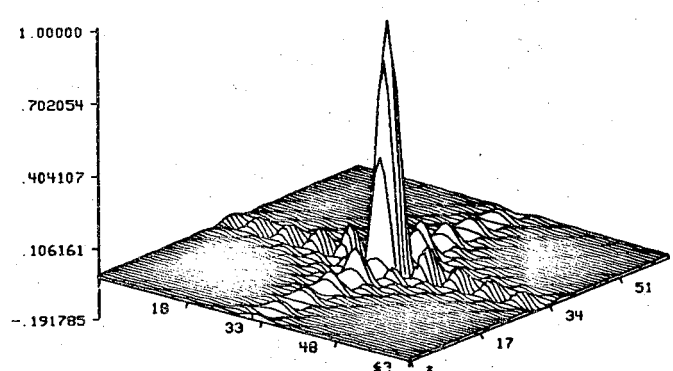
plot3d z=data,a -a -r phi2=20



plot3d z=data,a phi1=20

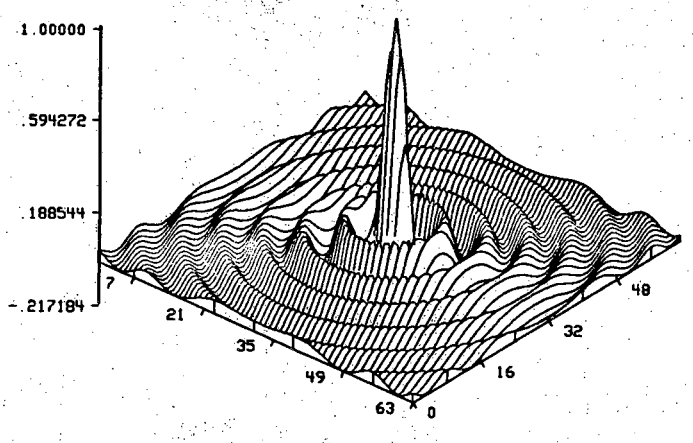


plot3d z=data,a phi2=20 direct=x

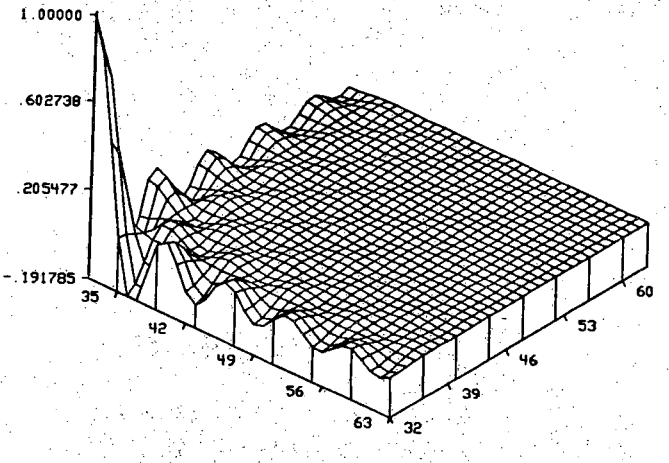


```
/*
 *      C Example 1 - Compute and plot a sinc(r) function.
 *
 *      Compile with
 *              cc ex1.c -lm -o ex1
 *
 *      Run with
 *              ex1
 *
 *      Look at the data by typing the command
 *              od -f data
 */

#include        <stdio.h>
#include        <math.h>
#define N       64

float   z[N][N];

main(){
        int     i, j;
        double  x, y, r;
        FILE    *output;

        output = fopen("data","w");     /* Open the output file */
        if (!output){                   /* And make sure the open succeeded */
                fprintf(stderr,"Can't open data file for output.\n");
                exit(1);
        }

        for (i=0;i<N;i++){              /* Increment the x-direction */
                x = i - N/2;
                for (j=0;j<N;j++){      /* Increment the y-direction */
                        y = j - N/2;
                        r = sqrt(x*x+y*y);
                        if ( r < .0001)
                                z[i][j] = 1.0;
                        else
                                z[i][j] = sin(r)/r;
                }
        }

                                        /* Write the data out in binary
                                         * format.
                                         */
        fwrite(z,sizeof(z[0][0]),N*N,output);
}
```

plot3d z=data

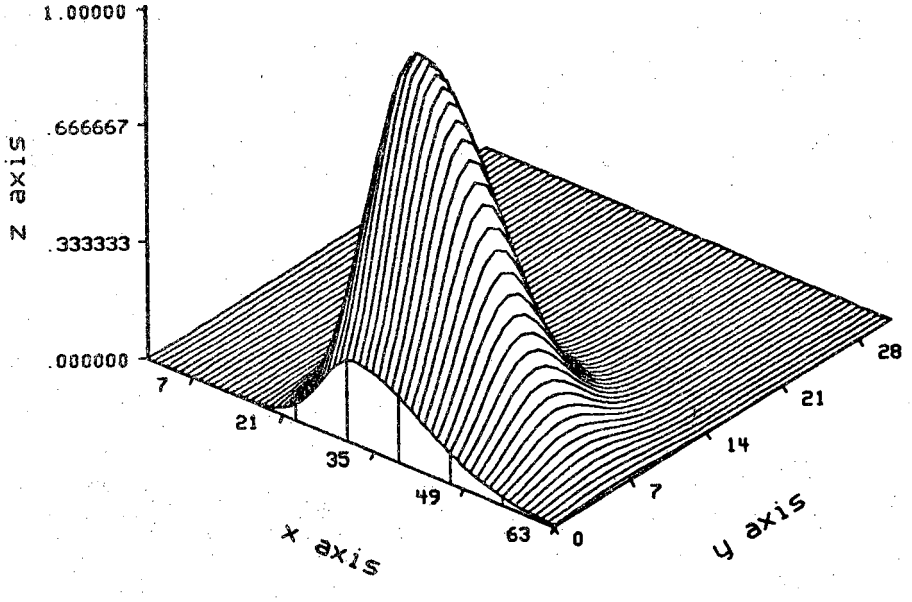


plot3d z=data begin=32 direct=xy

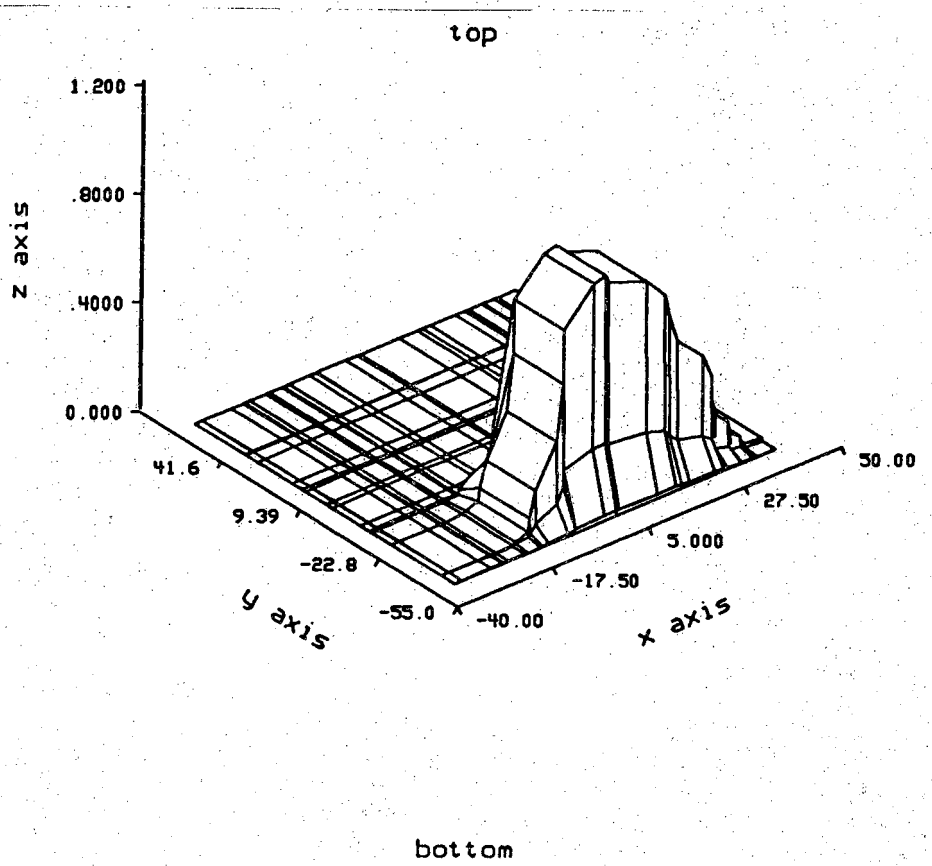


```
c
c       Plot3d Fortran Example - Compute and plot a semi-Gaussian function
c
c       Compile with
c               f77 ex.3 -lU77 -o ex3
c
c       Run with
c               ex3
c
c
c                               Set the different damping constants
c                               of the Gaussian function.
        sigx1   = 7.0
        sigx2   = 15.0
        sigy    = 5.0
c                               Open the z file.
        open(unit=2,file="z",status="unknown",form="formatted")
c                               The damping constant is different for
c                               negative and positive values of x while
c                               it is constant along the y axis.
c                               Compute the function.
        do 30 j=1 , 32
            yfact       = exp( - ( abs(j-7.0) / sigy ) ** 2 )
c                               Compute the first half of the Gaussian function.
            do 10 i=1 , 32
                tmp     = exp( - ( abs(i-33.0) / sigx1 ) ** 2 ) * yfact
                write(2,*)tmp
10          continue
c                               Compute the second half of the Gaussian
c                               function with a different damping constant.
            do 20 i=33 , 64
                tmp     = exp( - ( abs(i-33.0) / sigx2 ) ** 2 ) * yfact
                write(2,*)tmp
20          continue
30      continue
c                               Close the z file.
        close(2)
        stop
        end
```

plot3d z=z,a xsize=64 ysize=32 xl="x axis" yl="y axis" zl="z axis"

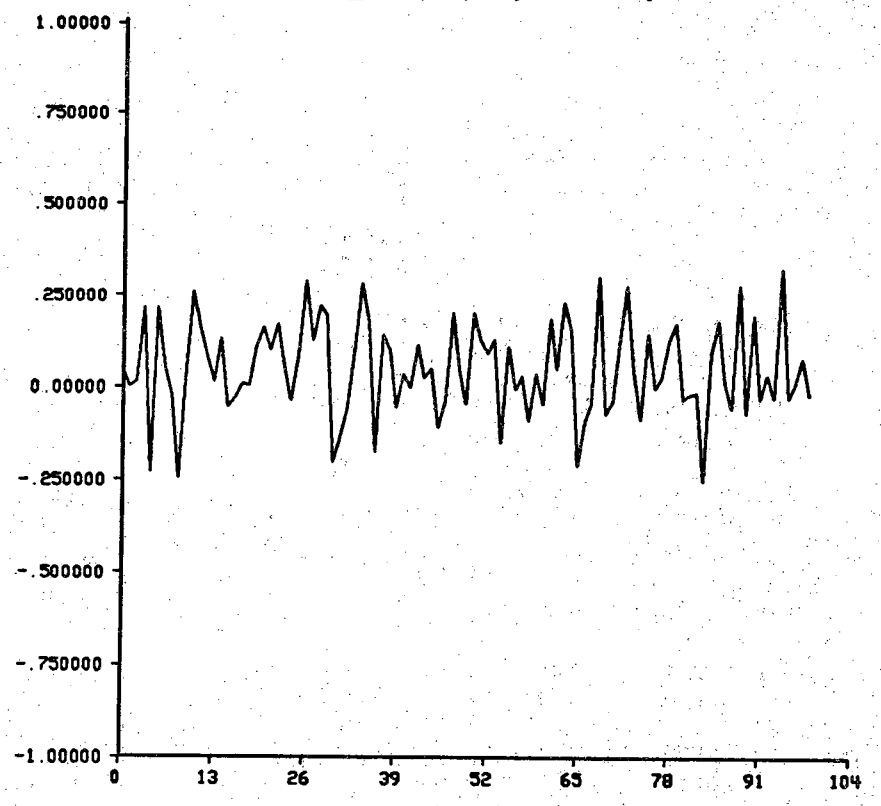


plot3d z=,f x=x,a y=y,a xsize=32 ysize=16 xmin=-40.0 xmax=50.0 ymin=-55.0 ymax=75.0 zmax=1.2 phi1=-40.0 dir=xy raxis=xyz res=3.0 xdig=4 ydig=3 zdig=4 xl=''x axis'' yl=''y axis'' zl=''z axis'' bl=''bottom'' tl=''top''

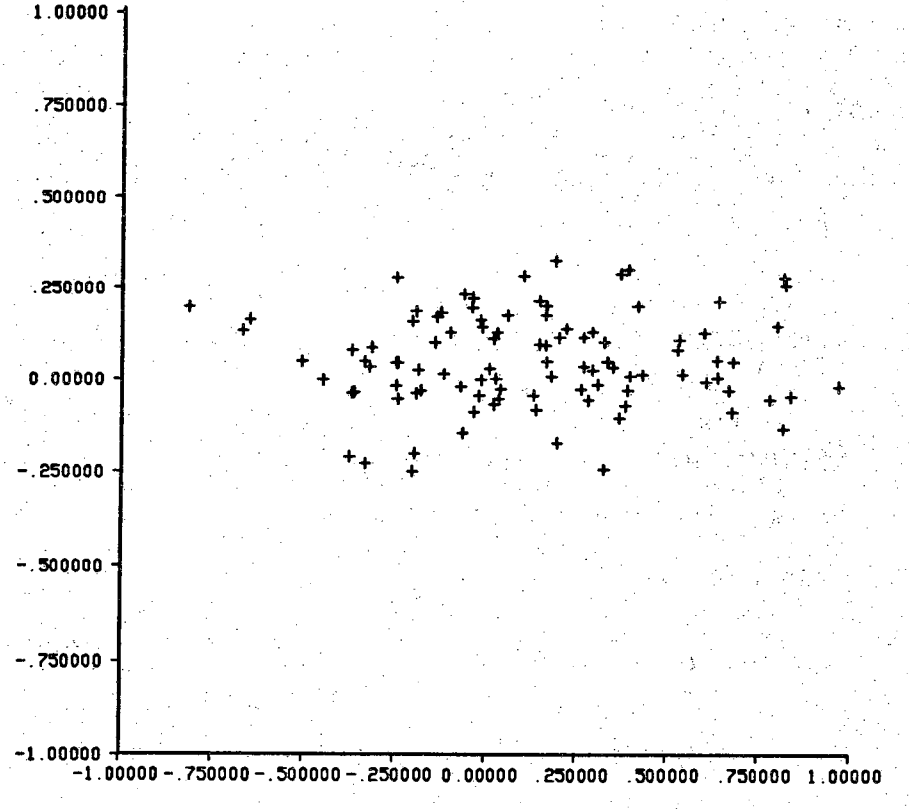


```
c
c       Plot3d Fortran Example - Compute and plot a semi-Gaussian function
c                                for non-uniform values of x and y.
c
c
c       Compile with
c              f77 ex.4 -lU77 -o ex4
c
c       Run with
c              ex4
c
        real    tmp(32)
        integer ucreat
                                The damping constant is different for
c                               negative and positive values of x while
c                               it is constant along the y axis.
c                               Set the different damping constants
c                               of the Gaussian function.
        sigx1   = 3.5
        sigx2   = 7.5
        sigy    = 12.5
c                               Create the z file.
        ifd     = ucreat('z',420)
c                               Compute the function.
        do 30 j=1 , 16
           yfact        = exp( - ( abs(j-3.5) / sigy ) ** 2 )
c                               Compute the first half of the Gaussian function.
           do 10 i=1 , 16
              tmp(i)    = exp( - ( abs(i-16.5) / sigx1 ) ** 2 ) * yfact
10         continue
c                               Compute the second half of the Gaussian
c                               function with a different damping constant.
           do 20 i=17 , 32
              tmp(i)    = exp( - ( abs(i-16.5) / sigx2 ) ** 2 ) * yfact
20         continue
           call uwrite(ifd,tmp,4*32)
30      continue
c                               Open the file x.
        open(unit=2,file='x',status='unknown',form='formatted')
c                               Compute the values of x.
        x       = - 10.0 * 16.0 * 0.5 * 0.5
c                               To make the sampling non-uniform,
c                               use the random function generater.
        do 40 i=1 , 32
           x    = x + 10.0 * rand(13*i) * rand(17*i)
           write(2,*)x
40      continue
c                               Close the file x.
        close(2)
c                               Open the file x.
        open(unit=3,file='y',status='unknown',form='formatted')
c                               Compute the values of y.
        y       = - 30.0 * 8.0 * 0.5 * 0.5
c                               To make the sampling non-uniform,
c                               use the random function generater.
        do 50 i=1 , 16
           y    = y + 30.0 * rand(13*i) * rand(17*i)
           write(3,*)y
50      continue
c                               Close the file y.
        close(3)
        stop
        end
```

qplot y=y,a ymin=-1 ymax=1



qplot y=y,a x=x,a sym=12 j=-1 xmin=-1 xmax=1 ymin=-1 ymax=1



```
/*
 *      Qplot C Example - Compute and plot a Gaussian Random Variable
 *
 *      Compile with
 *              cc ex3.c -lm -o ex3
 *
 *      Run with
 *              ex3
 */

#include        <stdio.h>
#include        <math.h>

main(){
        int     i;
        double  x, y, Gauss();
        FILE    *xfile, *yfile;

        xfile = fopen("x","w");                 /* Open the x and y files */
        yfile = fopen("y","w");

        if (!xfile || !yfile){                  /* Check for Errors */
                printf("Can't open files for output.\n");
                exit(1);
        }

        for (i=0;i<100;i++){                    /* Now compute 100 RV's */
                x = Gauss()*3.0;
                y = Gauss();
                                                /* Check for out of bounds */
                if (x < -1 || x > 1 || y < -1 || y > 1)
                        continue;
                fprintf(xfile,"%f\n",x);        /* Print out the values */
                fprintf(yfile,"%f\n",y);
        }
        fclose(xfile);                          /* Close the files */
        fclose(yfile);
}

#define NUM     25

                                                /*
                                                 * Compute a Gaussian random
                                                 * variable by summing a number
                                                 * of uniformly distributed
                                                 * variables.
                                                 *
                                                 * The returned value will
                                                 * have a mean of 0.
                                                 */
double
Gauss(){
        int     i;
        float   x;

        x = 0;
        for (i=0;i<NUM;i++)
                x += (float)random();

                                                /*
                                                 * Scale the sum by the
                                                 * maximum value from the
                                                 * random() subroutine and
                                                 * the number of RV's summed.
                                                 */
        return(x/((float)0x7fffffff*NUM/2) - 1.0);
}
```

NAME

Character Font Information

DESCRIPTION

The graphics package draws all of its own symbols. This allows the user great flexibility in the size and type of characters. The default character set is shown as an ADM-3A keyboard on the next page. The left hand side of each key is the character you would normally see. If the character on the left is preceded by a '$' then the character on the right is displayed. (Online a list follows. The first column is the normal character set and the second column is the alternate font.)

The 'open sup' and 'open sub' characters cause an effective half line shift up and down, respectively. The 'close sup' and 'close sub' negate the effects of the respective 'open' commands. The 'bs' character will back up one character. This allows for over-written characters. The '$:' character is a combination of a 'bs' and an overbar and the '$;' character is a combination of a 'bs' and an underbar.

The '$1' through '$0', '$,', '$*', and '$+' characters are all on-center symbols used for marking points in space. The rest of the characters assume that the lower left hand corner is the start of the character.

The actual character descriptions are the file:

/usr/lib/graphics/font.5x7.

The routine *symbol (3g)* defaults to this file. The user can supply his/her own font file by using the 'fontint' subroutine call.

A program called *genfont* is available to generate the font file from a set of ascii vector coordinates.

ALTERNATE FONT CORRESPONDENCE AND EXAMPLES

EXAMPLES :

To print	$A\underline{x} = \underline{b}$	enter	Ax$;$ =$ b$;
To print	I found $100!	enter	I found $$100!
To print	$\epsilon_o = 8.85 \times 10^{-12}$ F/m	enter	e[o$] = 8.85 x 10$(-12$) F/m
To print	$\sqrt{25} \doteq 4.99999$	enter	$!2$:5$: $# 4.99999
To print	$\alpha \equiv \beta * \lambda^2$	enter	$a $= $b*$1$(2$)