THE UNIVERSITY OF MICHIGAN

COMPUTING RESEARCH LABORATORY[1]

THE CRC PLOTTING PACKAGE

Edward Delp

CRL-TR-14-83

MARCH 1983

Room 1079, East Engineering Building
Ann Arbor, Michigan 48109
USA
Tel: (313) 763-8000

---

[1]Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors.

engr
UMR 1014

The CRC Plotting Package

August 1980
Modified for CIPRNET May 1981
Modified March 1983

The CRC Plotting Package is a fairly device independent graphics system intended to be used internally from FORTRAN or C programs or externally with binary vectors. Three distinct modes of plotting are available. The first two modes are plotting directly on Tektronix 4010 (or 4014) displays and HP 7221A flat-bed plotters. The third mode is achieved by plotting the desired graphics in an internal 512 by 512 bit plane. This bit map can then be copied to one of the following devices: Ramtek graphics overlay, Ramtek image display, Benson-Varian electrostatic printer, or a Printronix line printer. The package will write directly on all the devices with the exception of the Benson-Varian and the Printronix. In the case of the later devices, the bit plane can be written into an internal UNIX file and then GP or GPLP can be used to obtain outputs on the Versatec and Printronix, respectively. The following list indicates the devices currently supported.

DEVICE	STATUS	RESOLUTION	DISPLAY SIZE
Ramtek	OP	512 × 512	10 × 10
Benson-Varian	OP	512 × 512	10 × 10
Printronix	OP	512 × 512	10 × 10
HP 7221A	Non-OP	10240 × 7800	13.1 × 10
Tektronix 4014	OP	1024 × 780	13.1 × 10
Tektronix 4010	OP	1024 × 780	13.1 × 10

Where the resolution field indicates the number of points available in the display area. The units of the display area are arbitrary and vary from device to device. They indicate the size of the plotting area in plot units.

The user has two ways to obtain graphic outputs. If only vector versus vector plots on labeled axes are desired, then QPLOT will be sufficient in most cases. QPLOT is a program independent and needs binary vectors as its only input. The alternative is to write a graphics program using the plot subroutines callable from FORTRAN (with the F77 compiler) or C. To compile your program use the following commands:

From F77:
```
    %f77 prog.f  -lP
```
From C:
```
    %cc prog.c  -lP -lm
```

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

1) Documentation of Qplot , Strip , Gp , and Gplp
2) Documentation of user callable subroutines
3) Character Fonts
4) Examples

The documentation information for these programs is stored online and can be accessed by using the following command:

```
%help graphics
```

Up to date information and comments about the graphics package can be seen by using the following command:

```
%help graphics/news
```

These programs were written at Purdue University by Carl R. Crawford (that's where the name CRC comes from). The programs were modified for CIPRNET by Dave Olander. These programs were obtained from Purdue University under a licensing agreement with CIPRNET. The agreement states that the source programs and executable binaries are not be copied without the *written* permission of Prof. Edward J. Delp. Anyone having any questions about these programs or the licensing agreement should direct them to Prof. Edward J. Delp (login "ed"), Department of Electrical and Computer Engineering, Room 1520 East Engineering. Phone 764-2380.

PROGRAM 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 Ramtek's graphic overlay zero. A set of binary floating point numbers (4 bytes / number) will be read from a file called 'y' and used as the y vector. This will be displayed against a program generated x vector containing integers from zero to the number of points in the file minus one. The folowing options are available to modify the graph:

y=file	The y vector will be read from 'file' instead of 'y'. The number of bytes per vector element is set to n where n can be: 8 (double precision floating point), 4 (single precision floating point), 2 (short integer), 1 (character), 1s (signed characters), l (long integer). The default value of n is 4. If n isn't specified, a comma is not needed after the file name. See the appendix for more information on the byte declarations.
y=,n	The y vector is read from the default file 'y' but the number of bytes per element is set to n.
count=n	Only up to n points will be read from the input file. A maximum of 512 points can be plotted. The default value is 512.
begin=n	The first n points in the file will be skipped and then the number of points indicated by 'count' will be read. The starting point on the program generated x vector will be set to 'begin'. The default value is zero.
skip=n	Only every (n+1)'th point will be read from the input file. The program generated x vector will be incremented by 'skip' + 1. The default value is zero.
-x	The x vector will be read from a file called 'x' instead of qplot generating it for you. The default file type is integer.
x=file,n	The x vector is read from 'file' instead of 'x'. The number of bytes per vector element is set to n. The default value of n is 2. Note that this option invokes the -x flag.

x=,n	The x vector is read from the default input file 'x' but the number of bytes per element is set to n.
s=file	Normally the program generates the scale values for each vector by finding the minimum and maximum values of the vector. This method assumes that the graph will fill the display device. If 'file' exists, the minimum and maximum values used by the scaling routines will be read from 'file'. The format is xmin, xmax, ymin, ymax. The x values are n1 bytes and the y values are n2 bytes where n1 is from the x=file,n1 command and n2 is from the y=file,n2 command. If single byte data is used, the scale values should be short integers instead of single bytes. The xmin and xmax values will be skipped if the -x flag isn't used. The default name for 'file' is 'xybnd'.
ymin=r	The ymin value read from the scale file or obtained from the data is replaced by r.
ymax=r	Same as the ymin = option except works with the ymax value.
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. When the program generates the x vector if the 'xmin=r1', 'xmax=r2', and '-r' options are set, the x vector will be a set of equally spaced floating point numbers between r1 and r2.
-r	A floating point x vector will be generated if the -x option isn't used and the 'xmin=' and 'xmax=' options are used.
-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 graphs with the same scale. The file has the same format as the input scale file.
scfac=r	The graph is expanded or reduced in size by r. The default value is 1.0.
xp=r	The x starting coordinate of the graph is moved from the default origin of zero (bottom left corner of the display) to r. Where r is given plot units. The display is assumed to be ten units square and the axes are eight units long starting at x=1.5 and y=1.5.

yp=r	Same as the xp=r flag but the y origin is moved to r instead.
xl=string	The character string is used as a label below the x axis. If a character within the string is preceded with a '$', then an alternate character set will be used.
yl=string	The character string is used as a label next to the y axis.
-a	No axis will be plotted.
-z	A bar graph is made instead of connecting vector elements with a straight line.
-n	The plot is displayed on the Ramtek's graphic overlay n. Where n can be either 0 or 1. The default value is 0.
-c	The plot is displayed on a Ramtek image instead of a graphic overlay.
-g	Display the plot on a Ramtek graphic overlay. This flag is set by default.
-t	Display the plot on a Tektronix 4010 or 4014 display.
-h	Display the plot on the 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.
-b	The display device is not blanked before plotting. This has no effect on the HP plotter.
-p	The plot is written out to a file 'graph' instead of the Ramtek. This file can then be written out to the Benson-Varian using GP or to the Printronix line printer using GPLP.
-o	Same as -p except that the graph is written out to standard output. This is helpful if the '-i' option is used with GP and GPLP.
g=file	The file that the plot is saved in using the -p flag is changed to 'file'.
-d	A dashed line is used instead of a straight line when drawing the graph.
dash=r	

When using the -d flag r specifies the length of each dash.

gap=r	When using the -d flag, r specifies the length of gaps between dashes.
-f	A border will be generated around the plot.
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	Sets both axes length to r.
digits=n	The number of significant digits used in the axis annotations will be set to 'n'. The default value is six significant digits.
xdigits=n	Sets the number of significant digits in the x axis to n.
ydigits=n	Sets the number of significant digits in the y axis to n.
tic=r	The distance between tic marks on the axes will be changed from the default one plot unit to r units.
xtic=r	Set only the x tic mark distance.
ytic=r	Set only the y tic mark distance.
el="string"	The string will be plotted just to the right of the last point in the line.
-m	Mark every point in the line with an on-center symbol.
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.

MESSAGES & DIAGNOSTICS

bad flag: -c

The character 'c' is not recognized as a valid flag.

bad option: string

The character string is not recognized as a valid option.

byte declaration error	The number of bytes per vector element is not valid.
attempt to plot outside the picture	When plotting on the Ramtek or in a file, this message will occur when the plot goes outside the plotting area.
scale values identical	The xmin and xmax values or ymin and ymax values are identical and the scaling routines don't know how to scale the data.

SEE ALSO

gp
gplp
graphics/examples
strip7

FILES USED

/dev/image	Ramtek image
/dev/gov?	Ramtek Graphic overlays
/dev/plt0	HP plotter
x	Default x input array
y	Default y input array
graph	Default plot store file
xybnd	Default scale file
/usr/lib/plot/font.5x7	Character font

KNOWN BUGS

The x and y vectors can't have a single element.

If the plot size is reduced too much, the lettering on the axis will be unreadable.

If the plot is expanded too much on the Tektronix or the HP, the graph will wrap back around on itself.

AUTHOR

Carl Crawford (See Prof. Delp)

LOCATION OF SOURCE

See Prof. Delp

APPENDIX: Input file conventions

As mentioned before, Qplot expects the 'x' and 'y' files to be in a binary format. This format will be explained in detail in this section for the benefit of users who have had little or no experience with using 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.

Internally a program assigns a certain number of bytes to each variable used. There are two types of variables: fixed precision and floating point. Within the UNIX system fixed point numbers can have one, two, or four bytes. Floating point numbers can have four or eight bytes. When a formatted write is initiated, the number is converted from the internal binary representation to a string containing displayable ASCII characters. In most situations it will take far more space (one character takes up one byte to store) to store a variable in its ASCII format than it would to store it directly in its binary format. The alternative is to use unformatted writes to store the data in its internal binary representation.

When transferring data from one program to another, it is absurd to waste the time converting the data to an ASCII format and then convert it back to internal binary. This statement is also applicable to storing data files on a disc. The file will be a lot smaller if the data is in a binary format. If there is a need to see the data, a small program can be written to convert the binary file to a ASCII file.

The above discussion presented the overview for the reasons why binary files are expected. In a later section unformatted writes will be discused.

Qplot has adopted a notation to indicate what kind of binary data the 'x' and 'y' files contain. The byte declaration field follows a comma after the file name. The following table contains the list of valid byte declarations:

1:	Single byte, unsigned, fixed precision data
1s:	Single byte, signed, fixed precision data
2:	Two byte, signed, fixed precision data
l:	Four byte, signed, fixed precision data
4:	Four byte, floating point data
8:	Eight byte, floating point data

Each programming language's internal variable types map into the byte declaration types qplot expects. The mapping depends also on the compiler and machine (VAX UNIX versus non-VAX UNIX). 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: (Vax)

TYPE	DECLARATION
int	1
short	2
float	4
char	1 or 1s
double	8

FORTRAN: (F77)

TYPE	DECLARATION
integer	1
integer*2	2
real	4
double	8
character*1	1 or 1s

Unformatted writes can be obtained using the 'fwrite' subroutine in the standard i/o package in C. See "The C Programming Language" for more information.

The easisest way to get unformatted writes from FORTRAN is to just write the data without a 'format' statment. An example would be:

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

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

In this example two files will be generated. They will be named '2' and '3' if F77 was the compiler. The FORTRAN compiler also generates code to produce record counts in their binary files. The command 'strip7' will remove the record counts from binary files generated with F77. The output file names can be changed using the FORTRAN subroutine call 'open'. See the compiler manuals for more information.

NAME

strip7

SYNOPSIS

strip7 file

DESCRIPTION

Strip7 remove the byte counts that f77 put into it's binary output files respectively. The 'stripped' file replaces the original file only if no errors occur while stripping the file and the the user has write permission for the file.

DIAGNOSTICS

can't open : file	file cannot be opened for reading
can't create: file	can't create tmp file

FILES

strip7.tmp	temp file for strip7

AUTHOR

Carl Crawford (See Prof. Delp)

NAME

gp - output 512 X 512 X 1 bit pictures to the Benson-Varian.

SYNOPSIS

gp [file]

DESCRIPTION

Gp copies Ramtek graphic overlay format files to the Benson-Varian plotter. This program is actually a shell that sets on top of bp(1). It is invoked differently than that of gplp. See bp(1) for more details.

DIAGNOSTICS

Error messages will occur if the input files can't be opened or the Benson-Varian is busy.

SEE ALSO

gplp
qplot
graphics/introduction
bp

NAME

gplp - output 512 X 512 X 1 bit pictures to a Printronix line printer

SYNOPSIS

gplp [-i] file1 ... filen

DESCRIPTION

Gplp copies Ramtek graphic overlay format files to the 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.

LOCATION

/user/ece/gplp	binary

AUTHOR

Carl Crawford
(See Prof. Delp)

SEE ALSO

qplot
gp
graphics

Graphics Library Information

The graphics library (/usr/local/lib/libP.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' can be called again. The library is broken up into three different types of routines. The first group supports the HP plotter and the Tektronix terminal.
The second group is link between the F77 callable routines and the C routines. This group is used only by the loader.
The last group and the only one really accessible by the user are the high level plotting routines. Below is a list of the available routines.
The documentation for each file follows and there are examples at the end of this document (graphics/examples).

plots	initializes the plotting devices
fname	set output file name when graphics device is a file
plot	moves pen and terminates plotting
line	plots a line through a set of coordinate pairs
dline	same as 'line' but plots with a dashed line
sline	same as 'line' but can put on-center symbols on the line
scale	computes scale values for 'line', 'dline', and 'sline'
axis	plots numerically annotated axes
laxis	plots numerically annotated log axes
axisv	set environment variables for 'axis' and 'laxis'
symbol	puts strings on the graph
number	provides formatted numeric labeling
factor	change the scale factor of the plot
where	returns current pen and scale values
newpen	changes the pen on the HP plotter

NAME

plots

PURPOSE

Plots is used to allocate buffers and initialize the devices before plotting can begin. This subroutine must be called before any other subroutines are called.

USAGE

FROM C:

```
short int    dev,blank;

plots(dev,blank);
```

FROM F77:

```
integer dev,blank

call plots(dev,blank)
```

DESCRIPTION OF PARAMETERS

dev	0: write graph in a file "graph" 1: plot on Ramtek graphics overlay #0 2: plot on Ramtek image 3: plot on the Tektronix 4: plot on the HP
blank	The specified device will not be blanked before plotting unless blank is zero.

REMARKS

Ramtek graphics overlay image #1 can be accessed by setting dev to 9.

The default file name, when dev equals zero, can be changed using the subroutine 'fname'.

The int declaration above in C must be "short".

NAME

fname

PURPOSE

To change the default file name when dev = 0.

Usage

FROM C:

```
char  *name;

fname(name);
```

FROM F77:

```
character    name()

call fname(name)
```

DESCRIPTION OF PARAMETERS

name	pointer to zero terminated string containing the name of the file that the final graphics output will be written into.

REMARKS

Fname should be called immediately after the call to 'plots'.

If the name is "-", then the graphics output will be written out to standard output. Pipe the output of your program into "gplp -i" or "gp" if output is desired on the Benson-Varian or Printronix.

Fname keeps a pointer to the name instead of copying the name into an internal buffer. The correct file will not be opened if 'fname' is called from another subroutine and 'name' is on the stack.

NAME

plot

PURPOSE

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

USAGE

FROM C:

```
float  x,y;
short int   i;
plot(x,y,i);
```

FROM F77:

```
real   x,y
integer i

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

DESCRIPTION OF PARAMETERS

x and y	the coordinates of the point 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'. The plot routines are terminated and the the plot is displayed on the Ramtek if that is the selected device, or the plot is written out to a file, or the plot buffers for the Tektronix or HP are flushed. An error will occur if subsequent plot calls are made prior to another call to 'plots'.

The int declaration above in C must be "short".

NAME

line

PURPOSE

To draw a line of a parametric relationship (x[i],y[i]), i=1,2,...n.

USAGE

FROM C:

```
float     *x,*y,lx,ly;
short 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 OF PARAMETERS

x and 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 that (xmax - xmin) is mapped to in page coordinates. Should be the same value used in the axis subroutine.
ly	the length that (ymax - ymin) is mapped to in page coordinates. Should be the value used in the axis subroutine.

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.

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 int declaration above in C must be "short".

NAME

dline

PURPOSE

To draw a plot of a parametric relationship (x[i],y[i]), i=1,2,...n, with a dashed line.

USAGE

FROM C:

```
float      *x,*y,*dsh,*gap,lx,ly;
short 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 OF PARAMETERS

x and 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 inches) that (xmax - xmin) is mapped to in page coordinates. Should be the same value used in the axis subroutine.
ly	the length (in inches) 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.

The int declaration above in C must be "short".

NAME

sline

PURPOSE

To draw a line of a parametric relationship (x[i],y[i]), i=1,2,...n with on-center symbols plotted every jth point.

Usage

FROM C:

```
float     *x,*y,lx,ly;
short 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 OF PARAMETERS

x and 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 that (xmax - xmin) is mapped to in page coordinates. Should be the same value used in the axis subroutine.
ly	The length that (ymax - ymin) is mapped to in page coordinates. Should be the value used in the axis subroutine.
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. See table below for symbol definitions.

REMARKS

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

The minimum 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.

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

The int declaration above in C must be "short".

NAME

scale

PURPOSE

To find the minimum and maximum values of a vector.

USAGE

FROM C:

```
float   *a;
short int    n;

scale(a,n);
```

FROM F77:

```
real    a()
integer n

call scale(a,n)
```

DESCRIPTION OF PARAMETERS

a	pointer to an array.
n	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].

The minimum and maximum values are NOT adjusted to pleasing values as they are with 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 divisible by (size / ticdis).

The int declaration above in C must be "short".

NAME

axis

PURPOSE

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

USAGE

FROM C:

```
float  x,y,size,min,max;
short 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 OF PARAMETERS

x and 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 axis at zero degrees, tic marks below the axis. 1: plot axis at ninety degrees, tic marks to the left of the axis. 2: plot axis at zero degrees, tic marks above the axis. 3: plot axis at ninety degrees, tic marks to the right of the axis.
size	the length 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) must be divisible by (size / ticdis).

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.

The int declaration above in C must be "short".

NAME

laxis

AUTHOR

Jeffery L. Gray (See Prof. Delp)

PURPOSE

To draw a labeled logarithmic coordinate axis with numerically annotated tic marks.

USAGE

FROM C:

```
float x,y,size;
short 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 OF PARAMETERS

x and 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.
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.

The int declaration above in C must be "short".

NAME

axisv

PURPOSE

To change the default distance between axis tic marks and the number of significant digits in the numerical annotations.

USAGE

FROM C:

```
float   ticdis;
short int   digits;

axisv(ticdis,digits);
```

FROM F77:

```
real   ticdis
integer   digits

call axisv(ticdis,digits)
```

DESCRIPTION OF PARAMETERS

ticdis	the distance 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 int declaration above in C must be "short".

NAME

symbol

PURPOSE

To provide for alphanumeric labeling.

USAGE

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 OF PARAMETERS

x and y	the coordinates of the point where the lower left corner of the first character is to be plotted.
height	the height 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 preceding a valid character with a '$'. A '$' can be obtained using '$$'. The second set contains the Greek letters and some special mathematical symbols.

NAME

number

PURPOSE

To provide numeric labeling on plots.

USAGE

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 OF PARAMETERS

x and y	the coordinates of the point where the lower left corner of the first character is to be plotted.
height	the height of the alphanumeric character string. If height is negative, num is to be plotted as an integer.
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. For example a valid format would be "x = %.3f" for a real number or "i = %d" for an integer.
num	the number whose value 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 preceding a valid character with a '$'. A '$' can be obtained using '$$'. The second set contains the Greek letters and some special mathematical symbols.

See the documentation for 'printf' for more information on the syntax of the format statement 'number' expects.

NAME

factor

PURPOSE

To provide for changing the scale factor, initially 1.0, for subsequent x and y coordinates.

Usage

FROM C:

```
float   sf;

factor(sf);
```

FROM F77:

```
real    sf

call factor(sf)
```

DESCRIPTION OF PARAMETERS

sf	the new scale factor to be used in scaling all subsequent x and y coordinates unless and until factor is called again.

REMARKS

If the scale factor sf has been specified, then the coordinates (x,y) are scaled to the point (sf*x,sf*y).

NAME

where

PURPOSE

To aid in optimization of plotting by returning the current pen coordinates and scale factor to the calling program.

USAGE

FROM C:

```
float   x,y,sf;

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

FROM F77:

```
real    x,y,sf

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

DESCRIPTION OF PARAMETERS

x and 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 subroutine where 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

newpen

PURPOSE

To deselect the current pen and select one of the other three pens in the HP plotter (this command is ignored if dev is not equal to 4).

USAGE

FROM C:

```
short int    n;

newpen(n);
```

FROM F77:

```
integer n

call newpen(n)
```

DESCRIPTION

n	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.

The int declaration above in C must be "short".

Character Font Information

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.

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-striked 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/plot/font.5x7.
```

The routine 'charfont' (described in graphics/charfont) defaults to this file. The user can supply his/her own font file by using the 'fontint' subroutine call. A program is available to generate the font file from a set of ascii vector coordinates. On the VAX machine it is in:

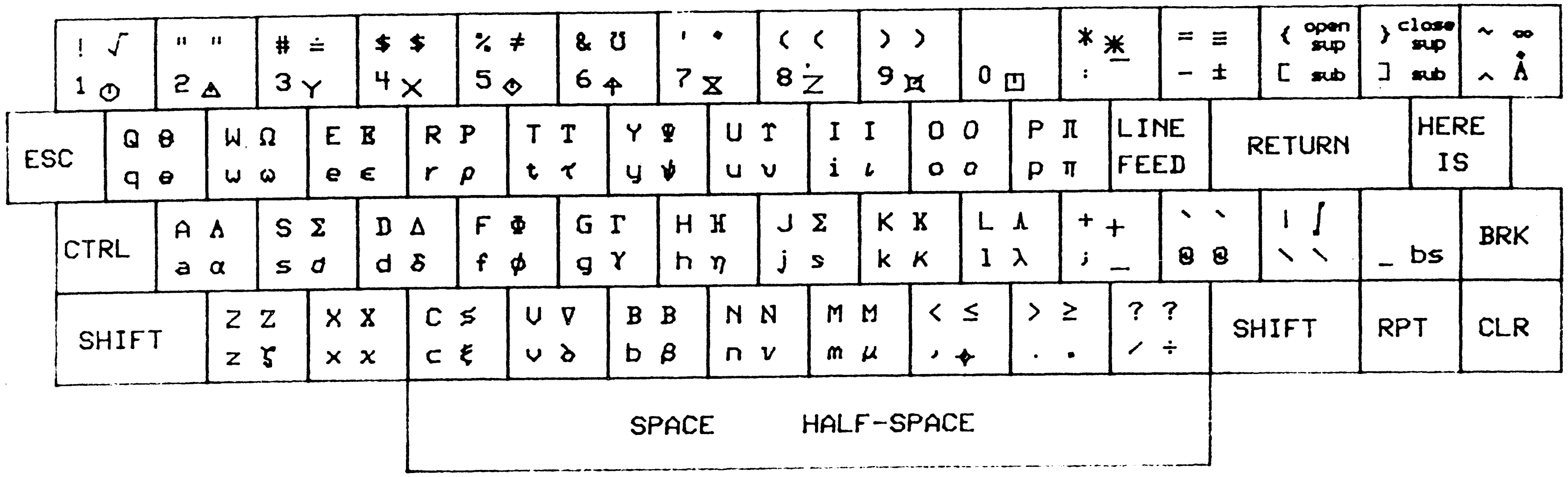
```
/ece/plot/genfont.c
```

The input to this program used to generate the default character font is on the VAX machine in:

```
/usr/lib/plot/ifont.5x7
```

The user should look at this file for the format used as input to genfont.

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$)

NAME

charfont

AUTHOR

W. C. Zurney (see Prof. Delp)

PURPOSE

Lookup routine for producing graphic characters.

USAGE

```
char        symbol
double       height, *x, *yd
short int    *visflg,charfont()

charfont(symbol,height,x,y,visflg)
```

DESCRIPTION OF PARAMETERS

symbol	Character to be produced
height	Height of the requested character
x	Next relative x coordinate.
y	Next relative y coordinate.
visflg	1=visible line 0=invisible line
charfont()	0 = this is the last coordinate pair 1 = there are more coordinate pairs

REMARKS

Charfont is a program that looks up the end points of line segments for plotting characters. Each time 'charfont' is called, it returns the relative coordinates, which when added to the current location yields the final end point for a line segment. The initial point of the line segment is the current location. Whether the line segment is to be visible or not is indicated by the returned value in "visflg".

Use 'fontinit("file")' to select the file containing the font information. The default font file is:

/usr/lib/plot/font.5x7,

which contains the ascii and alternate character sets.

The font file used by 'fontinit' to read in the data for 'charfont' has the following format:

int height	Default character height.

```
int  size       Bytes of core required to hold coordinates.
int  pnt[512]   Indexes to 1st coordinate of each symbol.
int  crd[size]  Coordinates of symbols
```

Where each crd[i] has the following format:

```
EVSXXXXXXXSYYYYYYY
III       IIIIIIII
III       I I---- Y coordinate (sign magnitude format)
IIIIIIIII
II  I----- X coordinate (sign magnitude format)
II----- Line segment visible flag (0=invisible, 1=visible)
I---- 1=more coordinates; 0=last coordinate
```

The int declaration above in C must be "short".

EXAMPLE 1

This section will contain examples of the usage of the graphics package. 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 was in a file 'test.f', it should be compiled with:

```
%f77 test.f
```

Run 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' wil 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
%strip7 y3
%strip7 y2
%strip7 x
```

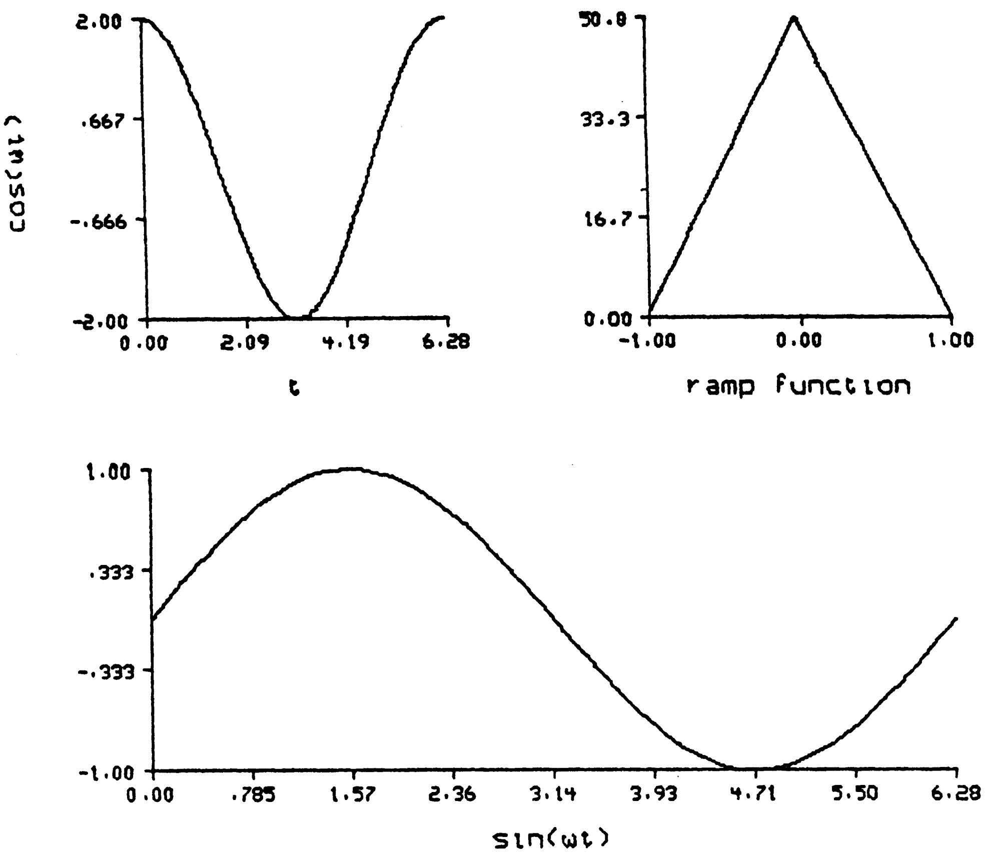
The following 'qplot' and 'gplp' sequence will plot the three curves on the Printronix:

```
%qplot y=y1 x=x,4 digits=3 ylen=3.0  -p 'xl=sin($wt)' g=g1
%qplot y=y2 x=,4 digits=3 ylen=3.0 xlen=3.0 yp=4.5 -p \
        'yl=cos($wt)' xl=t g=g2
```

```
%qplot y=y3 xmin=-1.0 xmax=1.0 len=3.0 yp=4.5 xp=5.0 -p 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:

```
call plots(0,0)
call fname("-")
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
```

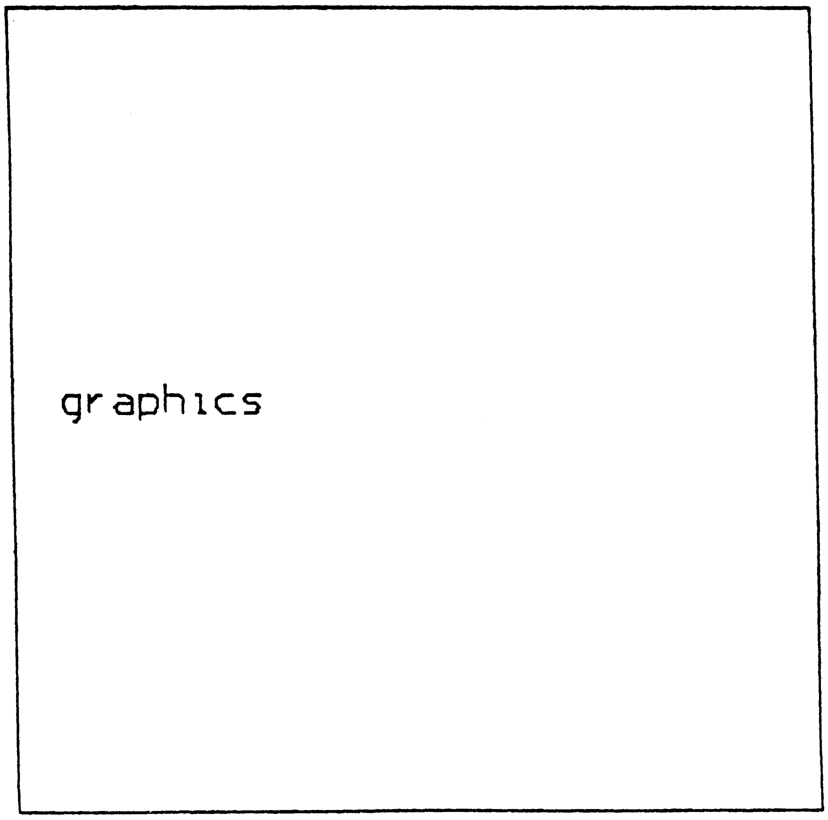
This program will plot a box with the word 'graphics' in the middle of it. It should be compiled with:

```
%f77 prog.f  -lP
```

To obtain the output use:

```
%a.out | gplp -i
```

The output generated would be:



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(0,0);
        fname("-");    /* output graph on standard output */

        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 ayis",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:

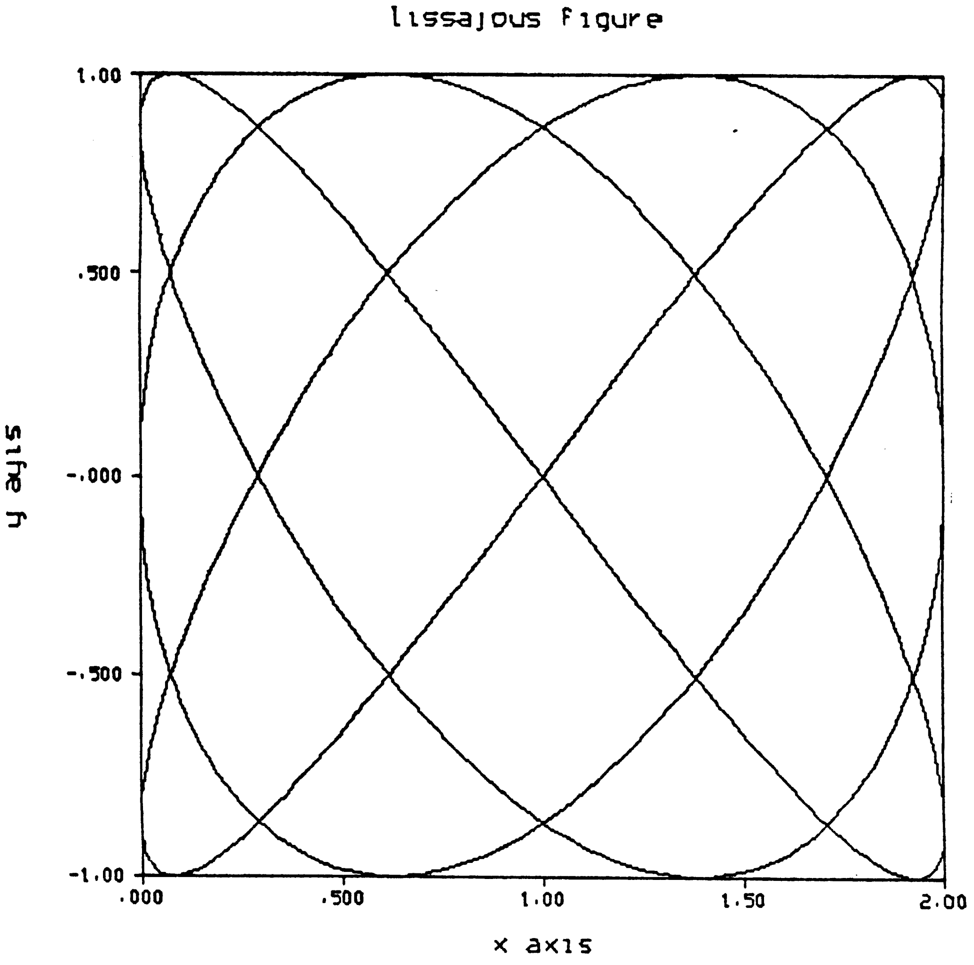
```
%cc. prog.c  -lP -lm
```

This program will send its output to standard output. To get the output on the Printronix, use:

```
%a.out | gplp -i
```

The graphics output is listed on the following page.

Output from example 3


lissajous figure
x axis
y axis