PLOTXY: A VERSATILE PLOT PROGRAM

Robert Parker and Loren Shure

(Modifications by Nessan Fitzmaurice)

Plotxy is a program for generating graphs from data files with a minimum amount of fuss. In the simplest case, data points have been generated as x-y pairs in an ASCII file called *xydata* perhaps written with a formatted FORTRAN write statement or created with an editor. The program reads the file, and plots y as a function of x, interpolating with straight lines; axes are automatically assigned with reasonable limits and annotations. All this can be done with the three commands

```
read
plot
stop
```

Here a data file with the name *xydata* has been read format free (that is with * instead of a format in FORTRAN). An output file in *Postscript* named *plotxy.ps* has been constructed; it may be plotted on any *PostScript* device or previewed under *SunView* using the *plotpage* program. As documented in the rest of this writeup, it is possible to add embellishments of considerable complexity: for example many series on one plot, plotting of symbols at the points, cubic spline interpolation of continuous curves, logarithmic scales, error bars, titles, axis labels and all the other common desiderata. The program operates in a command mode; this means that you are not prompted at the console, but instead you type simple commands instructing the program which options are needed. Almost every option has a default value, so that if nothing is mentioned about a particular parameter the default is taken: for example the default plotting scales are linear in x and y. Once a particular option has been invoked, it remains in force until altered. The advantages of this approach are that great flexibility is available but a minimum of typing is required and no rigid order is demanded of the typist. The last factor makes the program very easy to use in batch mode, in contrast with prompting systems where one must remember the questions and the branches taken by the program during execution.

Upon execution the program prints the single prompt:

```
Enter read and plot commands
```

Now you must type commands selected from the catalog below; each command begins in the 1st column of a new line; it may be followed by some literal or numerical parameters, which must be separated from the command word by a space. Any command may be abbreviated by its first four characters. To read data from an external disk file there are a number of commands that define the attributes of the data to be input; an obvious example is the name of the disk file, defined by **file**; other input attributes are things like whether this data set is to be connected with a smooth interpolating curve (**smooth**) or to be plotted as individual points (**symbol**). Having set up all the necessary specifications of the input, you actually perform the reading with the **read** command. Another data series may be read from the same or a different file, simply by resetting the input attributes as required, and using **read** again. All the parameters remain in force from the previous read unless they are specifically altered. Each of these data series with its different properties is accumulated in memory ready to be plotted. Plotting is accomplished with **plot**. If several data series have been read since the previous call to **plot**, or this is the first such call, they all appear on one graph. Actually all that is done by this command is to create the *plotxy.ps* which must be displayed with one of the system routines as we shall discuss in a moment. There are a number of parameters that apply to the whole plot, which may be set before invoking **plot**. For example, there are the x and y axis labels (**xlabel**, **ylabel**) and the plot title (**title**). The size of

the plot will default to 6 inches in x and 8 in y; the limits of the plot will default to values slightly larger than the extremes found in the series. These things can be overridden using **xlimit** and **ylimit**. When all the data are strictly positive, logarithmic scales can be set with **logxy**. Before typing **plot** it is sometimes helpful to inspect the series and the way in which the data are going to be interpreted. This is done with **status**, which provides a synopsis of each data series and the current parameter settings. Additional graphs may be created by reading in more data and invoking **plot** again as often as necessary. To terminate the program just type **stop**. After you have entered **stop**, the program writes a diskfile called *plotxy.ps*, or something else if you used **output**, that must be sent to graphics device.

The commands are given their full English names here (though only four characters are needed). Commands may be given in upper or lower case letters. The characters following the first blank after the command word itself are termed the 'command field'. A command line may be 120 characters long. Parameters that may be omitted in the command field are enclosed in square brackets. The parameters may be numbers, file names or text; it should be obvious from the context which is appropriate. A list separated by slashes denotes a set of possible alternative items. A command line beginning with a space is ignored and may be used as a comment. In mathematical formulas an asterisk denotes multiplication and double asterisk exponentiation.

affine a b c d Transforms the x and y coordinates of the next and subsequent data series to be read according to new(x) = a∗x + b, new(y) = c∗y + d. This is an affine transformation.
Blank command field a = 1, b = 0, c = 1, d = 0

cancel [n] Removes the last n data series read into memory. If no series has been read in, do nothing.
Blank command field n = 1

character h [angle] Change the height of the lettering in titles, labels and axis numbering to h inches. The new value applies to the next piece of text to be read, so that different height letters can appear in the title, the axis notations etc. The value of h just before **plot** determines the height of the axis numerals; if this is zero axis numerals and tick marks are suppressed. Negative h has the same effect as positive, except that axis numerals are suppressed, without removing the tick marks. The optional parameter angle specifies the angle at which text will be plotted in the next **note**. Letter height can also be controlled by the a special text phrase (see LETTERING).
Default h = 0.15, angle = 0

color n On those plotters capable of drawing in color, this command sets the color of subsequent items according to the code: 0 or 1 black, 2 red, 3 blue, 4 green, 5 brown, 6 orange, 7 purple, 8 yellow. For this to work the pens must be in a standard order in the magazine of the penplotter and of course only the first 4 work with four-color plotters. The color of a plotted data series is the one set at the time of the associated **read** command; similarly with labels, notes, etc. Also the axes and the frame are drawn in the color specified at the time **plot** is called.
Default n = 0

dash [s1 s2] Plot the next data series to be read in as a dashed line, with visible segments s1 inches long and missing segments s2 inches long. **dash** is an input attribute applying to the next **read** command. To return to an unbroken curve, set s1 or s2 to zero.
Default s1 = 0, s2 = 0
Blank command field s1 = 0.2, s2 = 0.1

file filename Defines the file name of the external disk file from which data are to be read; or the symbol ∗ which implies read from the console. The name must consist of 64 or fewer characters. After this command the next **read** statement will begin at the beginning of the file.
Default = *xydata*
Blank command field = Rewind existing file

format (format specifier) Defines a format for reading the next data series from an external disk file. The format specifier may be (1) a normal FORTRAN format specifier enclosed in parentheses (2) the single character ∗ meaning 'format free' reading (3) the character *b* meaning a binary read. In each case the data are read with a single FORTRAN read statement of the appropriate type. Usually, if the numbers can be unambiguously read by a person without the need to skip certain columns or other tricks, there is no need to use an explicit format – the default ∗ will work. Never use an I format because values are stored as REAL variables; thus a number written with I4 must be read with an F4.0 format. Always remember the space after the word **format** and the parentheses.
Default = ∗
Blank command field = ∗

frame *on/grid/off/none* If *on* causes two more sides to be added to the axes to complete a rectangular frame around the plot; *grid* includes a lightly dashed grid that corresponds to the axis tick marks as well as a frame; *off* cancels the surrounding box and returns to the normal situation with two orthogonal axes. Finally, *none* specifies the total absence of axes and frame.
Default *off*
Blank command field *on*

help Lists the four-letter abbreviation of all the commands. This may remind you of a name you have forgotten.

logxy [n] Specifies the type of scales for the next plot. The integer n may be 0, 1, 2, 3, meaning: 0 both x and y are linear variables; 1 implies that x is logarithmic; 2 means only y is logarithmic; 3 means both x and y are logarithmic.
Default n = 1
Blank command field n = 3

mode n [x0 dx] Defines how the input data are grouped in the next and subsequent reads. The integer n may be 1, 2, 3, –3, 4 or 20: 1 implies data are simply consecutive y values with uniformly increasing x values beginning at x = x0 increment dx; 2 means data are x y pairs; 3 means data are x y z triples in which z is taken to be the uncertainty in y (thus a bar is plotted between y–z and y+z). –3 means use third member of data group as an uncertainty in the x-value. A symbol may be plotted at the actual value of y itself if **symbol** is set; to guarantee the absence of a symbol put s = 0 in **symbol**. When **mode** is 4 the x and y data must be read by separate **read** commands, the x series being input first. The series length is that of the x series. 20 is like **mode** 2 but allows you read two columns from a table in an ASCII file without constructing a **format** for the purpose. An example of standard usage is **mode** 20 5 2 which means column 5 is used for x and column 2 for y in the x-y series. In this mode any supplied **format** is ignored and replaced by ∗. Chaos results if there are too few numbers in any row of the table.
Default n = 2
Blank field after n = 1, x0 = 1, dx = 1
Blank field after n = 20, x0 = 1, dx = 2

note (x y [*in*]) text
note (p q x y [*in*]) text Reads the characters of text to be plotted on the graph at the coordinates x, y. The text may be up to 80 characters in length. If the optional *in* appears the coordinates refer to the bottom left corner of the first text character, measured in inches from the intersection of the axes; otherwise x, y are in the units of the graph. Notice that the parentheses surrounding the coordinates are mandatory. The height of the plotted characters is the value h in the most recent **character** command. Similarly the angle the text makes with horizontal is the one previously set in **character**. Up to 20 separate notes may be input. To clear the notes enter **note** and a completely blank command field (this means omitting the coordinates, of course); this must be done explicitly for new graphs to begin a fresh set of notes. If four coordinates instead of two appear in the parentheses, an arrow is drawn with its tip at p, q and its tail tastefully near the text, which is plotted as before with x, y at the bottom left of the first character. The text is always horizontal with this option. If no text accompanies a four-coordinate note, the arrow tail goes exactly to the second position.
Blank field = delete all old notes

offset [dx dy] When several similar data series are to be displayed together it is often convenient to introduce a displacement between them to clarify the picture. Thus if dy is nonzero at plotting time, the nth series will be plotted with values of y + (n–1)∗dy, where y is the input value. Similarly with dx. When a logarithmic scale is used, the data are plotted as y∗10**((n-1)∗dy) to preserve equal apparent displacement on the graph. Notice all the series are displaced whenever **offset** is invoked; for a more flexible method of offsetting data see **affine**.
Default dx = 0, dy = 0
Blank command field dx = 0, dy = 0

output filename Defines the name of the plot file to be filename, which must be composed of 64 or fewer characters. Every time **output** is issued, the currently opened plot file is closed and a new one opened ready to receive further plots.
Default = *plotxy.ps*

plot [x0 y0] Creates the next complete graph containing all the data series that are currently in memory. Unless a **save** command has been used, the plotted series are the ones read in since the last **plot** command or, if this is the first such command, all the series. A *PostScript* file is generated named *plotxy.ps*, unless you have set a different name with **output**. Usually **plot** is invoked with a blank command field but, if x0 and y0 are specified, the new graph is plotted with its origin at those coordinates in inches relative to the previous plot origin. The plot origin for these purposes is the place where the annotated axes cross, not the point (0, 0).

read [n] Performs the reading of the external disk file according to the specifications in force at this point. Each **read** instruction is performed with a single FORTRAN READ statement with an implied DO; this means many data may appear on a single line in ASCII files. With binary files only one binary record is read with every **read** command. The integer n is the number of points to be read from the file, but if n is absent the file is read to the end of file (eof). When the eof is not reached, the file remains open and ready for further reading beginning at the next unread record (i.e. the next line in ASCII files); if the eof was reached, another read on this file will begin at the beginning. Up to 100 separate data series may be present at any one time. Note n must be explicit if **file** = ∗

save After a **plot** command, the series in memory are normally erased ready for new data. To prevent this, the command **save** must be entered before the next **read** statement. If no additional data are to be read in, there is no need to use **save** because the previous data are available for

plotting in this case.

smooth [*on*/*off*] Decides whether continuous curves of y against x are interpolated with straight lines, **smooth** *off*, or natural cubic splines, **smooth** *on*. When splines are used, the series is taken to be a single-valued function of x and the actual x values will be re-ordered to be increasing by the program if necessary. This command is an input attribute, applying to subsequent **read** commands, not to the whole plot. **smooth** *on* cancels a **symbol** command and vice versa. **smooth** *off* reverts to **symbol** mode if that was the previous style of plotting, with the same symbol number and height as before. Note that the automatic plot limits use the original data series, not the smoothed values, so that sometimes pieces of a **smooth**ed curve may be lost off the top or bottom of a graph even when you have let the program find its own limits.
Default = *off*
Blank command field = *on*

skip [n] Skips the next n records in the current data file. With ASCII files this means skipping n lines. The command examines the current read **format** to determine whether the current file is binary or ASCII. Although skipping can be performed by including slashes in a format specification it is often more convenient to use **skip**.
Blank command field n = 1

stack Causes the next complete graph, including axes, titles etc, to be drawn above the previous one with enough space to give a pleasing appearance. To stack several curves on one graph see **affine** or **offset**. This command is turned off internally after **plot** to prevent accidental plotting off the top of the paper. A more flexible way of organizing the relative positions of several complete graphs is by means of **plot** with origin parameters.

This command may no longer work --- use the **plot** *command.*

status Lists a synopsis of the current data series (their lengths, extreme values and other attributes), the plot and reading parameters, and the number of words available for further data series.

stop Closes the output file and brings program to an orderly halt. This must always be the last command of any run, otherwise part of your plot will be lost.

symbol n s Defines the next input series to be a set of discrete points with symbols rather than a curve. The height of symbol is s inches, the type of symbol defined by the integer n:

0 square	8 upward arrow	16 small circle
1 triangle	9 hourglass	17 circle
2 octagon	10 campstool	18 large circle
3 diamond	11 hexagon	19 small filled disk
4 plus	12 Y	20 small filled square
5 asterisk	13 vertical bar	21 small filled triangle
6 cross	14 star of David	
7 slashed square	15 dot	

To cause one of these symbols to be drawn in a text string (a **note** for example) just enclose the symbol number plus 2000 in backslashes, for example, \2019\. The value of n may be used to reset the input of continuous data: n = –1 means next data read will be continuous with straight line interpolation; n = –2 means go to cubic-splined curves.

Default n = –1, s = 0

title text Specifies a title for the plot. This may be up to 115 characters in length. A blank command field cancels the previous title and leaves the next plot untitled. The character font of the title is assumed by all the lettering of the graph unless explicitly reset. If the title consists only of a font-setting phrase, the font is set and the graph is untitled.
Default text = blanks

xlabel text Specifies a label to be written under the x axis. See **title** for other details.
Default text = blanks

xlimit xlength [x1 x2] Defines the length of the x axis, xlength, in inches and the lower and upper limits of x: x1, x2. All plotted points will have values inside (x1, x2); those outside are omitted from the plot. If x1 = x2 = 0, or if these values are omitted in the command, the x extremes will be chosen to encompass the values in the data series. If x2 is less than x1 the data and axes are plotted reversed, that is with x decreasing to the right, between the given limits. Reversed logarithmic axes are not permitted. This is a plot attribute, governing the behavior when **plot** is invoked.
Default xlength = 6, x1 = x2 = 0

ylabel text Same as **xlabel** but for the y axis.
Default text = blanks

ylimit ylength y1 y2 Same as **xlimit** but for the y axis.
Default ylength = 8, y1 = y2 = 0

affine	1, 0, 1, 0	**output**	*plotxy.ps*
character	0.15 0	**smooth**	*off*
dash	0, 0	**symbol**	–1, 0
file	*xydata*	**title**	blanks
format	*	**xlabel**	blanks
frame	*off*	**xlimit**	6, 0, 0
logxy	0	**ylabel**	blanks
mode	2	**ylimit**	8, 0, 0
offset	0, 0		

Plotxy provides a variety of fonts in which the titles, labels, notes may be written as well as the ability to include mathematical material and Greek letters. The names of the fonts are *simplex, complex, italic, duplex*; the default is the austere *simplex*. To get any of the others in a text string enclose the first three letters of the font name in backslashes (e.g. \ita\ or \dup\) ahead of the text. The font remains in force until explicitly changed. To obtain a uniform font throughout the graph and its labels include a font-setting phrase (e.g.\ita\) at the beginning of the title. If you want to vary the fonts within one plot you must specify the desired font for each character string plotted. Font changes may appear at any point in a piece of text. You may also get Greek letters by enclosing their names in backslashes, as \GAMMA\ or \lambda\; upper case Greek appears when the English name is upper case. The name of a Greek letter can be abbreviated to its fewest unambiguous leading letters: thus \s\ specifies sigma, but you need \ome\ for omega. Superscripts are possible with the construct \sup{...}, so that x-squared is rendered x\sup{2}. Similarly with subscripts one writes, for example, g\sub{ij}. As mentioned in **symbol** you may

plot a special graphics symbol by enclosing the symbol integer plus 2000 in backslashes. The code \bs\ suppresses the character advance so that characters may be superimposed. There are certain special characters that have no keyboard equivalent. To get them you must use a special code: a 4-digit key number enclosed in backslashes. The code acts just like an ordinary character so that, for example, the space of infinitely differentiable functions would be written C\sup{\1395\} since \1395\ is the code for infinity. Here is a table of the codes for the special symbols; every symbol in the graphics character set has such a code but only 1387 to 1431 are given because all those without keyboard equivalents are contained in this list.

1387 curly d	1403 summation
1388 del	1404 regular theta
1389 member of	1405 {
1390 less or equal	1406 }
1391 greater or equal	1407 @
1392 proportional	1408 hat
1393 integral	1409 [
1394 circuit int	1410]
1395 infinity	1411 #
1396 + or –	1412 paragraph
1397 – or +	1413 dagger
1398 times	1425 tall <
1399 division	1426 tall >
1400 product	1429 degree
1401 times dot	1430 tends to
1402 radical	1431 regular phi

Finally, another use for the four-digit code is to specify text size. The height of the text following the phrase \0025\ is changed from its current value to 0.25 inches; any number less than 500 is interpreted as the letter height in hundredths of an inch, but remember there must be four digits between the backslashes.

Plotxy is reasonably graceful with error conditions: explanatory messages are issued in most circumstances. For example, if unintelligible data are encountered, the program reject the whole series and then issue a warning. Attempting to plot negative data on a log scale will not cause a crash – an error message is printed and the offending scale is made linear instead of a logarithmic. If you have explicitly set plot extremes with **xlimit** and **ylimit** and a data value falls outside the window, when **smooth** is *on* the program draws a piece of curve between the last captured point(s) and the edge of the graph in the direction of the invisible point. When **smooth** is *off* the pen is simply lifted until onscale data are encountered. This allows you to insert breaks in your data records by using large values; remember to set limits explicitly at plot time. A useful idea for creating elaborate graphs that overcomes the apparent restrictions on the number of notes and the kinds of axes available is the superposition of several graphs on top of each other: **plot** 0 0 draws the next graph right over the old one. Thus if you only want an x axis, plot the data with **frame** *none*, then set the length of the y axis to zero and plot a new x axis with **plot** 0 0. To plot a line with symbols at the observations just read the data file twice, once with an interpolating line (**smooth** or not, as desired) and then with **symbol**. If you want lines with both long and short dashes this too can be done by reading the data twice each time with different, but compatible dash sizes, so that when superposed the desired effect is achieved. This is possible because *plotxy* is careful about dashed lines, always starting at the beginning of a drawn section. Not every combination of long and short dashes can be built this way, although a sufficient variety can. For example, if you use

```
dash  a b
read
dash  c d
read
```

then one way to obtain a repeating pattern is to set c=(2∗a–(n–1)∗b)/(n+1) and d=(n∗b–a)/(n+1), where n is an integer such that a/b < n < 1+2∗a/b. Artistic users seem to want to vary the font and size of every notation and label. This is straightforward: any piece of text may be preceded by a font phrase; the height may be defined in the text as described in the previous section or by the **character** command immediately before the text is entered (and similarly with **color**). The only writing that can not be specified in this way is the numerical annotation of the axes. To control its size and color set these parameters immediately before **plot**; to arrange a special font put the font phrase you desire *at the end of* the title text. If you want the two axes in different colors with different numeral sizes and fonts, this can be done too, but discovering how is left as an exercise.

Here are two examples of quite presentable plots made with relatively little effort. The first is a reproduction of some graphs of Bessel functions found in Abramowitz & Stegun's Handbook of Mathematical functions. A rather sparse table has been entered equally spaced directly into the input file and the values are spline smoothed to add authority. All modern operating systems allow you to prepare an input file and submit it to a program as if it were entered interactively; it is very handy then to put the data in the file together with the plot commands. Notice that comments have inserted by beginning a command line with a blank.

```
   A diagram from Chapter 9 of Handbook of Mathematical
   Functions
file *
smooth
mode 1 0 1
   First 16 values of J0 for x=0, 1, 2, .. 15
read 16
1.000  .762  .224 -.260 -.397 -.178  .151  .300  .172 -.090
-.246 -.171  .048  .207  .171 -.014
   Now 16 values of J1
read 16
0.000  .440  .577  .339 -.066 -.328 -.277 -.005  .235  .245
 .043 -.177 -.233 -.070  .133  .205
   Next 15 values of Y0 for x=1, 2, .. 15  dashed
mode 1 1 1
dash .05 .07
read 15
 .088  .510  .377 -.017 -.309 -.288 -.026  .224  .250  .056
-.169 -.225 -.078  .127  .205
   Finally values of Y1
read 15
-.781 -.107  .325  .398  .148 -.175 -.303 -.158  .104  .249
 .164 -.057 -.210 -.167  .021

xlim 3.5 0 16
ylim 3 0 0

title \dup\
xlab \com\FIGURE 9.1: \ita\J\sub{0}(x), Y\sub{0}(x),
note (1,0.8)J\sub{0}
```

```
note (1.8,-.5)Y\sub{1}
note (2,.577,3,.8)J\sub{1}
note (3,.377,5,.6)Y\sub{0}

plot 1 7
```

In the above listing the **xlab** command line has been truncated to fit on the page. Several different fonts have been used; notice that the axis numerals are in the font of the title, which is in fact blank. The picture has been placed at the top of the page leaving room for the second example below it. Next we illustrate how a disk file may be rescanned with a format to pick out different columns for various purposes. The data file *rhodata* contains a table of Wenner array apparent resistivity data at various electrode spacings, together with an estimated uncertainty (column 3) and the fit of a one-dimensional theoretical model (column 4). The file is read in **mode** 3 to get error bars, then reread and smoothed to put the theory on the graph. Log axes are appropriate here for obvious reasons.

```
title \ita\
logxy 3
frame

xlim 4 1.0 1000
ylim 3.6 10 2000
ylabel Apparent resistivity  \rho\\sub{a} (\OME\m)
xlabel Electrode separation  r  (meters)

   1st read uses the format to pick measurements and
   errors in mode 3
file rhodata
symbol 19 0.1
mode 3
format (3f11.0)
read
   2nd read picks up 1st and 4th column with mode 20
mode 20 1 4
dash 0 0
smooth
read

notes
plot
stop
```

Here is the data file *rhodata*

1.52	69.6	10.0	67.9
4.57	126.3	21.1	126.3
7.62	207.6	33.6	199.6
12.1	304.3	47.0	304.2
18.2	421.2	40.2	431.0
24.3	508.7	69.2	541.8
30.4	587.6	52.0	636.4
42.6	769.8	83.0	778.2
79.2	987.7	122.0	913.2
106.7	902.3	150.1	848.7
137.2	691.2	114.1	715.2
167.6	543.3	86.16	571.0
228.6	366.7	101.2	333.1

On the *ICOMP* Suns under UNIX, if you have /u/lerc/fsnssan/bin as one of your paths, *plotxy* is an executable command. The plot filter that causes the PostScript output file (usually *plotxy.ps*) to be displayed is called *plotpage*.

Some sample data files for plotxy and their resulting plot and PostScript files are available in /u/lerc/fsnssan/graphics/plotxy/sample. There is also a sample awk script which selects specific columns and rescales part of the data. (NB: *plotxy* can also selectively read columns, but has no provion for rescaling the input data). The awk script is named "awk.cmd".

INDEX

y data only **mode**