PLOTXY: A VERSATILE PLOT PROGRAM

Robert Parker and Loren Shure

INTRODUCTION

Plotxy is a flexible, command-line 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*. 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. For example:

```
file xydata
read
plot
stop
```

Here the data file has been read to the end. A PostScript file named *mypost* has been written, which may be displayed to the on terminal with *ghostscript* (gv) or sent to a printer for hardcopy. All printers at IGPP (UCSD) are PostScript printers. The *–watch* option in gv means that you need not restart the display to see updates as you develop a complicated graph. *Plotxy* allows you to produce a simple diagnostic diagram, or publication-quality graphics: there are logarithmic scales, error bars, axis labels, the Greek alphabet and every other element needed for scientific graphs.

Plotxy operates from the command line in a simple language. You are not prompted, but instead you enter instructions to indicate which options are needed. Almost every option has a default value, so that if nothing is mentioned about a particular parameter a sensible 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. (Exception: **fill**).

BASICS

Upon execution *plotxy* prints the prompt:

```
Enter commands for graph 1
```

Now you must type commands selected from the catalog below; each command begins in the 1st position of a new line; it may be followed by some text or numerical parameters, which must be separated from the command word by a space. Any command may be abbreviated by its first four characters. Alternatively, particularly with complex graphs, you should prepare a file of commands to run by redirecting the standard input as follows:

```
plotxy < myfile
```

where *myfile* contains the list of commands for *plotxy*.

To read data from a file there are several commands defining 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, you perform the actual input with the **read** command. A second data series may be read by using **read** again. The numbers may come from the same file or, by resetting the input file name, a different one. All the parameters remain in force from the previous read unless they are specifically altered. Each of the data series with its different properties is accumulated in memory ready to be plotted.

Plotting is accomplished with **plot**. If this is the first call to **plot**, all data series read up to this point will appear on one graph. Actually this command merely creates the

file *mypost*, which must be displayed or sent to a printer after *plotxy* has halted. The -w option of the PostScript viewer *gv* is very handy during graph development; this redisplays the PostScript file whenever it is rewritten. Some parameters apply to the whole plot, and may be set at any point before **plot**. For example, there are the x and y axis labels (**xlabel**, **ylabel**) and the plot title (**title**). The size of the plot defaults to about 3 across by 3 high, with extreme values slightly larger than those found in the input series. These things can be overridden using **xlimit** and **ylimit**. When all the data are strictly positive, logarithmic scales can be set with **logxy**.

Additional graphs may be created by reading in new data and invoking **plot** again as often as necessary. To terminate *plotxy* type **stop**. The program halts and the plot-file can be displayed or printed.

COMMAND CATALOG

The commands are given their full English names here (though only four characters are needed). 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 brackets; alternates are indicated by slashes. 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 line beginning with one or more spaces 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 [e f]

With 4 numbers: 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. With 6 numbers: uses new(x) = a∗x + b∗y + c, new(y) = d∗x + e∗y + f. This is an affine transformation.
Blank command field: cancels current transformation

background color [note]

Paint the rectangular region framed by the current axes uniformly with the specified color; see command **color** for how to specify the color, and **palette** on how to customize it. The background is opaque, so that it will cover previous plots if the current graph lies on top of earlier material; this makes the color white useful. Setting color to – (minus) returns to the transparent background.

When *note* is included the background color is applied instead to a rectangular region around each of the notes. Often the color is white (or that of the graph background) and then the text will appear to be posted on top of the graphical material and grid lines. This option does not work for notes in an external file.
Blank command field: retain previous setting

cancel [n]

Removes the last n data series read into memory. If no series have been read in, do nothing. If n is greater than the number of series currently present, all the series are removed without complaint.
Blank command field: n = 1

character h [angle]

Change the height of the lettering in titles, labels, notes and axis numbering to h inches. The new value applies to the subsequent 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. The optional parameter angle specifies the angle (counterclockwise) 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 = 0.08, angle = 0

clear [notes] [text] [axes] [read] [data]

Plotxy carries forward almost all attributes from a previous graph to the next, but it is often useful to keep some but not others. With *notes* this command erases the notes, or with *text* all labels, titles and notes. The *axes* option returns the axes to linear in x and y, places them at the bottom and left with default lengths and undefined ranges. The option *read* resets the read attributes to **mode** 2, straight-lines interpolation, no affine transformation, and input from the terminal (**file** *). *Data* deletes all the data series. All of these actions can be achieved with other commands (see **cancel**, **notes**, **xlim**, etc), but **clear** offers a concise alternative for a common requirement.
Blank command field: clear everything

color color
color [pale/light/dark] color
color n

This command sets a new color from the list: black, red, blue, green, brown, orange, yellow, purple (aka magenta), gray, cyan, white. White is useful for drawing on top of dark curves. The shade can be modified with the second form; light colors are best with the **fill** or **background** commands. Pale colors are even lighter. See **palette** for a way to customize the colors; then the new colors are associated with numbers, n. The color assigned to a data series is the one in force at the time of the associated **read** command; similarly with axis labels, notes, etc. The axes and frame are drawn in the color specified at the time **plot** is called.
Default black

dash [s1 s2 [s3 s4]]

The next data series to be read will be drawn as a dashed line, with visible segments s1 inches long and missing segments s2 inches long. If s3 and s4 are included the pattern now repeats with four elements, s1 and s3 representing the lengths of the visible segments, and s2 and s4 describing the gaps. Sensible choices are made if the user enters only one or three values rather than an even number. **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 = 0.03, s2 = 0.05

feet h
When error bars are drawn they are provided with feet, which are short, perpendicular segments terminating the bar. By default their size is that of the current letter height (see **char**). This command supersedes that default with the height h inches, probably most usefully with h=0.
Default: h=letter height
Blank command field: return to default setting

file filename

Defines the file name of the diskfile from which data are to be read; or the symbol * which implies read from the terminal. The name must consist of 64 or fewer

characters. After this command the next **read** statement will begin at the beginning of the file. See NOTE 11 for the special file O which contains points on a unit circle.
Default: ∗
Blank command field: Rewind current file

fill

Fills a closed polygon based on the next input series with the current color. A closed curve is created by joining the first and last points of the series with a straight line. **Fill**, unlike most commands, does not remain in force, but applies only to the next **read**. Solid symbols are invoked with the **symbol** command. If parts of the curve go outside the plot window, the results are unpredictable. Also a polygon with a dashed boundary will not be filled. *Plotxy* plots the data series in the order they are input; this allows the user to cover up parts of the graph with filled regions. Notes are plotted last on top of any filled areas. See also **background**.

format format specifier

An obsolete command from the days when Fortran was king: defines a Fortran format for reading the next data series. The format specifier may be

(1) a normal Fortran format specifier enclosed in parentheses, for example: (2g12.4)

(2) the single character ∗ meaning 'format free' reading

(3) the single character *b* meaning a Fortran binary read.

In each case the data are read with a single Fortran read statement of the appropriate type. In type (1) 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 in mode (1). If the numbers can be unambiguously read by a person there is no need to invoke this command.
Default: ∗
Blank command field: ∗

frame +box −box grid −grid none
frame −xnum −ynum −xaxis −yaxis
frame top/bottom left/right
frame +xeven +yeven

If +*box* is present, two more sides are added to the axes to complete a rectangular box around the plot; −*box* removes the box. *grid* adds a light grid corresponding to the axis tick marks as well as adding a box; *grid-dash* draws dashed grid lines rather than solid ones; −*box* cancels the grid and the surrounding box and returns to the normal situation with two orthogonal axes. −grid removes the grid but not the box. The option *none* strips the graph of axes, the box and the grid.

The second form of the command allows you to delete the numerical annotation from either axis: with −*xnum* the numbers are removed from the x-axis; with −*ynum* from the y-axis. The tick marks remain; they may be removed by setting the character height to zero immediately before plotting. Numbers can be restored with +*xnum* and +*ynum*. Similarly a complete axis may be removed with −*xaxis* or −*yaxis*, and restored later with plus signs. When contradictory options are issued (−*xaxis* and +*xaxis* both present, for example) *plotxy* takes the option omitting the least amount of information.

The third form allows the user to draw an axis on the "wrong" side of the graph: at the right or at the top; the labels move with the axes. Obviously *bottom* and *left* restore the axes to their conventional locations.

The last form controls the tick mark style: normally numbered ticks are larger than those without numerical labels. The desirability of this arrangement is not always unambiguous. The height of the ticks on an axis can be made even with +xeven for the x axis, +yeven for the y axis. The default is restored with −xeven, −yeven.
Default: −*box*
Blank command field: +*box*

help

Lists the four-letter abbreviation of all the commands. This may remind you of a name you have forgotten. Some systems also allow access to the documentation file.

landscape

Causes the output to be rotated by 90 degrees clockwise, that is, landscape mode. The command must be issued before the first **plot** command in order to be effective. The orientation state cannot be changed back to the conventional (portrait) style unless a new plotfile is created with **output**. Then, if **landscape** is used a second time, before the first plot on the new file, the state is reversed from its current one.

logxy [style]
logxy [n]

Specifies the type of scales for the next plot with one of the following styles: *linlin, loglin, linlog, loglog, linlogy+, loglogy+, equilin* where the first syllable describes the x axis, the second the y axis, so that *loglin* means the x axis is logarithmic and the y axis linear, and so on. The style may also be specified by an integer: *0 linlin, 1 loglin, 2 linlog, 3 loglog*. If nonpositive data values are encountered, *plotxy* switches to linear axes, unless one uses *linlogy+* or *loglogy+*: either of these causes nonnegative y values to be ignored, and the positive values to be plotted on the log y axis.

Somewhat obscurely, style may also be *equilin* (or 4). Then a linear scaling is performed, but instead of scaling x and y separately, the data are scaled equally and sized to insure the data area is contained within the plot window. The original proportions of the data are preserved, circles remain circles, etc. The values of x1, x2, y1, y2, in **xlim** and **ylim** must be unset (or zero).
Default: *linlin*
Blank command field: *loglog*

mode n [col1 col2 col3 col4]

Defines how the input data are grouped in the next and subsequent **read** commands. The integer n may be 1, −1, 2, 3, −3, 4 or 10, −10, 20, 30, −30: mode 1 implies data are simply consecutive y values with x uniformly increasing, beginning at x = 1 and unit increment (these x values may be modified with **affine**); mode = −1 means that x values are read and that y increases uniformly from 1 with unit increment; mode = 2, the default, 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 (an error bar is plotted between y−z and y+z). mode = −3 means the third member of data group is the uncertainty in the x-value. A symbol may be plotted at the actual value of y itself if **symbol** is set. If s, the symbol height, is set greater than or equal to 10, a symbol is drawn with height z instead of the bar. Also see **feet**. 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.

In modes 10, 20, 30, it is assumed the input file is in the form of a table. For mode 10 the single column, col1, is selected to be plotted against uniform x. With −10 x is plotted against uniform y. Similarly mode 20 allows you to read x from col1 and y from col2 from the table. For example, mode 20 5 2 means column 5 is used for x and

column 2 for y in the x-y series. Modes 30 and −30 are similar, picking the three columns from the table in the obvious way. In these modes any supplied **format** is ignored and replaced by ∗. The columns are separated from each other by spaces (not commas or tabs), and skipped columns may contain non-numerical material.

Asymmetric error bars can be input in modes 30 and −30 by giving four columns, instead of three. Then col1 and col2 contain x, y for the data point as before, col3 now holds the lower end of the error bar, col4 the upper end.

If **fill** is issued before a mode 3 read, and no symbol is indicated, a filled corridor is plotted, and the feet are omitted.
Default: n = 2
Blank field: after n = 10, 20, 30, col1 = 1, col2 = 2, col3 = 3

nocomment

In normal operation command lines can be annotated by comments, indicated by the % sign. Everything after a % on the line is ignored. But users may wish to use the symbol % in labels, notes, file names and so on. Including this command causes the commenting facility to be permanently turned off from that point on, thus restoring the use of % to the user.

note (x y [in] [c] [r]) text
note (p q x y [in] [c] [r] [f]) text
note (+) text
note (v[c] [r]) text
note filename

In the first two options above, *plotxy* reads the characters of text to be plotted on the graph at the coordinates x, y (which do **not** participate in the current **affine** transformation). The text may be up to 80 characters in length. If the optional *in* appears, the coordinates (x, y) 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. Normally, the coordinates x, y refer to the bottom leftmost point of the first character of the text, but if the letter *c* appears, the text is centered above the point (x, y); and if *r* appears, the base of the end of the line is placed there (right justification). Notice 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**. To see how many separate notes are allowed in your version of *plotxy*, enter **status** (200 is common). To delete the notes, enter **note** and a completely blank command field, or use **clear**. Old notes must be cleared for new graphs with fresh set of notes, otherwise notes from the previous plot will be carried forward onto the current one.

If no coordinates are supplied, but a single + appears instead, the text is appended to the material of the previous note, thus overcoming the limitation on character count in a single note.

If a single v appears, place the new material vertically under the previous note, left justified. With vc center the new line under the previous one, and with vr right justify the new text.

Notes may be written on an opaque background covering up other material if desired; see **background**.

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 as specified by r, c, or left justified by default. The text is always horizontal with this option. The letter f following the coordinates causes the arrow head to be filled in. If

no text accompanies a four-coordinate note, the arrow tail goes exactly to the second position.

The following option allows a large number of short notes to be read from a file. When a filename follows the command, coordinates and notes (one set per line) are read from the named diskfile to its end. Up to 600 such notes can normally be read (Use **status** to check this). Each note in the notefile is at most 16 characters long and the whole series comprises text, homogeneous in character size, angle, and color specified at the time when the file is read. Font changes can be induced in the usual way by phrases in the notes themselves. The coordinates are in graph units only and must *not* be enclosed in parentheses. The notes are all left justified. These notes are canceled by **note** with a blank command field just like those specified as commands. If it is desired to begin the note with a space, terminate the coordinates with a comma; spaces after the comma are part of the note.
Blank field: delete all old notes

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. Suppose you make a mistake; if you enter the command **output** without a filename the current plotfile is erased. Subsequent output is sent to the current file which has been re-initialized.
Default: *mypost*
Blank command field: restart plot in current file

palette n h s b

Assign to the integer n to a color with hue h, saturation s, and brightness b. Normally n should lie between 45 and 68. For n in the range 1 to 44, existing colors will be changed. The command may be repeated with different n to load a selection of colors. **Palette** sets the colors for the next graph to be drawn, not the next data series, so that resetting the color for same integer n several times will result in only one color being chosen, the last one set. The three HSB color coordinates (h, s, b) are defined in any book on PostScript: Roughly, h goes continuously around the color circle: from 0 (red), to 0.333 (green) to 0.667 (blue) back to 1 (red again); s sets the amount of white mixed in: it ranges from 0 (all white) to 1 (no white – full saturation); b is brightness: from 0 (black) to 1 (maximum brightness).
Blank command field: do nothing

plot [top/center/bottom] [left]

plot [up/down/right/left/top]

plot [x0 y0 [abs]]

Creates the next complete graph containing all the data series currently in memory, writing the PostScript file describing it to disk. The file is usually named *mypost*; you can set a different name with **output**. 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.

The very first plot of a series can be positioned as shown in the first form of the command: *top* at the top of the page; *center* in the middle; *bottom* at the bottom. The plot will be centered left-right, unless *left* is included, in which case it will be left justified. If the first **plot** command is issued without positioning parameters it will fall at the bottom centered left-right. The paper size is assumed to be 8.5 by 11 inches.

Subsequent graphs can be positioned relative to earlier ones by using the second form of **plot** in the obvious way. The option *top* will bring the current plot to the top

of the page to the right of all earlier plots; *center, bottom* will be ignored after the first graph. If a second or later **plot** command is issued without parameters, the graph will appear below the previous plot if the first plot was at the top of the page, above it otherwise.

If these arrangements are insufficient, you can take complete control of positioning by specifying coordinates x0, y0 (inches) of the plot origin, which for these purposes is the place where the annotated axes cross, not the point (0, 0). The coordinates refer to the bottom left corner of the paper with *abs*. Omitting *abs* implies relative positioning: that is, x0, y0 is the position of the new plot *relative to the previous origin.* The option *abs* is implicit in the first use of **plot**.

All graphs will be drawn on the same physical page. To get a second page you must make another plot file with **output**.
Blank command field: *bottom*

read [n]

Performs the reading of the file according to the specifications in force at this point; see **mode**. Each **read** instruction is performed with a single Fortran READ statement with an implied DO. This means in **mode** 2 consecutive points are allowed to run across a line as x1 y1 x2 y2 x3 y3 ... ; and similarly in modes 1 and 3. 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 (that is, the next line in ASCII files); if the eof was reached, another read on this file will begin at the beginning. Up to 500 separate data series may be present at any one time. Note that n must be explicit if **file** = ∗
Blank command field: read to end of file

save

Normally when new points are read after a **plot** command, the data values for the previous graph are erased. To prevent this, the command **save** must be entered before the next **read** statement; then the old and the new data are plotted together on the new graph. If no new data are to be read in, there is no need to use **save** since the earlier data are retained for plotting in this case.

smooth [natural/akima/off]

Decides whether continuous curves of y against x are interpolated with straight lines, **smooth** *off*, or natural cubic splines, *natural*, or Akima splines with *akima*. Akima splines are piecewise cubic curves with less overshoot than natural splines. When splines are used, the series is taken to be a single-valued function of x and the actual x values are re-ordered to be increasing by *plotxy* if necessary. Also see NOTES 2. This command is an input attribute, applying to subsequent **read** commands, not to the whole plot. **Smooth** 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 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 *plotxy* find its own limits. The command field may be abbreviated to the first two letters.
Default: *off*
Blank command field: *natural*

skip [n]

Skips the next n records in the current data file. With normal ASCII files this means skipping n lines. The command examines the current **format** to determine whether the current file is binary or ASCII.
Blank command field: n = 1

status

Lists a synopsis of the current data series (their lengths, extreme values and other attributes), the plot and reading parameters, and the memory available for further data series. Also lists some program array limitations of the current version of *plotxy*.

stop

Closes the output file and brings *plotxy* to an orderly halt. This must always be the last command of any run from a terminal, otherwise part of your plot may be lost. **Stop** may be omitted if *plotxy* is running from a script, since the end-of-file tells the program input is finished.

symbol n [h]
symbol [filled/solid] name [h]

Specifies that the next input series be a set of discrete points designated by symbols of height h, rather than a line. In the first form the kind of symbol is defined by the integer n.

0 square	8 upward arrow	17 circle
1 triangle	9 hourglass	22 dash
2 octagon	10 campstool	23 pentagram
3 diamond	11 hexagon	24 pentangle
4 plus	12 Y	
5 asterisk	13 vertical bar	
6 cross	14 star of David	
7 barred square	15 dot	

Missing integers in the list refer to obsolete entries. The more common symbols can be referenced by name instead of the integer code: *asterisk, circle, cross, diamond, dot, hexagon, octagon, plus, square, star, triangle, and pentagram*. The objects drawn are outlines; solid symbols are obtained from this list by prepending *filled* or *solid*, as in *filled circle* or *solid hexagon*.

The approximate height each of symbol is h inches. Negative h causes the symbol to be drawn upside down, useful for triangle and arrow. If h is omitted, the value from the previous **symbol** command is inherited. The size can be variable, set through the third column in **mode** input; see below.

To revert to previous line style, use **symbol off**.

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, \2017\ for a circle. Add 2050 for a filled symbol.

To plot a series of a single kind of symbol which varies in size, you must combine **symbol** with **mode** 3 or 30; as the mode number suggests, three values are then read for each input point. Then, provided h in the **symbol** command is set greater than or equal to 10, the three numbers on each line of the file are read as x, y, z, where z is the symbol height in inches.
Default: n = −1, h = 0.15

title text

Provides 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 comprises a font-setting phrase, that sets the font for the rest of the graph and the graph is untitled.
Default: text = blanks

xlabel text

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

xlimit xlength [x1 x2]
xlimit xlength [x1 x2 dx]
xlimit

Defines the length of the x axis, xlength, in inches and the lower and upper limits of x: x1, x2. Only points lying inside (x1, x2) are plotted; those outside are omitted from the plot. If x1 = x2 = 0, or if these values are omitted in the command, the x extremes are chosen to encompass all the values in the data series (autoscaling). 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.

When the fourth argument, dx, is supplied, tick marks and numbers are written at integer multiples of dx, provided this results in a reasonable quantity of them; otherwise, the defaults are invoked. With log axes, dx has no effect except when it is set negative, and then tick marks between powers of ten are suppressed.

If the command is given with a blank field, the current plot retains the same x axis attributes and scaling as those in the previous graph. Simply omitting the command has the same effect, except when the prior plot was auto-scaled; now that scaling continues to apply, rather than a new one based on the current data.

The default behavior for plot size is somewhat complex: if the x and y variables cover similar intervals (within a factor of two), the default height and width are arranged to be in the proportions of the data, thus giving the same scales for the x and y variables. Otherwise the height and width are both 3 inches.
Default: xlength = 3, x1 = x2 = 0
Blank command field: retain attributes of the previous plot

ylabel text

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

ylimit [ylength [y1 y2 [dy]]]

Same as **xlimit** but for the y axis.
Default: ylength = 3, y1 = y2 = 0
Blank command field: retain attributes of the previous plot

weight w [wlines]

This command controls the weight of plotted lines by giving the integer w, the line thickness in one-thousandths of an inch. Optionally, lines and points on the graph can be given a different (usually heavier) weight from text and other material; this is set in wlines. Invoking **weight** with only one parameter implicitly sets wlines=w for subsequent input data.

Default: w = 6, wlines = 12
Blank command field: restore default

DEFAULT VALUES

See NOTES 10 for the way to change these.

affine 1, 0, 1, 0
character 0.08, 0
dash 0, 0
file *
format *
frame *off*
logxy 0
mode 2
weight 6, 12
output *mypost*
smooth *off*
symbol *off* 0
title blanks
xlabel blanks
xlimit 3, 0, 0
ylabel blanks
ylimit 3, 0, 0

LETTERING

Plotxy provides four fonts in which the title, 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 *complex*. 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 **title**. If you want to vary the fonts within one plot you can specify the desired font changes in the text; font changes may appear at any point in a piece of text. There is also a math font obtained by: \$\. In it letters are set in italics, while all other characters retain the previous font setting. The math state toggles to and fro, so that \ $\x=y\$ \ will be set in italics in a line set in some other font. Math mode ends with the end of the text line (unlike other font changes). Note: *plotxy* draws its own letters as a set of lines, and does not use PostScript fonts.

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. The default theta and phi characters are the flowery script versions: to get plain ones type: \rgtheta\ and \rgphi\, for regular theta and phi. 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. Characters may be embellished with a hat \^\, or a tilde \~\, or a an overbar \–\. The code follows the decorated letter: x\–\ for example.

There are some characters with no keyboard equivalent. Some of them can be specified by their names enclosed in backslashes; these may always be shortened to the first 4 letters. Here are the special characters for which this is permitted:
gradient, integral, infinity, times, partial derivative, degree. For square root, use \sqrt\.

There are other, perhaps more obscure, symbols that you can access only with numerical codes: a 4-digit key number enclosed in backslashes. Here is a table of the codes for these special symbols; curly braces can be accessed only through their 4-digit

codes, not the characters on the keyboard. Astronomical symbols generously provided by Duncan Agnew.

1387 partial d	1403 summation	1439 Venus
1388 grad	1404 regular theta	1440 Earth
1389 member of	1405 {	1441 Mars
1390 less or equal	1406 }	1442 Jupiter
1391 greater or equal	1412 paragraph	1443 Saturn
1392 proportional	1413 dagger	1444 Uranus
1393 integral	1425 tall <	1445 Nepture
1394 circuit int	1426 tall >	1446 Pluto
1395 infinity	1429 degree	1447 crescent moon
1396 + or –	1430 tends to	1448 comet
1397 – or +	1431 regular phi	1450 Aries
1398 times	1434 not equal	
1399 division	1435 approximates	
1400 product	1436 shah	
1401 times dot	1437 Sun	
1402 radical	1438 Mercury	

Finally, another use for the numerical code is to specify text size: the height of the text following the phrase \0.25\ is changed from its current value to 0.25 inches; any number less than 5 is interpreted as the letter height in inches. The values are rounded to hundredths of an inch. Letter heights defined in this way hold only for the note or other text in which they appear, subsequent material reverts to the size set by default or **character**.

NOTES

1. *Plotxy* is reasonably graceful with error conditions: explanatory messages are issued in most circumstances. The messages are usually in two parts: what went wrong and what *plotxy* has done about it. For example, if unintelligible data are encountered, *plotxy* will reject the whole series and then issue a warning. Attempting to plot negative data on a log scale generates an error message and the offending scale is made linear instead of logarithmic, unless the *logy+* option has been selected in **logxy**.

2. When **smooth** is used, the data series is re-ordered if necessary to make x increase; repeated x-values will turn off smoothing, and the re-ordered series are drawn joined by straight-line segments.

3. If plot extremes have been set with **xlimit** or **ylimit** and a data value falls outside the window, the result depends on whether smoothing has been set. Without smoothing, the virtual "pen" is simply lifted until onscale data are encountered again. This allows you to insert breaks in your data records by inserting large values; but remember to set limits explicitly before **plot**. When **smooth** is on, *plotxy* draws a piece of curve between the last captured point(s) and the edge of the graph in the direction of the exterior point.

4. To plot a line with symbols at the coordinates, read the data file twice, once with an interpolating line and then with **symbol**. Lines with both long and short dashes can be drawn by entering four values in the **dash** command.

5. The sizes of the arrow heads can be controlled: they are proportional to the current symbol height set with **character**. If you would like a larger arrow head with regular size text in a note, you can regulate the text height with a size phrase, like \0.10\.

6. Any piece of text may be preceded by a font phrase; its height may be specified either by the **character** command, or as described in the previous section. To arrange a special font for the axes numerals put the font phrase at the end of the title.

7. Reversing x and y axes (as when coordinates are specified as latitude and longitude) is easily done with **mode** 20:

```
mode 20 2 1
read
```

Modes 10, 20, 30 can read from tables in which the columns contain characters besides numbers. However, the nonnumerical columns must not contain spaces, since that how the columns are separated.

8. *Plotxy* is written almost entirely with single-precision (real*4) variables. If data are to be plotted in which the only variation occur in the seventh (or later) significant figure, there will be inaccuracies. One solution (other than recompilation) is to subtract a constant value from the data values before plotting to map them into a more manageable interval.

9. The preset defaults may not be to the user's taste. Finding and replacing them in the code would be tedious. Instead, the last subroutine, 'dfault', executes commands internally before *plotxy* reads from the command line. Preferred defaults, such as font, letter height, background color, etc can be set there, and will override the defaults given in the documentation.

10. Sometimes data are best displayed as symbols on top of a vertical stalk that extends to zero. Obviously a special data file could be prepared to do this, but here is simple alternative: first, create a data file with x,y,0, one triplet per line. Then this plot script will display the series with a filled square atop a stalk:

```
feet 0                          % Erase feet from error bars
symbol filled square 0.06       % Plot a square at (x,y)
file data
mode 30 1 2 2 3                 % Treat data as asymmetric error bar
read                            % between 0 and y
```

11. There is one special filename for **read**, the letter O. This creates an internal 'file' usually with 361 lines comprising x-y pairs, cos(n-1)a, sin(n-1)a, where a is one degree, defining an approximate unit circle centered on the origin. Read to the end, O will produce a unit circle, which can be displaced and scaled or stretched with **affine** if desired. Arcs of a circle can be made by skipping and reading the required portion of the whole. Thus

```
file O
skip 90
read 91
```

gives a 90 degree arc in the second quadrant. The 'file' O is normally read with **mode** 2, but 10 and -10 are also permitted. In these modes, the first column of cosines is treated as input and a uniformly spaced x or y series created as appropriate (see **mode** 10), thus creating a cosine series. For this purpose the number of points specified in the accompanying **read** command can be greater than 361 to obtain a series longer than one period.

12. **Mode 3** reads with 4 columns, used to specify asymmetric error bars, can be adapted to several other uses. For example, a filled region can be built by giving in tabular form x and its upper and lower boundaries at that x. Then

```
fill
mode 3 1 2 3 3
read
```

draws the region. Symbol must be off. Notice how there are only 3 columns in the table, but the mode command field has 5 entries, repeating the last column. Also see NOTE 10.

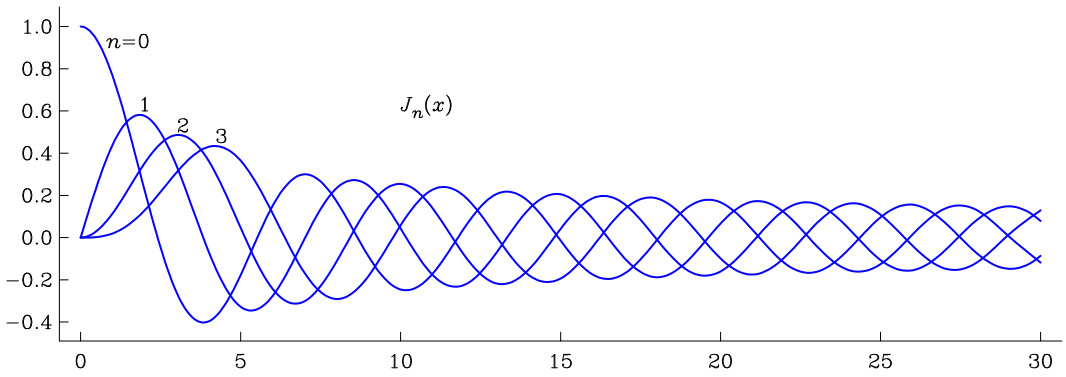
EXAMPLES

Here are two examples of quite presentable plots made with relatively little effort. The first is an illustration of Bessel functions of low order. A rather sparse table comprising 5 columns is held in the file *ex1.dat*, with x in the first column and the functions in those that follow. These are picked off one at a time with **mode** 20.

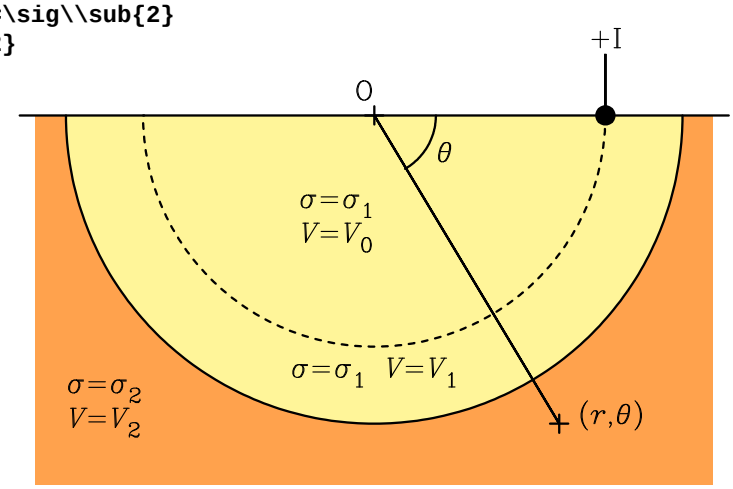
```
 Bessel functions Jn (x) for n=0,1,2,3

xlim 6 0 0 5
ylim 2

smooth
color blue
file ex1.dat

mode 20 1 2
read
mode 20 1 3
read
mode 20 1 4
read
mode 20 1 5
read

color black
note (10 0.6)\$\J\sub{n}(x)
note (0 0.9)  \$\n=0
note (1.8 0.6)1
note (3.0 0.5)2
note (4.2 0.45)3

plot top
```

In the script above the picture has been placed at the top of the page but here I have put it at the bottom. You should be able to reproduce it from the example files on the website.

On the next page we show a diagram illustrating the layout of a boundary value problem for the potential in a DC electrical conductivity problem. Current is injected at the electrode, shown by the black dot. The internal file O contains points on a circle of radius one, and this is read four times, at different scales using **affine**. Very short series are entered in the script itself with **file** *. Of course, to help with positioning the notes during development, the graph was drawn with **frame** *grid*, but the grid was removed at the end.



```
char 0.1
logxy equilin
frame none

color light orange
fill    % Orange background
read 4
-1.1 0 1.1 0 1.1 -1.2 -1.1 -1.2

affine 1 0 -1 0
file O
color light yellow
fill
read 181
color black
read 181           % Outline yellow region
dash 0.03 0.04
affine 0.75 0 -0.75 0
read 181         % Divides 2 types of solution
dash 0 0
affine 0.2 0 -0.2 0
read 61       % What does this do?

file *
affine
symbol filled circle 0.10
read 1
0.75 0
symbol plus
read 2
0 0 0.6 -1
symbol off
read 8
 -1.15 0 0 0 0.6 -1 0 0
 0.75 0 0.75 0.2 0.75 0 1.15 0

note (0 -0.85 c)\$\\sig\=\sig\\sub{1} V=V\sub{1}
note (0.6 -1) \$\(r,\rgtheta\)
note (0.20 -0.15)\$\\rgtheta\
note (0 -0.3 r)\sig\=\sig\\sub{1}
note (vc)\$\V=V\sub{0}
note (-1 -0.9 )\sig\=\sig\\sub{2}
note (vc)\$\V=V\sub{2}
note (0.75 0.22 c)+I
note (0 0.05r)O

xlim 4
ylim 3

plot
```



LOCAL IMPLEMENTATION

On the various Unix networks at IGPP, UCSD, *plotxy* is an executable command made available through one of the standard paths like /usr/local/bin.

POSTSCRIPT ANNEX

All installations of *plotxy* produce PostScript output, although the program was originally designed to be independent of any particular graphics implementation. *Plotxy* running in PostScript mode prints a message informing you of the PostScript output; then the default output file is named *mypost*. Because PostScript files are ASCII, you may edit them to produce special effects. To help you, *plotxy* inserts comment lines to indicate the section being plotted.

PostScript has only limited capacity to draw filled regions (only 1000 points). You may be able to get round this by using the device described in NOTE 12.

Plotxy gathers information to enable you to create an encapsulated PostScript file. At the end of the plotfile two lines are added that contain information about the overall size of the picture. Move these two lines to the top of the plotfile, and it is encapsulated.

NEW STUFF

feet new command to set error bar foot size. (8/12)

dash takes up to four numbers, allowing dashed lines with a repeating pattern of four elements rather than just two. (7/12)

xlim, ylim allowed without argument: retain earlier attributes. (2/12)

file O: special internal file containing unit circle; see NOTES 11. (4/10)

background color note, added (1/10)

clear command added, (11/09)

Filled version available for all named symbols in all functions. (9/09)

pale colors added. (8/09)

15 new characters added, including astronomical symbols; **symbol**: 3 new centered symbols; **color**: new color cyan; **logxy**: linlogy+, loglogy+ ignores nonpositive y values on log axes; **affine**: Accepts 6 parameters, permitting rotations as well as stretching. (11/08)

mode: 10,20,30 now skip columns containing non-numerical material.

plot: options left, right, up, down. (12/07)

Asymmetric error bars in **mode** 30. (11/05)

Centered or right justified text blocks in **note** (vc) (vr). (9/05)

Dark and light colors added in **color**. (3/05)

Alternative default parameters can be conveniently set at compile time: see NOTES 10. (8/04)

INDEX