Using QPlot as a stand-along program

Daniel Wagenaar, 2013

For most users, it will be much more convenient to use QPlot from within Octave or Matlab. However, in certain situations, such as when producing a large number of graphs in an automated fashion, or when implementing a shell around QPlot for another computation environment or programming language, it may be advantageous to use QPlot directly. This document briefly describes how to do that.

1 User interface

QPlot can be run on the command line, like this:

qplot *source output*

where *source* is a file with commands and *output* can specify either pdf, svg, png, or tiff output. Output can also be a postscript (.ps) file, in which case a single full page is produced with the graph in the middle and crop marks around it. Output to eps is not directly supported, but “pdftoeps -eps -level3” can be used as a postprocessor.

QPlot can also be run interactively, like this:

qplot *source*

In that case, graphics are rendered in a window. Keys “+” and “-” zoom in and out, “0” scales to fit, “1” scales to 100%. “E” resizes the window to fit the whole scene. “G” toggles between white and gray borders. “C” enables or disables reporting coordinates below the mouse pointer. “Ctrl-Q” quits. The graphics are automatically rerendered if the *source* file changes on disk.

2 Commands

Optional arguments are given in parentheses. Vertical bars indicate alternatives.

align left|right|center|top|bottom|middle|base

Sets horizontal and/or vertical alignment for subsequent text.

area [dx_1 dx_2 …] [dy_1 dy_2 …]

Draws a polygon, filled using the current brush. dx_i and dy_i are specified in points and are relative to the position set by **at**. Note that the “[” and “]” are literal brackets that separate the x- and y-coordinates. See also **patch**.

at x y
at x y ξ η
at x y ϕ
at –
at x y *ID*
at *ID*

Places subsequent text and lines at graph position (x, y). If (ξ, η) are given, this specifies a rotation such that the baseline of the text is in the direction of the vector (ξ, η). This vector is specified in data coordinates. Alternatively, a rotation may be specified as a direct (clockwise) angle ϕ. Besides a numeric value, x may be one of "left", "right", or "center" to place relative to the bounding box of the last drawn object (or group, see **group**), "abs" or "absolute" to revert to absolute placement in the horizontal direction, or a dash ("–") to retain the previous anchor for the horizontal direction. Likewise, y may be one of "top", "bottom", "middle", "abs", "absolute", or a dash. "at –" reverts to absolute placement (i.e., relative to the top left of the current panel) in both horizontal and vertical directions. **at** may also be used to mark a location for later reference, as per the final two syntax forms.

brush (*ID*) *color*|none|*opacity* ...

Selects a brush by *ID*, defines its color (or sets it to "none"), and/or its *opacity* (as a number between 0 and 1).

caligraph [x_1 x_2 ...] [y_1 y_2 ...] [w_1 w_2 ...]

Draws a polyline of variable width. x_i and y_i are specified in data coordinates. w_i specify the line width (in points) at each vertex. (Thus, the length of the vector w must match the length of x and y.) The line is rendered using the color of the current pen; dash patterns and join and cap styles are not respected.

figsize w h

Sets the size of the figure to (w x h) points. This should appear before any other commands.

font *family* (bold) (italic) *size*

Selects a new font with a given family, point size, weight and/or slant.

garea (*ptspec*) ...
gline (*ptspec*) ...

Ultraflexible polygon and line series drawing. Each vertex is specified by a *pt-spec*, i.e., a sequence of one or more subcommands:

absdata *x y*	Absolute data coordinates
reldata *dx dy*	Relative data coordinates
abspaper *x y*	Absolute paper coordinates (in pt)
relpaper *dx dy*	Relative data coordinates (in pt)
rotdata ξ η	Rotate by atan2(η, ξ) in data space (this affects subsequent relative positioning)
rotpaper ϕ	Rotate by ϕ radians (this affects subsequent relative positioning)
retract L	Retract preceding and following segments by L pt
retract L_1 L_2	Retract preceding and following segments by L_1 and L_2 pt respectively
at *ID*	Absolute paper coordinates of location set by **at**
atx *ID*	Absolute paper x-coordinate of location set by **at**
aty *ID*	Absolute paper y-coordinate of location set by **at**

For **absdata** or **abspaper**, either *dx* or *dy* may be given as a dash ("-"), in which case the corresponding coordinate is not affected. (To achieve the same for **reldata** or **relpaper**, just use zero.) Note that the parentheses are literal, unlike in the rest of this manual, where they designate optional parameters. For instance:

```
gline ( absdata 0 1 relpaper 5 0 )   ( absdata 0 1 relpaper 0 5 )
```

draws a line from 5 pt to the right of the point (0,1) in the graph to 5 pt above the point (1,0) on the graph.
(Note: The rather cumbersome syntax of **gline** makes **line** and **plot** more attractive for general usage. The same applies to **garea** versus **area** and **patch**.)

group
endgroup

Groups statements to accumulate bounding boxes for **at**. **endgroup** also restores pen, brush, alignment, font, and reference text to their states before the corresponding **group**. Note that named pens and brushes changed inside a group are not restored. All groups must be closed before changing panels, else the group stack is cleared automatically and a warning message is issued.

hairline *width*

Specifies a width for lines plotted with zero nominal width, in points. If *width* is zero, hairlines are precisely one pixel wide in the output. This is very useful for raster output but not recommended for pdf or svg output, since the resulting file would become device dependent. The default is 0 for raster output (including interactive output), and 0.25 pt for svg/pdf/postscript.

image *x y w h K* [*cdata*]

Renders an RGB image at given data location. *cdata* is stored as (R,G,B) pixels in row order (unlike matlab's convention); K specifies the number of pixels per row. The length of *cdata* must be an even multiple of $3K$. Values must be between 0 and 1.

line [dx_1 dx_2 ...] [dy_1 dy_2 ...]

Draws a polyline. dx_i and dy_i are specified in points and are relative to the position set by **at**. See also **plot**.

mark [x_1 x_2 ...] [y_1 y_2 ...]

Renders markers set by **marker** at the given data coordinates.

marker *size*|*fill*|*shape* ...) (open|solid|brush) () ...

Define a marker of the given *size* in points and shape (one of "circle", "square", "diamond", "left", "right", "up", "down", "penta", "hexa", "hbar", "vbar", "plus", or "cross") for later use by **mark** and **pmark**. The *fill* style specifies how the marks are rendered: An "open" mark is is outlined with the current pen and filled with white, a "solid" is outlined with the current pen and filled with the pen color, and a "brush" mark is outlined with the current pen and filled with the current brush (which may be "none"). The fill style has no effect on "hbar", "vbar", "plus", or "cross" marks.

panel *ID*|–
panel *ID* x_0 y_0 *w h*

Defines a new panel with given ID to have its top left corner on paper position (x_0, y_0), in points, and size (w x h), in points. Or, reenters a previously defined panel. Or drops out to the top level. While drawing inside a panel, **figsize** changes the size of the panel, and **shrink** affects the panel rather than the figure as a whole. Also **xaxis** and **yaxis** affect the axes in the panel. Choices of pen, brush, font, etc., are not local to panels.

patch [x_1 x_2 ...] [y_1 y_2 ...]

Draws a polygon, filled using the current brush. x_i and y_i are specified in data coordinates. See also **area**.

pen (*ID*) *color*|*width*|*capstyle*|*joinstyle*|*linestyle* ...

Selects a pen by *ID*, defines its color, its width (in points), its join style (one of “miterjoin”, “beveljoin”, or “roundjoin”), its capstyle (one of “flatcap”, “squarecap”, or “roundcap”), and/or its line style (one of “solid”, “dash”, “dot”, or “none”). The word “dash” may optionally be followed by a single number or a vector of numbers (in brackets) that defines the lengths of marks and spaces (in points); the word “dot” may be followed by a single number or a vector of numbers (in brackets) that defines the lengths of the spaces between dots (in points). The default is 3 pts. Setting the color or width while the dash pattern is “none” automatically switches to “solid.” Any number of subcommands may be given on one line in any order. A **pen** command without an *ID* makes changes to the current pen but doesn’t store the result as a named pen.

plot [x_1 x_2 . . .] [y_1 y_2 . . .]

Draws a polyline. x_i and y_i are specified in data coordinates. See also **line**.

pmark [dx_1 dx_2 . . .] [dy_1 dy_2 . . .]

Renders markers set by **marker** .

dx_i and dy_i are specified in points and are relative to the position set by **at**. See also **mark** and **marker**.

reftext *string*|–

Sets or unsets a fixed text that will be used to calculate the ascent and descent of text for the “bottom” and “top” alignment modes.

sharelim (x—y) *ID* . . .

Shrinks the x-axis and/or y-axis of the current panel and the named panel(s) so that they have a common scale. If the panels overlap horizontally (vertically) on the page, the x-axis (y-axis) are additionally aligned.

shrink (*margin*)
shrink *margin ratio*
shrink – *ratio*

Shrinks the axes as necessary so that all graphics and text fits within the bounding box of the figure as defined by **figsize**. Margin is specified in points. Optional *ratio* specifies the desired aspect ratio of y:x data units.

text *dx dy string*

Places the given text *string* at the position (*dx*, *dy*), specified in points relative to the anchor set by **at**.
Underscores and hats make subscripts and superscripts (up to the next space). Slashes and asterisks make enclosed words appear in /italics/ and *bold*. Unicode is supported. The string must be enclosed in single or double quotes.

xlim x_0 x_1

Fixes the limits of the x-axis. If no xlim is given, tight automatic axis limits apply.

ylim y_0 y_1

Fixes the limits of the y-axis.

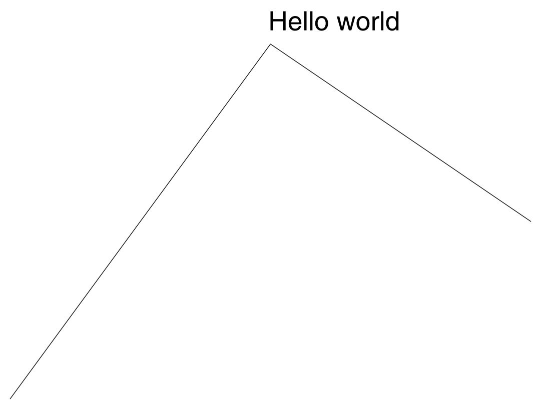
3 Specifying data

For specifying very long vectors or image data, the text-based "[a b c ...]" syntax may be cumbersome. Instead, you can write "*n" and place n binary doubles directly after the command. Or you can write "*uc$n$" and place n binary unsigned 8-bit integers directly after the command. The matlab functions **qplot.m** and **qimage.m** give examples.

4 Example

Here is a very basic example of a QPlot script with its result:

```
figsize 200 150
plot [1 2 3] [1 3 2]
at 2 3
align left bottom
text 0 -5 ”Hello world”
shrink 1
```


Hello world