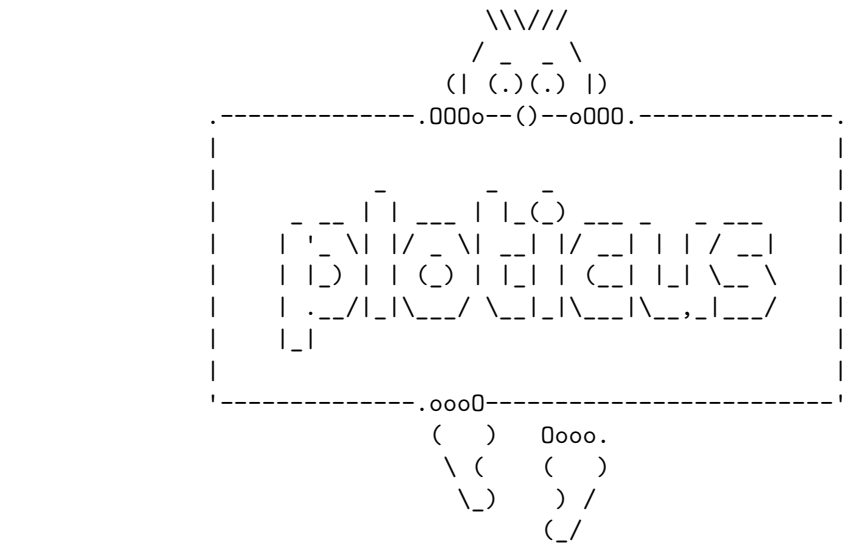



```
Codeblock class: ploticus

    sudo apt-get install ploticus
    http://ploticus.sourceforge.net/doc/welcome.html

    runs:
    > ploticus -{im_fmt} -o <fname>.{im_fmt} {im_opt} <fname>.ploticus

    class->cmd
      ploticus -> ploticus

Metadata options
    imagine.im_out: img,fcb
    imagine.im_log: 4
    imagine.ploticus.im_fmt: svg
```

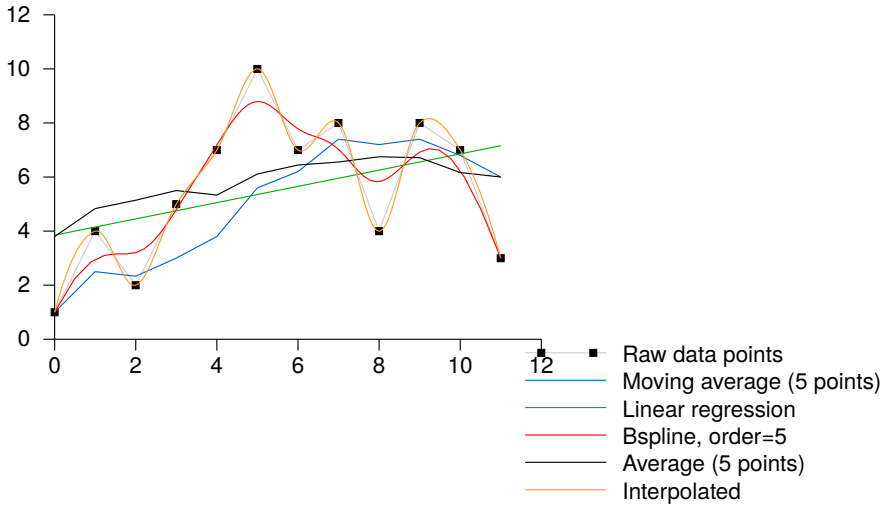
Notes:

- metadata sets ploticus' default output format to `svg`

Ploticus

prefab

Curves script

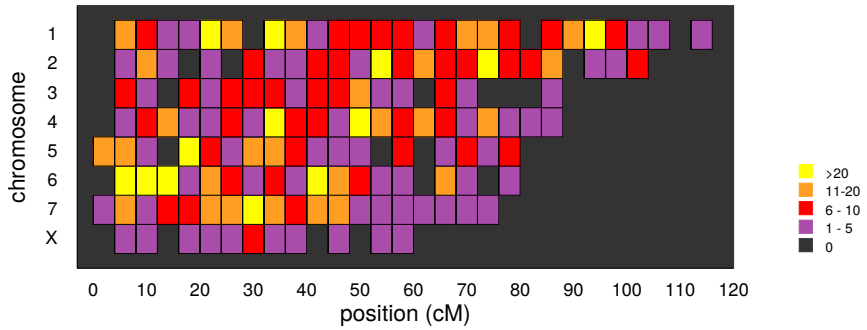



```ploticus
#proc getdata
  data:
  0 1
  1 4
  2 2
  3 5
  4 7
  5 10
  6 7
  7 8
  8 4
  9 8
  10 7
  11 3

#proc areadef
  rectangle: 1 1 4 3
  xrange: 0 12
  yrange: 0 12
```

```
  xaxis.stubs: inc
  yaxis.stubs: inc

#proc lineplot
  xfield: 1
  yfield: 2
  pointsymbol: radius=0.03 shape=square style=filled
  linedetails: color=gray(0.8) width=0.5
  legendlabel: Raw data points
  legendsampletype: line+symbol

#proc curvefit
  xfield: 1
  yfield: 2
  curvetype: movingavg
  order: 5
  linedetails: color=blue width=0.5
  legendlabel: Moving average (5 points)

#proc curvefit
  xfield: 1
  yfield: 2
  curvetype: regression
  linedetails: color=green width=0.5
  legendlabel: Linear regression

#proc curvefit
  xfield: 1
  yfield: 2
  curvetype: bspline
  order: 5
  linedetails: color=red width=0.5
  legendlabel: Bspline, order=5

#proc curvefit
  xfield: 1
  yfield: 2
  curvetype: average
  order: 5
  linedetails: color=black width=0.5
  legendlabel: Average (5 points)

#proc curvefit
  xfield: 1
  yfield: 2
  curvetype: interpolated
```

```
  linedetails: color=orange width=0.5
  legendlabel: Interpolated

#proc legend
  location: max+0.5 max
```

Heatmap (script)




```ploticus
#set SYM = "radius=0.08 shape=square style=filled"
#setifnotgiven CGI = "http://ploticus.sourceforge.net/cgi-bin/showcgiargs"

// read in the SNP map data file..
#proc getdata
file: dta/snpmap.dat
fieldnameheader: yes

// group into bins 4 cM wide..
filter:
   ##set A = $numgroup( @@2, 4, mid )
   @@1 @@A

// set up the plotting area
#proc areadef
rectangle: 1 1 6 3
areacolor: gray(0.2)
yscaletype: categories
clickmapurl: @CGI?chrom=@@YVAL&cM=@@XVAL
ycategories:
    1
    2
    3
    4
    5
    6
    7
    X

yaxis.stubs: usecategories
```

```
// yaxis.stubdetails: adjust=0.2,0
//yaxis.stubslide: 0.08
yaxis.label: chromosome
yaxis.axisline: no
yaxis.tics: no
yaxis.clickmap: xygrid

xrange: -3 120
xaxis.label: position (cM)
xaxis.axisline: no
xaxis.tics: no
xaxis.clickmap: xygrid
xaxis.stubs: inc 10
xaxis.stubrange: 0
// xaxis.stubdetails: adjust=0,0.15

// set up legend for color gradients..
#proc legendentry
sampletype: color
details: yellow
label: >20
tag: 21

#proc legendentry
sampletype: color
details: orange
label: 11-20
tag: 11

#proc legendentry
sampletype: color
details: red
label: 6 - 10
tag: 6

#proc legendentry
sampletype: color
details: lightpurple
label: 1 - 5
tag: 1

#proc legendentry
sampletype: color
details: gray(0.2)
label: 0
tag: 0
```

```
// use proc scatterplot to count # of instances and pick appropriate color from legend..
#proc scatterplot
yfield: chr
xfield: cM
cluster: yes
dupsleg: yes
rectangle: 4 1 outline


// display legend..
#proc legend
location: max+0.7 min+0.8
textdetails: size=6

```

Documentation

ploticus usage

```
#!/bin/bash
ploticus
```

man page

```
ploticus(1)              General Commands Manual              ploticus(1)

Name
       ploticus - data display package

Synopsis
       ploticus  -prefab prefabname   parm=value ..  [-options]
        .. OR ..
       ploticus scriptfile  [-options]

Description
       ploticus  is the primary component of the  'ploticus' data display
       package

       ploticus is a program that produces plots and  charts  from  data,
       and  produces  results  that  can  be  viewed on web pages, paper,
       slides, or interactively on the screen.  Standard types  of  plots
       may  be  done  using   prefab plot templates , or a user-developed
        script file may be supplied  for  greater  flexibility  and  cus-
       tomization.  ploticus may be executed from the command line or  as
       a CGI program.

       For  complete  online  docs  and  downloads   see    http://ploti-
       cus.sourceforge.net

Where to find examples
       See  the  various   prefab  examples  .  A large number of  script
       examples are also available.  Some  usage examples are also  shown
       below.

Command line arguments
       Command  line  arguments  may generally be given in any order.  If
       there are arguments that you want to always have  in  effect,  you
       can  invoke  them  from a  config file.  Many settings can also be
```

made dynamically from scripts via proc settings or proc page. Processing occurs in this order: first the config file is read; then command line args are processed (left to right); then proc page and/or proc settings. Later settings override earlier ones.

Basic command line options

-prefab prefabname

Produce a plot using a prefab plot template. prefabname identifies the template, eg. cron or vbars. Necessary parameters are supplied on the command line using the form parm=value.

scriptfile
-f scriptfile

names a script file that will be interpreted to produce results. Alternatively, -stdin may be used to indicate that script will be available on standard input.

variable=value

Declares the named variable and sets it to the given value. This is a convenient way to pass information to prefabs and scripts. Variable names are case-sensitive.
Example: CUTDATE=10-31-98
sets the variable CUTDATE to 10-31-98.

-o outfile | stdout

Specify a filename where the result will be written. No processing is applied to this name.. so the ending should be appropriate for the selected output format, eg. use .png for PNG files. If -o stdout is used, result will be sent to standard output. If -o is not specified, a default output filename will be used.
Example: -o fp001.png

-dir dirname

Set ploticus' working directory to dirname. If used, this argument should be specified leftmost on the command line, since it affects evaluation of other args.

Result format options
(Availability depends on your ploticus configuration/build)

-png PNG image

-gif pseudo-GIF image

-jpeg JPEG image

-svg or -svgz SVG graphic. See also SVG / XML options below.

-swf SWF (flash) result.

-wbmp WBMP image

-eps EPS (encapsulated PostScript)

-ps paginated PostScript to stdout

-x11 display on X11 screen

-drawdump filename produce no visible graphic; save a generic representation of the graphic result to a file. By using -drawdump and -drawdumpa you can easily overlay or combine results from separate ploticus runs. The drawdump file can be rendered later in any desired format, using this command: ploticus -prefab draw dumpfile=filename or by using proc drawcommands. Drawdump capability is available in all builds. (2.30+)

-drawdumpa filename same as -drawdump but result is appended to file.

Clickable image maps and mouseover options

-csmap

produce a client-side clickable imagemap to accompany a png, gif, or jpeg. These can be used for hyperlinks, and also for providing pop-up text labels that appear when the mouse passes over a region. By default, client-side map content is written to stdout.

-csmapdemo

Same as -csmap but all mapped regions are shown outlined in green, and a complete HTML chunk is produced which involves the output image name.

-mapfile filename | stdout | stderr

explicitly name the output file containing the map info. The name may also be set in proc page. If a name is not specified, client-side image map info will be written to stdout; For SVG this parameter is not needed, since image map info is embedded in the SVG file.

-map

produce a server-side clickable imagemap file to accompany a png, gif, jpeg, or SVG. Note: server-side maps are deprecated.

Result sizing options

-scale sx[,sy]

Scale the final result. If one value is given, the result is scaled by this amount in both x and y. If two values are given, scaling in x and scaling in y may be done independently. A scale value of less than 1.0 reduces the size; an scale value of greater than 1.0 enlarges. Scaling is done relative to the origin (0,0) which is at the lower left.
Example: -scale 0.7

-pagesize width,height

Sets the pre-crop size of the result image for GIF/PNG/JPEG, or sets the display window size when drawing to X11. On other output devices this option does nothing. width and height are in absolute units. 0,0 is the lower left corner. If -pagesize is not specified, the default size will be 8" x 8". Size is set before any drawing takes place and is unaffected by the -scale option.

When rendering PNG/GIF/JPEG images, this option determines amount of internal memory allocation for accommodating the image. The result can never be bigger than this size, and any drawing outsize the bounds will not be visible. To create PNG/GIF/JPEG images larger than 8" x 8", this option MUST be specified to set a bigger size. Cropping options (below) can be used along with -pagesize as long as they result in a smaller rectangle than the pagesize; they take effect after all drawing has been completed.

-pagesize has no effect with EPS or paginated PostScript

results (the PostScript BoundingBox will be determined by the extent of the graphic).

Example: -pagesize 7,3

-tightcrop

For image or EPS output, crop the result tightly to the extent of the design. Normally a small margin is allowed on all four sides. This option sometimes crops a bit too tight; if so try -croprel.

-crop x1,y1,x2,y2

Crop image or EPS result to the box specified by x1,y1 and x2,y2, in absolute units.

Note that there may be no spaces in the coordinates specification. Cropping takes place after design is rendered and does not affect coordinate locations.
Example: -crop 1.2,0.8,4.4,5.2

-croprel left,bottom,right,top

Crop image or EPS result tightly to the extent of the design (like -tightcrop), but then adjust the cropping outward or inward on one or more sides. left is the amount to adjust the left side, in absolute units. Similarly for bottom, right, and top. Positive values always adjust outward from center; negative values adjust inward (tighter). There may be no spaces in the left,bottom,right,top specification. Cropping takes place after design is rendered and does not affect coordinate locations.
Example: -croprel 0,-0.1,0,0.1

-pixsize width,height

If specified, result PNG/GIF/JPG image will be created at exactly this width and height in pixels. Does not interact with scaling or cropping... user is responsible for ensuring that content fits appropriately into the specified size. User is also responsible for setting -pagesize appropriately for larger images. New in 2.40

Graphics environment options

-font font

sets the overall font to font. See fonts for more info.

-textsize pointsize

sets the overall default textsize to pointsize. All embedded size specifications will be rendered relative to this.

-linewidth w

sets the overall default linewidth to w. All embedded line width specifications will be rendered relative to this. See linedetails(pli) for more on line width.

-color color

sets the overall default text and line drawing color to color.

-backcolor color

sets the background color to color.

-cm

Use centimeters as your absolute units, instead of inches. On the command line this must appear to the left of any arguments dealing with absolute unit values, such as -pagesize. Centimeter absolute units can also be set via proc settings. If cm will always be the desired absolute units, the preferred way to achieve this is by using units: cm in a ploticus config file.

-inches

Use inches as your absolute units. This is the default.

-outlabel label

Set the label or title for the output. For X11 this sets the window title; for PostScript and SVG it sets the %%Title attribute.

Capacity setting options

These options (new with version 2.10) allow capacities to be raised for accomodation of very large data sets, or lowered to

minimize memory usage. The defaults in this section are defined in pl.h.

-maxrows nrows

Set the capacity for data rows to nrows. Default nrows is 10,000. Ploticus will allocate one pointer for each row.

-maxfields nfields

Set the capacity for data fields to nfields. Default nfields is 200,000. Ploticus will allocate one pointer for each field.

-maxproclines nlines

Set the capacity for script lines for active procs to nlines. Default nlines is 5000. Active procs are the current proc, all #saved procs, and all proc getdata procs that contain embedded data. Ploticus will allocate one pointer for each line in each active proc.

-maxvector ncells

Set the capacity for the data plotting vector to ncells. Default ncells is 100,000. The data plotting vector is an array which holds plottable values for situations where the values must be sorted or pre-screened for bad values. Ploticus will allocate one double value for each cell.

-maxdrawpoints n

Use this if you need to render a polygon having more than 500 points in PNG/GIF/JPEG, X11, or SWF, or any continuous line having more than 500 points in SWF.

Note: raising the maximum number of categories may be done using proc categories from within the script.

-cpulimit #Include nbsp2 s

Set unix resource limit on cpu time to s seconds. Default is 30 seconds. New in 2.40

SVG / XML options:

-svg_tagparms string

This allows arbitrary text to be inserted into the opening <svg> tag.
Example: -svg_tagparms 'height="10cm" width="15cm"'

-omit_xml_declaration

By default the first line of the SVG result will be the XML declaration <?xml .. > . Use this option to suppress the XML declaration line if the SVG result is to be embedded into a larger XML document.

-xml_encoding method

Set the XML character encoding method. This encoding will be indicated in the XML declaration line. The default is iso-8859-1 which provides Latin and Western European character sets. For Unicode fonts this should be set to utf-8 (for more discussion see the Unicode section in fonts).

-tag

Causes a suitable HTML <EMBED> tag to be written to standard output.

-zlevel n

This may be used to set the compression level to n for SVGZ output (0 - 9 where 9 is highest level of compression and the default).

Interactive (workstation) use options

-winloc x,y

Control where on the screen the upper-left corner of the X11 display window will be placed. x and y are in pixels.
Example: -winloc 200 0

-v command
-viewer command

After generating results in the specified format, execute command in order to view the results on your screen. The output file will automatically be included in the command. For example, if a GIF file is being generated you might use this to invoke the xv utility: -viewer xv. If PostScript

is being generated you could use something like this to invoke the ghostview utility: -viewer "gv -magstep -1". The given command must be available on your system and locatable in your command search path. This option may not be used with -o stdout.

-noshell

If specified, ploticus is prohibited from issuing any shell commands. This is a security feature useful for example when running a script that was sent to you by an unknown party. New in 2.31

Paper orientation options

-landscape

For paginated postscript, set paper orientation to landscape (oblong).

-portrait

For paginated postscript, set paper orientation to portrait.

-posteroffset x,y

Allows production of large-size posters made up of multiple standard sheets of paper butted together. May be used only with paginated PostScript, and should be used in combination with the -scale and -textsize options. x,y is the point within your result (in absolute units) that is to be placed at the lower left corner of the page. For further discussion of this, see posters .

Development and debugging options

-debug

Debug mode. Causes dianostic information to be written to the diagnostic stream (stderr by default, see -diagfile below). Highly recommended if you are experiencing difficulty. Best to use -debug as the first (leftmost) argument so that it can report on all arguments gotten. Another effect of debug mode is that any temporary files are not removed upon termination.

-ping

Write the ploticus name and version number to standard output and exit. versions 2.33+

-echo [diag | stdout]

Write ploticus script lines as they are executed. Lines are written to the diagnostic stream (standard error by default) or standard output. Lines are written after variables and most script directives, including flow-of-control directives, are evaluated.

-showbad

Identify unplottable data, showing the value, and its row and field.

-diagfile filename | stderr | stdout

All non-error messages and output will be written to this file (default is stderr).

-errmsgpre tag

Allows developer to set the first portion of all ploticus error messages to tag for purposes of presentation or identification.

-errfile filename | stderr | stdout

All error messages will be written to this file (default is stderr).

-help or -? or -version

Print version number, copyright info, web site address, etc.

Output file names

The output file may be specified on the command line using the -o option, or via Proc Page's outfilename attribute. If so, the result is written to a file of that name. -o stdout may also be used to send result to standard output.

Otherwise, if your script filename has a "recognized extension" (.p, .pl, .plo, .pls, .htm or .html), the base part of the script

file name is used and .png, .gif, etc. is appended. If your script filename doesn't have a recognized extension, the generic name out.* will be used.

X11 output is always displayed on the screen, and paginated PostScript is written to standard output unless -o is used.

If page breaks (Proc Page) are encountered when rendering in any format other than paginated PostScript, special action is necessary since each page must go into a separate file. A Proc Page outfilename may be specified for each page; otherwise a pn prefix will be attached to the beginning of each page's output file name to indicate page n.

If a clickmap is being generated, the result file is named similarly to the above.

Usage examples

The following example uses the scat prefab:

```
ploticus -prefab scat -png datafile=results.dat x=2 y=3
```

The following examples assume that you have a script file called lineplot1.p.

ploticus -x lineplot1.p = view on X11 screen

ploticus -png lineplot1.p = create PNG image lineplot1.png

ploticus -gif lineplot1.p -o stdout = create GIF image on standard output

ploticus -gif lineplot1.p -viewer xv = produce GIF and view using xv (assuming xv image viewer is available on your system).

ploticus -eps lineplot1.p = produce EPS file lineplot1.eps

ploticus -eps lineplot1.p -viewer gv = produce EPS and view using gv (that's ghostview, assuming it is available on your system).

ploticus -eps lineplot1.p -o lineplot.eps = produce EPS into file lineplot.eps

ploticus -ps lineplot1.p | lp = produce paginated post-

script and send to unix lp print spooler.

ploticus -ps lineplot1.p -veiwer gv = produce paginated postscript and view using ghostview.

Environment

PLOTICUS_CONFIG

The name of a ploticus configuration file , for setting default date notations, number notations, measurement units, etc.

PLOTICUS_PREFABS

The path name of a directory where ploticus will look for prefab scripts. The "factory" prefabs are located in the ploticus ./prefabs subdirectory.

LC_CTYPE, LC_COLLATE, LANG

Locale support. Thanks to Oleg Bartunov oleg@sai.msu.su for contributing this. ploticus must be built with -DLOCALE for this to work.

TDH_ERRMODE

Control the disposition of error messages. Allowable values: stderr which is the default, and cgi which causes error messages to be written to stdout with html formatting.

Bugs

Ploticus has some stated limitations (mostly related to capacities that you may run into if you're dealing with large data sets). To report problems or get help see the ploticus support page.

Author, Copyright, Licensing

The primary author is Stephen C. Grubb. Ploticus covered by the General Public License (GPL)... please see the ploticus copyright page for more info.

See also

http://ploticus.sourceforge.net