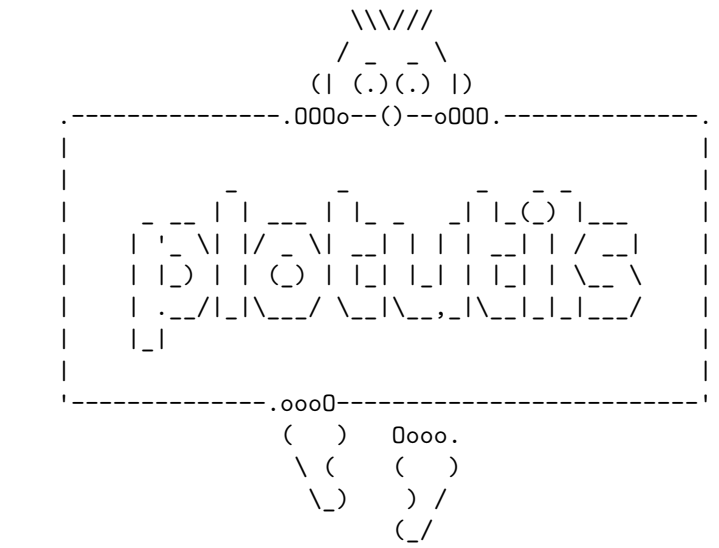

Plotutils

It includes:

- GNU *graph*, which plots 2-D datasets or data streams in real time.
- GNU *plot*, which translates GNU Metafile format to any of the other formats.
- GNU *tek2plot*, for translating legacy Tektronix data to any of the above formats.
- GNU *pic2plot*, for translating the pic language (a scripting language for designing box-and-arrow diagrams) to any of the above formats. The pic language was designed at Bell Labs as an enhancement to the troff text formatter.
- GNU *plotfont*, for displaying character maps of the fonts that are available in the above formats.
- GNU *spline*, which does spline interpolation of data. It normally uses either cubic spline interpolation or exponential splines in tension, but it can function as a real-time filter under some circumstances.
- GNU *ode*, which numerically integrates a system consisting of one or more ordinary differential equations.

Note:

- Imagine only wraps `plot` and `pic2plot` (`pic` is an alias for `pic2plot`).

graph

Each invocation of graph reads one or more datasets from files named on the command line or from standard input, and prepares a plot. There are many command-line options for adjusting the visual appearance of the plot. The following sections explain how to use the most frequently used options, by giving examples.

```
```{.graph im_opt="-X x-axis -Y y-axis -f 0.1 --bitmap-size 200x200" im_out="fcb,img" captio
0.0  0.0
1.0  0.2
2.0  0.0
3.0  0.4
4.0  0.2
5.0  0.6
```
```

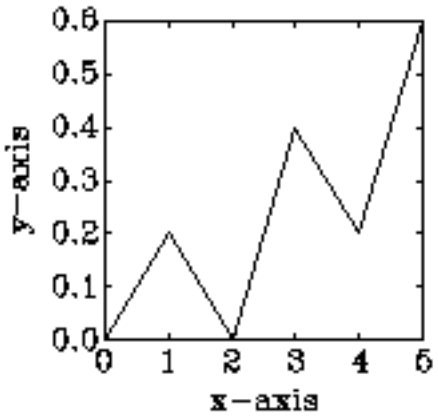

Figure 1: PlotUtil's graph

plot

The GNU *plot* filter displays GNU graphics metafiles or translates them to other formats. It will take input from files specified on the command line or from standard input. The '-T' option is used to specify the desired output format. Supported output formats include "X", "png", "pnm", "gif", "svg", "ai", "ps", "cgm", "fig", "pcl", "hpgl", "regis", "tek", and "meta" (the default).

The metafile format is a device-independent format for storage of vector graphics. By default, it is a binary rather than a human-readable format (see Metafiles). Each of the graph, pic2plot, tek2plot, and plotfont utilities will write a graphics metafile to standard output if no '-T' option is specified on its command line. The GNU libplot graphics library may also be used to produce metafiles. Metafiles may contain arbitrarily many pages of graphics, but each metafile produced by graph contains only a single page.

plot, like the metafile format itself, is useful if you wish to preserve a vector graphics file, and display or edit it with more than one drawing editor.

```{.plot im_opt="--bitmap-size 300x200" im_out="fcb,img" caption="Created by plot"}
dta/input.meta
```

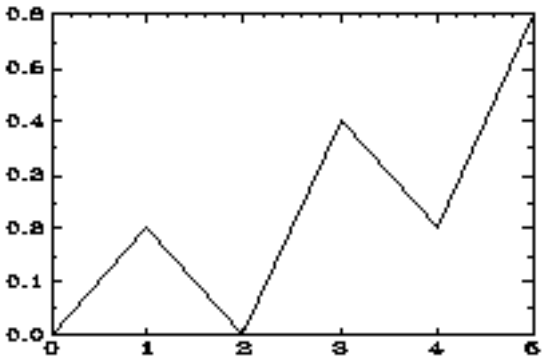

Figure 2: Created by plot

pic2plot

From the gnu website:

The pic language is a 'little language' that was developed at Bell Laboratories for creating box-and-arrow diagrams of the kind frequently found in technical papers and textbooks. A directory containing documentation on the pic language is distributed along with the plotting utilities. On most systems it is installed as /usr/share/pic2plot or /usr/local/share/pic2plot. The directory includes Brian Kernighan's original technical report on the language, Eric S. Raymond's tutorial

on the GNU implementation, and some sample pic macros contributed by the late W. Richard Stevens.

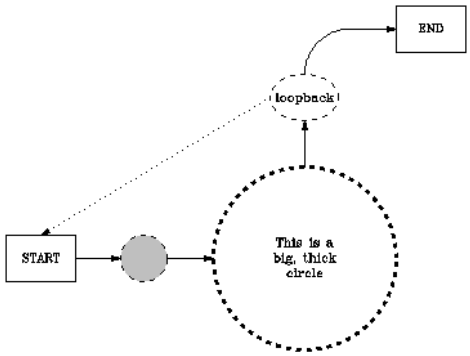

Figure 3: Created by pic

Documentation

Imagine

Codeblock class: graph

```
    sudo apt-get install plotutils
    https://www.gnu.org/software/plotutils
    notes:
    - graphic data is printed to stdout
    - so 'stdout' in im_out option is silently ignored

    runs:
    > graph -T png {im_opt} <fname>.graph
```

```
    class->cmd
      graph -> graph

Metadata options
    imagine.im_out: img,fcb
    imagine.im_log: 4


Codeblock class: plot

    sudo apt-get install plotutils
    https://www.gnu.org/software/plotutils
    notes:
    - graphic data is printed to stdout
    - so 'stdout' in im_out option is silently ignored

    runs:
    > plot -T {im_fmt} {im_opt} <code-text-as-filename>

    class->cmd
      plot -> plot

Metadata options
    imagine.im_out: img,fcb
    imagine.im_log: 4


Codeblock class: pic

    sudo apt-get install plotutils
    https://www.gnu.org/software/plotutils
    notes:
    - graphic data is printed to stdout
    - so 'stdout' in im_out option is silently ignored

    runs:
    > pic2plot -T png {im_opt} <fname>.pic2plot

    class->cmd
      pic2plot -> pic2plot
      pic -> pic2plot

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

graph –help

```
Usage: graph [--output-format | -T arg] [--auto-abscissa | -a [arg(s)]]
 [--clip-mode | -K arg] [--fill-fraction | -q arg]
 [--font-name | -F arg] [--font-size | -f arg]
 [--grid-style | -g arg] [--height-of-plot | -h arg]
 [--input-format | -I arg] [--line-mode | -m arg]
 [--line-width | -W arg] [--right-shift | -r arg]
 [--save-screen | -s] [--symbol | -S [arg(s)]] [--tick-size | -k arg]
 [--toggle-auto-bump | -B] [--toggle-axis-end | -E arg]
 [--toggle-frame-on-top | -H] [--toggle-log-axis | -l arg]
 [--toggle-no-ticks | -N arg] [--toggle-rotate-y-label | -Q]
 [--toggle-round-to-next-tick | -R arg]
 [--toggle-transpose-axes | -t] [--toggle-use-color | -C]
 [--top-label | -L arg] [--upward-shift | -u arg]
 [--width-of-plot | -w arg] [--x-label | -X arg]
 [--x-limits | -x [arg(s)]] [--y-label | -Y arg]
 [--y-limits | -y [arg(s)]] [--bg-color arg] [--bitmap-size arg]
 [--blankout arg] [--emulate-color arg] [--frame-line-width arg]
 [--frame-color arg] [--max-line-length arg] [--pen-colors arg]
 [--reposition arg] [--rotation arg] [--symbol-font-name arg]
 [--title-font-name arg] [--title-font-size arg]
 [--page-size arg] [--portable-output | -O] [--help-fonts]
 [--list-fonts] [--version] [--help] [FILE]...
With no FILE, or when FILE is -, read standard input.

To list available fonts, type `graph -T "format" --help-fonts',
where "format" is the output format, and is one of:
X, png, pnm, or gif (bitmap formats), or
svg, ps, ai, cgm, fig, pcl, hpgl, regis, or tek (vector formats).
The default format is "meta", which is probably not what you want.

Report bugs to bug-plotutils@gnu.org.
```

pic –help

```
usage: pic [ -nvCSU ] [ filename ... ]
       pic -t [ -cvzCSU ] [ filename ... ]
```

plot –help

```
Usage: plot [--output-format | -T arg] [--font-name | -F arg]
 [--font-size | -f arg] [--line-width | -W arg] [--bg-color arg]
 [--bitmap-size arg] [--emulate-color arg]
```

```
 [--max-line-length arg] [--merge-pages | -s]
 [--page-number | -p arg] [--page-size arg] [--pen-color arg]
 [--rotation arg] [--portable-output | -O]
 [--high-byte-first-input | -h] [--low-byte-first-input | -l]
 [--ascii-input | -A] [--help-fonts] [--list-fonts]
 [--version] [--help] [FILE]...
With no FILE, or when FILE is -, read standard input.

To list available fonts, type `plot -T "format" --help-fonts',
where "format" is the output format, and is one of:
X, png, pnm, or gif (bitmap formats), or
svg, ps, ai, cgm, fig, pcl, hpgl, regis, or tek (vector formats).
The default format is "meta", which is probably not what you want.

Report bugs to bug-plotutils@gnu.org.
```

man pages

man graph

```
PLOTUTILS(1)                Debian GNU/Linux                PLOTUTILS(1)

NAME
       plotutils - The GNU plotting utilities.

DESCRIPTION
       plotutils  does  not have a full set of useful man pages.  Most of
       the documentation is available through the info system.  The rele-
       vant command is "info plotutils".

Debian Project               27th July 1998                PLOTUTILS(1)
```

man pic

```
PIC(1)                   General Commands Manual                   PIC(1)

NAME
       pic - compile pictures for troff or TeX

SYNOPSIS
       pic [ -nvCSU ] [ filename ... ]
       pic -t [ -cvzCSU ] [ filename ... ]

DESCRIPTION
```

This manual page describes the GNU version of pic, which is part of the groff document formatting system. pic compiles descriptions of pictures embedded within troff or TeX input files into commands that are understood by TeX or troff. Each picture starts with a line beginning with .PS and ends with a line beginning with .PE. Anything outside of .PS and .PE is passed through without change.

It is the user's responsibility to provide appropriate definitions of the PS and PE macros. When the macro package being used does not supply such definitions (for example, old versions of -ms), appropriate definitions can be obtained with -mpic: These will center each picture.

OPTIONS

Options that do not take arguments may be grouped behind a single -. The special option -- can be used to mark the end of the options. A filename of - refers to the standard input.

-C Recognize .PS and .PE even when followed by a character other than space or newline.

-S Safer mode; do not execute sh commands. This can be useful when operating on untrustworthy input (enabled by default).

-U Unsafe mode; revert the default option -S.

-n Don't use the groff extensions to the troff drawing commands. You should use this if you are using a postprocessor that doesn't support these extensions. The extensions are described in groff_out(5). The -n option also causes pic not to use zero-length lines to draw dots in troff mode.

-t TeX mode.

-c Be more compatible with tpic. Implies -t. Lines beginning with \ are not passed through transparently. Lines beginning with . are passed through with the initial . changed to \. A line beginning with .ps is given special treatment: it takes an optional integer argument specifying the line thickness (pen size) in milliinches; a missing argument restores the previous line thickness; the default line thickness is 8 milliinches. The line thickness thus specified takes effect only when a non-negative line thickness has not been specified by use of the thickness attribute or

by setting the linethick variable.

-v Print the version number.

-z In TeX mode draw dots using zero-length lines.

The following options supported by other versions of pic are ignored:

-D Draw all lines using the \D escape sequence. pic always does this.

-T dev Generate output for the troff device dev. This is unnecessary because the troff output generated by pic is device-independent.

USAGE

This section describes only the differences between GNU pic and the original version of pic. Many of these differences also apply to newer versions of Unix pic. A complete documentation is available in the file

/usr/share/doc/groff-base/pic.ms.gz

TeX mode

TeX mode is enabled by the -t option. In TeX mode, pic will define a vbox called \graph for each picture. Use the figname command to change the name of the vbox. You must yourself print that vbox using, for example, the command

```
\centerline{\box\graph}
```

Actually, since the vbox has a height of zero (it is defined with \vtop) this will produce slightly more vertical space above the picture than below it;

```
\centerline{\raise 1em\box\graph}
```

would avoid this.

To make the vbox having a positive height and a depth of zero (as used e.g. by LaTeX's graphics.sty), define the following macro in your document:

```
\def\gpicbox#1{%
   \vbox{\unvbox\csname #1\endcsname\kern 0pt}}
```

Now you can simply say \gpicbox{graph} instead of \box\graph.

You must use a TeX driver that supports the tpic specials, version 2.

Lines beginning with \ are passed through transparently; a % is added to the end of the line to avoid unwanted spaces. You can safely use this feature to change fonts or to change the value of \baselineskip. Anything else may well produce undesirable results; use at your own risk. Lines beginning with a period are not given any special treatment.

Commands

for variable = expr1 to expr2 [by [*]expr3] do X body X

Set variable to expr1. While the value of variable is less than or equal to expr2, do body and increment variable by expr3; if by is not given, increment variable by 1. If expr3 is prefixed by * then variable will instead be multiplied by expr3. The value of expr3 can be negative for the additive case; variable is then tested whether it is greater than or equal to expr2. For the multiplicative case, expr3 must be greater than zero. If the constraints aren't met, the loop isn't executed. X can be any character not occurring in body.

if expr then X if-true X [else Y if-false Y]

Evaluate expr; if it is non-zero then do if-true, otherwise do if-false. X can be any character not occurring in if-true. Y can be any character not occurring in if-false.

print arg...

Concatenate the arguments and print as a line on stderr. Each arg must be an expression, a position, or text. This is useful for debugging.

command arg...

Concatenate the arguments and pass them through as a line to troff or TeX. Each arg must be an expression, a position, or text. This has a similar effect to a line beginning with . or \, but allows the values of variables to be passed through. For example,

```
.PS
x = 14
command ".ds string x is " x "."
```

```
.PE
\*[string]
```

prints

```
x is 14.
```

sh X command X
: Pass command to a shell. X can be any character not occurring in command.

copy "filename"
: Include filename at this point in the file.

copy ["filename"] thru X body X [until "word"]
copy ["filename"] thru macro [until "word"]
: This construct does body once for each line of filename; the line is split into blank-delimited words, and occurrences of $i in body, for i between 1 and 9, are replaced by the i-th word of the line. If filename is not given, lines are taken from the current input up to .PE. If an until clause is specified, lines will be read only until a line the first word of which is word; that line will then be discarded. X can be any character not occurring in body. For example,

```
.PS
copy thru % circle at ($1,$2) % until "END"
1 2
3 4
5 6
END
box
.PE
```

is equivalent to

```
.PS
circle at (1,2)
circle at (3,4)
circle at (5,6)
box
.PE
```

The commands to be performed for each line can also be taken from a macro defined earlier by giving the name of

the macro as the argument to thru.

reset
reset variable1[,] variable2 ...
Reset pre-defined variables variable1, variable2 ... to their default values. If no arguments are given, reset all pre-defined variables to their default values. Note that assigning a value to scale also causes all pre-defined variables that control dimensions to be reset to their default values times the new value of scale.

plot expr ["text"]
This is a text object which is constructed by using text as a format string for sprintf with an argument of expr. If text is omitted a format string of "%g" is used. Attributes can be specified in the same way as for a normal text object. Be very careful that you specify an appropriate format string; pic does only very limited checking of the string. This is deprecated in favour of sprintf.

variable := expr
This is similar to = except variable must already be defined, and expr will be assigned to variable without creating a variable local to the current block. (By contrast, = defines the variable in the current block if it is not already defined there, and then changes the value in the current block only.) For example, the following:

```
.PS
x = 3
y = 3
[
  x := 5
  y = 5
]
print x " " y
.PE
```

prints

```
5 3
```

Arguments of the form

X anything X

are also allowed to be of the form

```
{ anything }
```

In this case anything can contain balanced occurrences of { and }. Strings may contain X or imbalanced occurrences of { and }.

Expressions

The syntax for expressions has been significantly extended:

```
x ^ y (exponentiation)
sin(x)
cos(x)
atan2(y, x)
log(x) (base 10)
exp(x) (base 10, i.e. 10^x)
sqrt(x)
int(x)
rand() (return a random number between 0 and 1)
rand(x) (return a random number between 1 and x; deprecated)
srand(x) (set the random number seed)
max(e1, e2)
min(e1, e2)
!e
e1 && e2
e1 || e2
e1 == e2
e1 != e2
e1 >= e2
e1 > e2
e1 <= e2
e1 < e2
"str1" == "str2"
"str1" != "str2"
```

String comparison expressions must be parenthesised in some contexts to avoid ambiguity.

Other Changes

A bare expression, expr, is acceptable as an attribute; it is equivalent to dir expr, where dir is the current direction. For example

```
line 2i
```

means draw a line 2 inches long in the current direction. The 'i'

(or 'I') character is ignored; to use another measurement unit, set the scale variable to an appropriate value.

The maximum width and height of the picture are taken from the variables maxpswid and maxpsht. Initially these have values 8.5 and 11.

Scientific notation is allowed for numbers. For example

```
x = 5e-2
```

Text attributes can be compounded. For example,

```
"foo" above ljust
```

is valid.

There is no limit to the depth to which blocks can be examined. For example,

```
[A: [B: [C: box ]]] with .A.B.C.sw at 1,2
circle at last [].A.B.C
```

is acceptable.

Arcs now have compass points determined by the circle of which the arc is a part.

Circles, ellipses, and arcs can be dotted or dashed. In TeX mode splines can be dotted or dashed also.

Boxes can have rounded corners. The rad attribute specifies the radius of the quarter-circles at each corner. If no rad or diam attribute is given, a radius of boxrad is used. Initially, boxrad has a value of 0. A box with rounded corners can be dotted or dashed.

Boxes can have slanted sides. This effectively changes the shape of a box from a rectangle to an arbitrary parallelogram. The xslanted and yslanted attributes specify the x and y offset of the box's upper right corner from its default position.

The .PS line can have a second argument specifying a maximum height for the picture. If the width of zero is specified the width will be ignored in computing the scaling factor for the picture. Note that GNU pic will always scale a picture by the same

amount vertically as well as horizontally. This is different from the DWB 2.0 pic which may scale a picture by a different amount vertically than horizontally if a height is specified.

Each text object has an invisible box associated with it. The compass points of a text object are determined by this box. The implicit motion associated with the object is also determined by this box. The dimensions of this box are taken from the width and height attributes; if the width attribute is not supplied then the width will be taken to be textwid; if the height attribute is not supplied then the height will be taken to be the number of text strings associated with the object times textht. Initially textwid and textht have a value of 0.

In (almost all) places where a quoted text string can be used, an expression of the form

```
sprintf("format", arg,...)
```

can also be used; this will produce the arguments formatted according to format, which should be a string as described in printf(3) appropriate for the number of arguments supplied.

The thickness of the lines used to draw objects is controlled by the linethick variable. This gives the thickness of lines in points. A negative value means use the default thickness: in TeX output mode, this means use a thickness of 8 milliinches; in TeX output mode with the -c option, this means use the line thickness specified by .ps lines; in troff output mode, this means use a thickness proportional to the pointsize. A zero value means draw the thinnest possible line supported by the output device. Initially it has a value of -1. There is also a thick[ness] attribute. For example,

```
circle thickness 1.5
```

would draw a circle using a line with a thickness of 1.5 points. The thickness of lines is not affected by the value of the scale variable, nor by the width or height given in the .PS line.

Boxes (including boxes with rounded corners or slanted sides), circles and ellipses can be filled by giving them an attribute of fill[ed]. This takes an optional argument of an expression with a value between 0 and 1; 0 will fill it with white, 1 with black, values in between with a proportionally gray shade. A value greater than 1 can also be used: this means fill with the shade of

gray that is currently being used for text and lines. Normally this will be black, but output devices may provide a mechanism for changing this. Without an argument, then the value of the variable fillval will be used. Initially this has a value of 0.5. The invisible attribute does not affect the filling of objects. Any text associated with a filled object will be added after the object has been filled, so that the text will not be obscured by the filling.

Three additional modifiers are available to specify colored objects: outline[d] sets the color of the outline, shaded the fill color, and colo[u]r[ed] sets both. All three keywords expect a suffix specifying the color, for example

```
circle shaded "green" outline "black"
```

Currently, color support isn't available in TeX mode. Predefined color names for groff are in the device macro files, for example ps.tmac; additional colors can be defined with the .defcolor request (see the manual page of troff(1) for more details).

To change the name of the vbox in TeX mode, set the pseudo-variable figname (which is actually a specially parsed command) within a picture. Example:

```
.PS
figname = foobar;
...
.PE
```

The picture is then available in the box \foobar.

pic assumes that at the beginning of a picture both glyph and fill color are set to the default value.

Arrow heads will be drawn as solid triangles if the variable arrowhead is non-zero and either TeX mode is enabled or the -n option has not been given. Initially arrowhead has a value of 1. Note that solid arrow heads are always filled with the current outline color.

The troff output of pic is device-independent. The -T option is therefore redundant. All numbers are taken to be in inches; numbers are never interpreted to be in troff machine units.

Objects can have an aligned attribute. This will only work if the

postprocessor is grops, or gropdf. Any text associated with an object having the aligned attribute will be rotated about the center of the object so that it is aligned in the direction from the start point to the end point of the object. Note that this attribute will have no effect for objects whose start and end points are coincident.

In places where nth is allowed 'expr'th is also allowed. Note that 'th is a single token: no space is allowed between the ' and the th. For example,

```
for i = 1 to 4 do {
   line from 'i'th box.nw to 'i+1'th box.se
}
```

CONVERSION

To obtain a stand-alone picture from a pic file, enclose your pic code with .PS and .PE requests; roff configuration commands may be added at the beginning of the file, but no roff text.

It is necessary to feed this file into groff without adding any page information, so you must check which .PS and .PE requests are actually called. For example, the mm macro package adds a page number, which is very annoying. At the moment, calling standard groff without any macro package works. Alternatively, you can define your own requests, e.g. to do nothing:

```
.de PS
..
.de PE
..
```

groff itself does not provide direct conversion into other graphics file formats. But there are lots of possibilities if you first transform your picture into PostScript(R) format using the groff option -Tps. Since this ps-file lacks BoundingBox information it is not very useful by itself, but it may be fed into other conversion programs, usually named ps2other or pstoother or the like. Moreover, the PostScript interpreter ghostscript (gs) has built-in graphics conversion devices that are called with the option

```
gs -sDEVICE=<devname>
```

Call

```
       gs --help

for a list of the available devices.

An alternative may be to use the -Tpdf option to convert your pic-
ture directly into PDF format.  The MediaBox of the file  produced
can be controlled by passing a -P-p papersize to groff.

As the Encapsulated PostScript File Format EPS is getting more and
more important, and the conversion wasn't regarded trivial in  the
past  you  might  be interested to know that there is a conversion
tool named ps2eps which does the right job.  It  is  much  better
than the tool ps2epsi packaged with gs.

For bitmapped graphic formats, you should use pstopnm; the result-
ing (intermediate) PNM file can be then converted to virtually any
graphics format using the tools of the netpbm package .

FILES
       /usr/share/groff/1.22.3/tmac/pic.tmac
              Example definitions of the PS and PE macros.

SEE ALSO
       troff(1),  groff_out(5),  tex(1),  gs(1),  ps2eps(1),  pstopnm(1),
       ps2epsi(1), pnm(5)

       Eric S. Raymond, Making Pictures With GNU PIC.
       /usr/share/doc/groff-base/pic.ps (this  file,  together  with  its
       source file, pic.ms, is part of the groff documentation)

       Tpic: Pic for TeX

       Brian  W.  Kernighan,  PIC  --  A Graphics Language for Typesetting
       (User Manual).  AT&T Bell Laboratories, Computing Science  Techni-
       cal Report No. 116
       <http://cm.bell-labs.com/cm/cs/cstr/116.ps.gz>    (revised    May,
       1991).

       ps2eps is available from CTAN mirrors, e.g.
       <ftp://ftp.dante.de/tex-archive/support/ps2eps/>

       W. Richard Stevens, Turning PIC Into HTML
       <http://www.kohala.com/start/troff/pic2html.html>

       W. Richard Stevens, Examples of picMacros
       <http://www.kohala.com/start/troff/pic.examples.ps>
```

BUGS

Input characters that are invalid for groff (i.e., those with ASCII code 0, or 013 octal, or between 015 and 037 octal, or between 0200 and 0237 octal) are rejected even in TeX mode.

The interpretation of fillval is incompatible with the pic in 10th edition Unix, which interprets 0 as black and 1 as white.

PostScript(R) is a registered trademark of Adobe Systems Incorporation.

COPYING

man plot

PLOT(1) GNU Plotting Utilities PLOT(1)

NAME

plot - translate GNU metafiles to other graphics formats

SYNOPSIS

plot [options] [files]

DESCRIPTION

plot translates files in GNU metafile format to other graphics formats, or displays them on an X Window System display. GNU

metafile format is a device-independent format for the storage of graphic data. It is the default output format of the programs graph(1), pic2plot(1), tek2plot(1), and plotfont(1), and is further documented in plot(5), since it is an enhanced version of the traditional plot(5) format found on non-GNU systems. It can also be produced by the GNU libplot 2-D graphics export library (see plot(3)).

The output format is specified with the -T option. The possible output formats and display types are the same as those supported by graph(1), plotfont(1), pic2plot(1), and tek2plot(1). If an output file is produced, it is written to standard output.

Options and file names may be interspersed on the command line, but the options are processed before the file names are read. If -- is seen, it is interpreted as the end of the options. If no file names are specified, or the file name - is encountered, the standard input is read.

OPTIONS

General Options

-T type

--output-format type

Select type as the output format. It may be "X", "png", "pnm", "gif", "svg", "ai", "ps", "cgm", "fig", "pcl", "hpgl", "regis", "tek", or "meta" (the default). These refer respectively to the X Window System, PNG (Portable Network Graphics) format, portable anymap format (PBM/PGM/PPM), a pseudo-GIF format that does not use LZW encoding, the new XML-based Scalable Vector Graphics format, the format used by Adobe Illustrator, Postscript or Encapsulated Postscript (EPS) that can be edited with idraw(1), CGM format (by default, confirming to the WebCGM profile), the format used by the xfig(1) drawing editor, the Hewlett-Packard PCL 5 printer language, the Hewlett-Packard Graphics Language, ReGIS graphics format (which can be displayed by the dxterm(1) terminal emulator or by a VT330 or VT340 terminal), Tektronix format (which can be displayed by the xterm(1) terminal emulator), and device-independent GNU metafile format itself. Unless type is "X", an output file is produced and written to standard output.

Omitting the -T option is equivalent to specifying -T meta. Translating from metafile format to itself is occasionally useful, since there are two versions of metafile format

(see the -O option below).

A listing of the fonts available in any specified output format may be obtained with the --help-fonts option (see below). If a requested font is unavailable, a default font will be substituted. The default font is "Helvetica" for "X", "svg", "ai", "ps", "cgm", and "fig", "Univers" for "pcl", and "HersheySerif" for "png", "pnm", "gif", "hpgl", "regis", "tek", and "meta".

-p n
--page-number n
Output only page number n, within the metafile or sequence of metafiles that is being translated.

Metafiles may consist of one or more pages, numbered beginning with 1. Also, each page may contain multiple `frames'. plot -T X, plot -T regis, and plot -T tek, which plot in real time, will separate successive frames by screen erasures. plot -T png, plot -T pnm, plot -T gif, plot -T svg, plot -T ai, plot -T ps, plot -T cgm, plot -T fig, plot -T pcl, and plot -T hpgl, which do not plot in real time, will output only the last frame of any multi-frame page.

The default behavior, if -p is not used, is to output all pages. For example, plot -T X displays each page in its own X window. If the -T png, -T pnm, -T gif, -T ai, or -T fig option is used, the default behavior is to output only the first nonempty page, since files in those output formats contain only a single page of graphics.

Metafiles produced by graph(1) and plotfont(1) contain only a single page (page #1), which consists of two frames: an empty frame to clear the display, and a second frame that contains the graphics.

-s
--merge-pages
Merge all displayed pages into a single page, and also merge all `frames'.

This option is useful when merging together single-page plots from different sources. For example, it can be used to merge together plots obtained from separate invocations of graph(1).

--bitmap-size bitmap_size
Set the size of the graphics display in which the plot will be drawn, in terms of pixels, to be bitmap_size. The default is "570x570". This is relevant only to plot -T X, plot -T png, plot -T pnm, and plot -T gif, all of which produce bitmaps. If you choose a rectangular (non-square) window size, the fonts in the plot will be scaled anisotropically, i.e., by different factors in the horizontal and vertical directions. For plot -T X, this requires an X11R6 display. Any font that cannot be scaled in this way will be replaced by a default scalable font, such as the vector font "HersheySerif".

The environment variable BITMAPSIZE can equally well be used to specify the window size. For backward compatibility, the X resource Xplot.geometry may be used instead.

--emulate-color option
If option is yes, replace each color in the output by an appropriate shade of gray. This is seldom useful, except when using plot -T pcl to prepare output for a PCL 5 device. (Many monochrome PCL 5 devices, such as monochrome LaserJets, do a poor job of emulating color on their own.) You may equally well request color emulation by setting the environment variable EMULATE_COLOR to "yes".

--max-line-length max_line_length
Set the maximum number of points that a polygonal line may contain, before it is flushed out, to be max_line_length. If this flushing occurs, the polygonal line will be split into two or more sub-lines, though the splitting should not be noticeable. The default value of max_line_length is 500.

The reason for splitting long polygonal lines is that some display devices (e.g., old Postscript printers and pen HP-GL plotters) have limited buffer sizes. The environment variable MAX_LINE_LENGTH can also be used to specify the maximum line length.

--page-size pagesize
Set the size of the page on which the plot will be positioned. This is relevant only to plot -T svg, plot -T ai, plot -T ps, plot -T cgm, plot -T fig, plot -T pcl, and plot -T hpgl. The default is "letter", which means an 8.5 inch

by 11 inch page. Any ISO page size in the range "a0"..."a4" or ANSI page size in the range "a"..."e" may be specified ("letter" is an alias for "a" and "tabloid" is an alias for "b"). "legal" and "ledger" are recognized page sizes also. The environment variable PAGESIZE can equally well be used to specify the page size.

The graphics display in which the plot is drawn will, by default, be a square region that occupies nearly the full width of the specified page. An alternative size for the graphics display can be specified. For example, the page size could be specified as "letter,xsize=4in,ysize=6in", or "a4,xsize=5.0cm,ysize=100mm". For all of the above except plot -T hpgl, the graphics display will, by default, be centered on the page. For all of the above except plot -T svg and plot -T cgm, the graphics display may be repositioned manually, by specifying the location of its lower left corner, relative to the lower left corner of the page. For example, the page size could be specified as "letter,xorigin=2in,yorigin=3in", or "a4,xorigin=0.5cm,yorigin=0.5cm". It is also possible to specify an offset vector. For example, the page size could be specified as "letter,xoffset=1in", or "letter,xoffset=1in,yoffset=1.2in", or "a4,yoffset=-1cm". In SVG format and WebCGM format it is possible to specify the size of the graphics display, but not its position.

--rotation angle
Rotate the graphics display by angle degrees. Recognized values are "0", "90", "180", and "270". "no" and "yes" are equivalent to "0" and "90", respectively. The environment variable ROTATION can also be used to specify a rotation angle.

Parameter Initialization Options

The following options set the initial values of drawing parameters. However, all of these may be overridden by directives in a metafile. In fact, these options are useful primarily when plotting old metafiles in the traditional (pre-GNU) plot(5) format, which did not support such directives.

--bg-color name
Set the color initially used for the background to be name. This is relevant only to plot -T X, plot -T png, plot -T pnm, plot -T gif, plot -T svg, plot -T cgm, and plot -T regis. An unrecognized name sets the color to the default,

which is "white". The environment variable BG_COLOR can equally well be used to specify the background color.

If the -T png or -T gif option is used, a transparent PNG file or a transparent pseudo-GIF, respectively, may be produced by setting the TRANSPARENT_COLOR environment variable to the name of the background color. If the -T svg or -T cgm option is used, an output file without a background may be produced by setting the background color to "none".

-f size
--font-size size
Set the size of the font initially used for rendering text, as a fraction of the width of the graphics display, to be size. The default is 0.0525.

-F name
--font-name name
Set the font initially used for text to be name. Font names are case-insensitive. If the specified font is not available, the default font will be used. Which fonts are available, and the default font, depend on which -T option is specified (see above). A list of available fonts can be obtained with the --help-fonts option (see below).

-W line_width
--line-width line_width
Set the initial width of lines, as a fraction of the width of the display, to be line_width. A negative value means that a default value should be used. This value is format-dependent. The interpretation of zero line width is also format-dependent (in some output formats, a zero-width line is the thinnest line that can be drawn; in others, a zero-width line is invisible).

--pen-color name
Set the initial pen color to be name. An unrecognized name sets the pen color to the default, which is "black".

Options for Metafile Output
The following option is relevant only if the -T option is omitted or if -T meta is used. In this case the output of plot, like the input, will be in GNU graphics metafile format.

-O
--portable-output

Output the portable (human-readable) version of GNU metafile format, rather than the binary version (the default). The format of the binary version is machine-dependent.

Options for Backward Compatibility

By default, plot assumes that its input file(s) are in either the binary version or the portable version of GNU metafile format. You may specify that the input is, instead, in the traditional Unix (pre-GNU) graphics metafile format, which is documented in plot(5). The traditional graphics metafile format was produced by pre-GNU versions of graph(1).

-h
--high-byte-first-input
Input file(s) are assumed to be in the binary, `high byte first' version of traditional metafile format. This variant is uncommon.

-l
--low-byte-first-input
Input file(s) are assumed to be in the binary, `low byte first' version of traditional metafile format. This variant is the most common.

-A
--ascii-input
Input file(s) are assumed to be in the ASCII (human-readable) variant of traditional metafile format. On some older Unix systems, this variant was produced by plotoa(1).

Informational Options

--help Print a list of command-line options, and exit.

--help-fonts
Print a table of available fonts, and exit. The table will depend on which output format is specified with the -T option. plot -T X, plot -T svg, plot -T ai, plot -T ps, plot -T cgm, and plot -T fig each support the 35 standard Postscript fonts. plot -T svg, plot -T pcl, and plot -T hpgl support the 45 standard PCL 5 fonts, and the latter two support a number of Hewlett-Packard vector fonts. All seven support a set of 22 Hershey vector fonts, as do plot -T png, plot -T pnm, plot -T gif, plot -T regis, and plot -T tek. plot without a -T option in principle supports any

of these fonts, since its output must be translated to other formats by a further invocation of plot.

The plotfont(1) utility may be used to obtain a character map of any supported font.

--list-fonts
Like --help-fonts, but lists the fonts in a single column to facilitate piping to other programs. If no output format is specified with the -T option, the full set of supported fonts is listed.

--version
Print the version number of plot and the plotting utilities package, and exit.

ENVIRONMENT

The environment variables BITMAPSIZE, PAGESIZE, BG_COLOR, EMULATE_COLOR, MAX_LINE_LENGTH and ROTATION serve as backups for the options --bitmap-size, --page-size, --bg-color, --emulate-color, --max-line-length, and --rotation, respectively. The remaining environment variables are specific to individual output formats.

plot -T X, which pops up a window on an X Window System display and draws graphics in it, checks the DISPLAY environment variable. Its value determines the display that will be used.

plot -T png and plot -T gif, which produce output in PNG format and pseudo-GIF format respectively, are affected by the INTERLACE environment variable. If its value is "yes", the output will be interlaced. Also, if the TRANSPARENT_COLOR environment variable is set to the name of a color, that color will be treated as transparent in the output.

plot -T pnm, which produces output in portable anymap (PBM/PGM/PPM) format, is affected by the PNM_PORTABLE environment variable. If its value is "yes", the output will be in a human-readable format rather than binary (the default).

plot -T cgm, which produces output in CGM (Computer Graphics Metafile) format, is affected by the CGM_MAX_VERSION and CGM_ENCODING environment variables. By default, it produces a binary-encoded version of CGM version 3 format. For backward compatibility, the version number may be reduced by setting CGM_MAX_VERSION to "2" or "1". Irrespective of version, the output CGM file will use the human-readable clear text encoding if

CGM_ENCODING is set to "clear_text". However, only binary-encoded CGM files conform to the WebCGM profile.

plot -T pcl, which produces PCL 5 output for Hewlett-Packard printers and plotters, is affected by the environment variable PCL_ASSIGN_COLORS. It should be set to "yes" when producing PCL 5 output for a color printer or other color device. This will ensure accurate color reproduction by giving the output device complete freedom in assigning colors, internally, to its "logical pens". If it is "no" then the device will use a fixed set of colored pens, and will emulate other colors by shading. The default is "no" because monochrome PCL 5 devices, which are much more common than colored ones, must use shading to emulate color.

plot -T hpgl, which produces Hewlett-Packard Graphics Language output, is affected by several environment variables. The most important is HPGL_VERSION, which may be set to "1", "1.5", or "2" (the default). "1" means that the output should be generic HP-GL, "1.5" means that the output should be suitable for the HP7550A graphics plotter and the HP758x, HP7595A and HP7596A drafting plotters (HP-GL with some HP-GL/2 extensions), and "2" means that the output should be modern HP-GL/2. If the version is "1" or "1.5" then the only available fonts will be vector fonts, and all lines will be drawn with a default width (the -W option will not work). Additionally, if the version is "1" then the filling of arbitrary curves with solid color will not be supported (circles and rectangles aligned with the coordinate axes may be filled, though).

The position of the plot -T hpgl graphics display on the page can be rotated 90 degrees counterclockwise by setting the HPGL_ROTATE environment variable to "yes". This is not the same as the rotation obtained with the --rotation option, since it both rotates the graphics display and repositions its lower left corner toward another corner of the page. Besides "no" and "yes", recognized values for HPGL_ROTATE are "0", "90", "180", and "270". "no" and "yes" are equivalent to "0" and "90", respectively. "180" and "270" are supported only if HPGL_VERSION is "2" (the default).

By default, plot -T hpgl will draw with a fixed set of pens. Which pens are present may be specified by setting the HPGL_PENS environment variable. If HPGL_VERSION is "1", the default value of HPGL_PENS is "1=black"; if HPGL_VERSION is "1.5" or "2", the default value of HPGL_PENS is "1=black:2=red:3=green:4=yellow:5=blue:6=magenta:7=cyan". The format should be self-explanatory. By setting HPGL_PENS you may specify a color for any pen in

the range #1...#31. All color names recognized by the X Window System may be used. Pen #1 must always be present, though it need not be black. Any other pen in the range #1...#31 may be omitted.

If HPGL_VERSION is "2" then plot -T hpgl will also be affected by the environment variable HPGL_ASSIGN_COLORS. If its value is "yes", then plot -T hpgl will not be restricted to the palette specified in HPGL_PENS: it will assign colors to "logical pens" in the range #1...#31, as needed. The default value is "no" because other than color LaserJet printers and DesignJet plotters, not many HP-GL/2 devices allow the assignment of colors to logical pens.

Opaque filling and the drawing of visible white lines are supported only if HPGL_VERSION is "2" and the environment variable HPGL_OPAQUE_MODE is "yes" (the default). If its value is "no" then white lines (if any), which are normally drawn with pen #0, will not be drawn. This feature is to accommodate older HP-GL/2 devices. HP-GL/2 pen plotters, for example, do not support opacity or the use of pen #0 to draw visible white lines. Some older HP-GL/2 devices may, in fact, malfunction if asked to draw opaque objects.

plot -T tek, which produces output for a Tektronix terminal or emulator, checks the TERM environment variable. If the value of TERM is a string beginning with "xterm", "nxterm", or "kterm", it is taken as a sign that plot is running in an X Window System VT100 terminal emulator: a copy of xterm(1), nxterm(1), or kterm(1). Before drawing graphics, plot -T tek will emit an escape sequence that causes the terminal emulator's auxiliary Tektronix window, which is normally hidden, to pop up. After the graphics are drawn, an escape sequence that returns control to the original VT100 window will be emitted. The Tektronix window will remain on the screen.

If the value of TERM is a string beginning with "kermit", "ansi.sys", or "nansi.sys", it is taken as a sign that plot is running in the VT100 terminal emulator provided by the MS-DOS version of kermit(1). Before drawing graphics, plot -T tek will emit an escape sequence that switches the terminal emulator to Tektronix mode. Also, some of the Tektronix control codes emitted by plot -T tek will be kermit-specific. There will be a limited amount of color support, which is not normally the case (the 16 `ansi.sys' colors will be supported). After drawing graphics, plot -T tek will emit an escape sequence that returns the emulator to VT100 mode. The key sequence `ALT minus' can be employed manu-

ally within kermit to switch between the two modes.

SEE ALSO

graph(1), pic2plot(1), tek2plot(1), plotfont(1), plot(3), plot(5), and "The GNU Plotting Utilities Manual".

AUTHORS

plot was written by Robert S. Maier (rsm@math.arizona.edu).

BUGS

Email bug reports to bug-gnu-utils@gnu.org.