

# gnuplot FAQ

This material describes **gnuplot** version 6

FAQ version: December 2023

PDF version of current document: <http://www.gnuplot.info/faq/faq.pdf>.

## Contents

<b>0</b>	<b>Meta – Questions</b>	<b>4</b>
0.1	Where do I get this document? . . . . .	4
0.2	Where do I send comments about this document? . . . . .	4
<b>1</b>	<b>General Information</b>	<b>4</b>
1.1	What is <b>gnuplot</b> ? . . . . .	4
1.2	How did it come about and why is it called <b>gnuplot</b> ? . . . . .	4
1.3	What does <b>gnuplot</b> offer? . . . . .	4
1.4	Is <b>gnuplot</b> suitable for scripting? . . . . .	5
1.5	Can I run <b>gnuplot</b> on my computer? . . . . .	5
1.6	Legalities . . . . .	5
1.7	Does <b>gnuplot</b> have anything to do with the FSF and the GNU project? . . . . .	5
1.8	Where do I get further information? . . . . .	5
<b>2</b>	<b>Setting it up</b>	<b>6</b>
2.1	What is the current version of <b>gnuplot</b> ? . . . . .	6
2.2	Where can I get <b>gnuplot</b> ? . . . . .	6
2.3	Why would I care about the development version? . . . . .	6
2.4	Where can I get current development version of <b>gnuplot</b> ? . . . . .	6
2.5	How do I compile <b>gnuplot</b> on my system? . . . . .	6
2.6	What documentation is there, and how do I get it? . . . . .	7
2.7	Worked examples . . . . .	7
2.8	How do I determine which options were compiled into my <b>gnuplot</b> executable? . . . . .	7
<b>3</b>	<b>Working with it.</b>	<b>7</b>
3.1	How do I get help? . . . . .	7

3.2	How do I print out my graphs? . . . . .	8
3.3	How do I include my graphs in my favorite word processor? . . . . .	8
3.4	How do I edit or post-process a <b>gnuplot</b> graph? . . . . .	9
3.5	How do I save and restore my current settings? . . . . .	9
3.6	Can I put both commands and data into a single file? . . . . .	9
3.7	How do I run my data through a filter before plotting? . . . . .	10
3.8	Can I use <b>gnuplot</b> routines for my own programs? . . . . .	10
<b>4</b>	<b>Customizing the appearance of your plot</b>	<b>10</b>
4.1	How to inspect or change the default colors, line, and point properties? . . . . .	10
4.2	Hidden line or surface removal . . . . .	10
4.3	How do I force exact positions for the graph borders on the page? . . . . .	10
4.4	Arranging multiple plots next to each other on a single page . . . . .	11
4.5	How to request 1:1 scaling of axes? . . . . .	11
4.6	Palette that works for both color and black&white printing? . . . . .	11
4.7	How do I skip data points? . . . . .	11
4.8	How do I plot every nth point? . . . . .	11
4.9	How do I plot a vertical line? . . . . .	11
4.10	Y axis label in the wrong place and/or left margin too big . . . . .	11
<b>5</b>	<b>Plot types that people often ask about</b>	<b>12</b>
5.1	Animations . . . . .	12
5.2	Implicit defined graphs . . . . .	12
5.3	How to fill an area between two functions . . . . .	12
5.4	Drawing a 2D projection of 3D data . . . . .	12
5.5	How to overlay dots/points scatter plot onto a pm3d map/surface . . . . .	13
5.6	How to produce labeled contours . . . . .	13
5.7	Does <b>gnuplot</b> support bar-charts/histograms/boxes? . . . . .	13
5.8	Does <b>gnuplot</b> support pie charts? quarterly time charts? . . . . .	13
5.9	Does <b>gnuplot</b> support multiple y-axes on a single plot? . . . . .	13
5.10	How to draw a solid made of triangular facets? . . . . .	13
5.11	How do I plot two functions in non-overlapping regions? . . . . .	14
5.12	How do I plot lines (not grids) using splot? . . . . .	14
5.13	How do I plot a function $f(x,y)$ that is bounded by other functions in the x-y plane? . . . . .	14
<b>6</b>	<b>Text formatting and special symbols</b>	<b>14</b>
6.1	Text markup using "enhanced text" mode . . . . .	14
6.2	How do I turn text markup on or off? . . . . .	15

6.3	Is UTF-8 the answer to all my special character problems? . . . . .	15
6.4	What if I need h-bar (Planck's constant)? . . . . .	15
6.5	What if I need the Solar mass symbol? . . . . .	16
6.6	How to use Greek letters or other special symbols? . . . . .	16
6.7	How do I include accented characters? . . . . .	16
6.8	Can I put different text sizes into my plots? . . . . .	16
<b>7</b>	<b>Miscellaneous</b>	<b>16</b>
7.1	Does <b>gnuplot</b> support a driver for graphics format XXX? . . . . .	16
7.2	I've found a bug, what do I do? . . . . .	16
7.3	Can I do heavy-duty data processing with <b>gnuplot</b> ? or What is beyond <b>gnuplot</b> ? . . . . .	17
7.4	What if I need special function that gnuplot doesn't have? . . . . .	17
7.5	How to use hotkeys in my interactive terminals . . . . .	17
7.6	I want to help in developing the next version of <b>gnuplot</b> . What can I do? . . . . .	17
<b>8</b>	<b>Common problems</b>	<b>18</b>
8.1	Help! None of my fonts work. . . . .	18
8.2	The first plot in a qt terminal session fails or has bad layout . . . . .	18
8.3	Pm3d splot from a datafile does not draw anything . . . . .	18
8.4	Why does <b>gnuplot</b> ignore my very small numbers? . . . . .	19
8.5	When I <i>replot</i> or <i>resize</i> a multiplot I get only a fragment . . . . .	19
8.6	My formulas (like 1/3) are giving me nonsense results! What's going on? . . . . .	19
8.7	My output files are incomplete! . . . . .	20
8.8	Calling gnuplot in a pipe or from a script doesn't show a plot on my screen! . . . . .	20
<b>9</b>	<b>Credits</b>	<b>20</b>

## 0 Meta – Questions

### 0.1 Where do I get this document?

The newest version of this document is on the web at <http://www.gnuplot.info/faq/>.

### 0.2 Where do I send comments about this document?

Send comments, suggestions etc to the developer mailing list <mailto://gnuplot-beta@lists.sourceforge.net>.

## 1 General Information

### 1.1 What is gnuplot ?

**gnuplot** is a command-driven plotting program. It can be used interactively to plot functions and data points in both two- and three-dimensional plots in many different styles and many different output formats. **Gnuplot** can also be used as a scripting language to automate generation of plots. It is designed primarily for the visual display of scientific data. **gnuplot** is copyrighted, but freely distributable; you don't have to pay for it. You are welcome to download the source code.

### 1.2 How did it come about and why is it called gnuplot ?

The authors of **gnuplot** are: Thomas Williams, Colin Kelley, Russell Lang, Dave Kotz, John Campbell, Gershon Elber, Alexander Woo and many others.

The following quote comes from Thomas Williams:

I was taking a differential equation class and Colin was taking Electromagnetics, we both thought it'd be helpful to visualize the mathematics behind them. We were both working as sys admin for an EE VLSI lab, so we had the graphics terminals and the time to do some coding. The posting was better received than we expected, and prompted us to add some, albeit lame, support for file data.

Any reference to GNUplot is incorrect. The real name of the program is "gnuplot". You see people use "Gnuplot" quite a bit because many of us have an aversion to starting a sentence with a lower case letter, even in the case of proper nouns and titles. gnuplot is not related to the GNU project or the FSF in any but the most peripheral sense. Our software was designed completely independently and the name "gnuplot" was actually a compromise. I wanted to call it "llamaplot" and Colin wanted to call it "nplot." We agreed that "newplot" was acceptable but, we then discovered that there was an absolutely ghastly pascal program of that name that the Computer Science Dept. occasionally used. I decided that "gnuplot" would make a nice pun and after a fashion Colin agreed.

### 1.3 What does gnuplot offer?

- Two-dimensional functions and data plots combining many different elements such as points, lines, error bars, filled shapes, labels, arrows, ...
- Polar axes, log-scaled axes, general nonlinear axis mapping, parametric coordinates
- Data representations such as heat maps, beeswarm plots, violin plots, histograms, ...

- Three-dimensional plots of data points, lines, and surfaces in many different styles (contour plot, mesh)
- Algebraic computation using integer, floating point, or complex arithmetic
- Data-driven model fitting using Marquardt-Levenberg minimization
- Support for a large number of operating systems, graphics file formats and output devices
- Extensive on-line help
- T<sub>E</sub>X-like text formatting for labels, titles, axes, data points
- Interactive command line editing and history

## 1.4 Is gnuplot suitable for scripting?

Yes. Gnuplot can read in files containing additional commands during an interactive session, or it can be run in batch mode by piping a pre-existing file or a stream of commands to stdin. Gnuplot is used as a back-end graphics driver by higher-level mathematical packages such as Octave and can easily be wrapped in a cgi script for use as a web-driven plot generator. Gnuplot supports context- or data-driven flow control and iteration using familiar statements *if else continue break while for*.

## 1.5 Can I run gnuplot on my computer?

**Gnuplot** is in widespread use on many platforms, including MS Windows, linux, unix, and macOS. The current source code retains supports for older systems as well, including VMS, Ultrix, OS/2, and MS-DOS. However 16-bit platforms are no longer supported.

You should be able to compile the **gnuplot** source more or less out of the box in any reasonably standard (ANSI/ISO C, POSIX) environment.

## 1.6 Legalities

**Gnuplot** is authored by a collection of volunteers, who cannot make any legal statement about the compliance or non-compliance of **gnuplot** or its uses. There is no warranty whatsoever. Use at your own risk.

**Gnuplot** is freeware in the sense that you don't have to pay for it. You can use or modify gnuplot as you like, however certain restrictions apply to further distribution of modified versions. Please read and abide by the modification and redistribution terms in the *Copyright* file. Some individual source files are explicitly dual-licensed; in those cases alternative terms for redistribution of code in that specific file are listed at the head of the file.

## 1.7 Does gnuplot have anything to do with the FSF and the GNU project?

**Gnuplot** is neither written nor maintained by the FSF. At one time it was distributed by the FSF but this is no longer true. **Gnuplot** as a whole is not covered by the GNU General Public License (GPL).

## 1.8 Where do I get further information?

See the main gnuplot web page <http://www.gnuplot.info>.

Some documentation and tutorials are available in other languages than English. See <http://gnuplot.sourceforge.net/help.html>, section "Localized learning pages about gnuplot", for the most up-to-date list.

## 2 Setting it up

### 2.1 What is the current version of gnuplot ?

The current stable version of **gnuplot** is 6.0, first released in December 2023. Incremental versions (patchlevel 1, 2, ...) are typically released every six months. The development version is currently **gnuplot** 6.1.

### 2.2 Where can I get gnuplot ?

The best place to start is <http://www.gnuplot.info>. From there you find various pointers to other sites, including the project development site on SourceForge <http://sourceforge.net/projects/gnuplot>.

The source distribution ("gnuplot-6.0.0.tar.gz" or a similar name) is available from the official distribution site <http://sourceforge.net/projects/gnuplot>.

### 2.3 Why would I care about the development version?

The current development version will generally include features that are not yet part of the most recent stable release of gnuplot.

### 2.4 Where can I get current development version of gnuplot ?

The development version of gnuplot is held in a git repository which you can clone as shown below to inspect or build an executable program from the source.

```
git clone https://git.code.sf.net/p/gnuplot/gnuplot-main gnuplot
```

Questions related to the development version should go to <mailto://gnuplot-beta@lists.sourceforge.net>.

### 2.5 How do I compile gnuplot on my system?

Read the release notes and files *README* and *INSTALL*. You will need C and C++ compilers and installed versions of various support libraries depending on what configuration options you choose and what terminal types you want your executable to support.

- To build from a release version on linux, use *./configure* (or *./configure --prefix=\$HOME/usr* for an installation for a single user), *make* and finally *make install*. Pay close attention to the output from the *configure* script (yes there is a lot of it). There you will find hints as to what additional support libraries may be needed and what additional options may be desirable. In general you will need to have installed the "development" package for each of the support libraries.
- To build from the development source on linux you must run a separate script *./prepare* prior to running *./configure*.
- On Windows, makefiles can be found in *config/mingw*, *config/msvc*, *config/watcom*, and *config/cygwin*. Update the options in the makefile's header and run the appropriate *make* tool in the same directory as the makefile. Additional instructions can be found in the makefiles.
- For other platforms, copy the relevant makefile (e.g. *makefile.os2* for OS/2) from *config/* to *src/*, optionally update options in the makefile's header, then change directory to *src* and run *make*.

## 2.6 What documentation is there, and how do I get it?

Full documentation is included in the release distribution as a PDF file. Individual sections can be browsed from inside a gnuplot session by typing *help* keyword. Documentation in other formats can be compiled from source in the *docs* subdirectory.

Online copies in English and Japanese are available at <http://gnuplot.sourceforge.net/documentation.html>.

## 2.7 Worked examples

There is a directory of worked examples in the the source distribution. Many of these examples and the resulting plots may also be found online at <http://gnuplot.sourceforge.net/demo/>.

## 2.8 How do I determine which options were compiled into my gnuplot executable?

Given that you have a compiled version of **gnuplot**, you can use the *show* command to display a list of configuration and build options that were used to build your copy. The output formats (a.k.a. "terminals") built into your copy of gnuplot are reported by *set terminal*.

```
gnuplot> show version long
gnuplot> set terminal
```

# 3 Working with it.

## 3.1 How do I get help?

Give the *help* command at the initial prompt. After that, keep looking through the keywords. Good starting points are *help plot* and *help set*.

- Read the manual, if you have it.
- Run the demos in the *demo* subdirectory or look at the online copies. They should give you some ideas. <http://gnuplot.info/demo>
- Ask your colleagues, the system administrator, or the person who set up **gnuplot** .
- There is an old-school usenet group devoted to gnuplot questions <news://comp.graphics.apps.gnuplot> inhabited mostly by users who found it last century.
- A more active help forum may be found on StackOverflow <http://stackoverflow.com/questions/tagged/gnuplot>
- If all these fail, send mail to the gatewayed mailing list <mailto://gnuplot-info@lists.sourceforge.net>. Please note that to reduce the amount of spam it would otherwise receive, you must subscribe before you can post to it. Subscription instructions may be found at <http://lists.sourceforge.net/lists/listinfo/gnuplot-info>.

When asking a question, always mention the **gnuplot** version and operating system you are using. If you are asking about a plot that did not come out the way you expected, please try to show a minimal set gnuplot commands that produce a plot showing the problem.

### 3.2 How do I print out my graphs?

The output format produced by a plot command is determined by a prior *set terminal* command. For non-interactive output you should pair this with a *set output* command to provide a file name.

As an example, the following session first plots a graph of  $\sin(x)$  to the screen and then redraws that same plot as a PDF output file. Note: the PDF plot may not look exactly like the plot on the screen.

```
gnuplot> plot sin(x)
gnuplot> set terminal pdf
Terminal type is now 'pdfcairo'
Options are ' transparent enhanced fontscale 0.5 size 5.00in, 3.00in '
gnuplot> set output "sin.pdf"
gnuplot> replot
gnuplot> unset output          # close output file (otherwise it stays open)
gnuplot> unset terminal        # return to default interactive terminal
gnuplot>
```

If the start point is not the default interactive terminal you can accomplish the same thing using *push* and *pop*

```
gnuplot> set terminal push      # save current terminal type (may not be default)
gnuplot> set terminal pdf
gnuplot> set out 'a.pdf'
gnuplot> replot
gnuplot> unset out
gnuplot> set term pop          # restore saved terminal type
```

Some interactive terminal types (*win*, *wxt*, *qt*) provide a printer icon on their toolbar. This widget prints the current plot or saves it to file using generic system tools rather than by using a different gnuplot terminal type. That is, the file you get by selecting "save to png" in the print menu may be different than the file you get from *set term png; replot;*. In general a plot saved in this way will more closely reproduce the screen image than a plot generated from the command line by changing the terminal type.

### 3.3 How do I include my graphs in my favorite word processor?

Basically, you save your plot to a file in a format your word processor can understand, and then you read in the file from your word processor. Vector formats (PostScript, emf, svg, pdf,  $\text{\TeX}$ ,  $\text{\LaTeX}$ ) should be preferred, as you can scale your graph later to the right size.

Use *set term* to get a list of available file formats.

Many word processors can use Encapsulated PostScript (\*.eps) for graphs. You can generate eps output in **gnuplot** using either *set terminal postscript eps* or *set terminal epscairo*. **Gnuplot** does not embed a bitmap preview image in the output eps file. To accommodate some word processors you may have to add a preview image yourself using an external tool before importing it into the word processor.

Some applications, including the LibreOffice and Microsoft Office suites, can handle vector images in EMF format. These can be either produced by the emf terminal, or by selecting 'Save as EMF...' from the toolbar of the windows terminal plot window.

LibreOffice can also read SVG, as well as AutoCAD's dxf format.

There are many ways to use gnuplot to produce graphs for inclusion in a  $\text{\TeX}$  or  $\text{\LaTeX}$  document. Some terminals produce \*.tex fragments for direct inclusion; others produce \*.eps, \*.pdf, \*.png output to be included using the \includegraphics command. The epslatex and cairolatex terminals produce both a graphics file (\*.eps or \*.pdf) and



a \*.tex document file that refers to it. The tikz terminal produces full text and graphics to a pdf file when the output is processed with pdflatex.

Most word processors can import bitmap images (png, pbm, etc). The disadvantage of this approach is that the resolution of your plot is limited by the size of the plot at the time it is generated by **gnuplot**, which is generally a much lower resolution than the document will eventually be printed in.

### 3.4 How do I edit or post-process a gnuplot graph?

This depends on the terminal type you use.

- **svg** terminal (scalable vector graphics) output can be further edited by a svg editor, e.g. **Inkscape** (<http://www.inkscape.org>), **Skencil** (<http://www.skencil.org>) or **Dia** (<http://projects.gnome.org/dia/>), or loaded into **OpenOffice.org** with an on-fly conversion into OO.o Draw primitives.
- PostScript or PDF output can be edited directly by tools such as Adobe Illustrator or Acrobat, or can be converted to a variety of other editable vector formats by the **pstoedit** package. Pstoedit is available at <http://www.pstoedit.net>.
- The DXF format is the AutoCAD's format, editable by several other applications.
- Bitmapped graphics (e.g. png, jpeg, pbm) can be edited using tools such as ImageMagick or Gimp. In general, you should use a vector graphics program to post-process vector graphic formats, and a pixel-based editing program to post-process pixel graphics.

### 3.5 How do I save and restore my current settings?

Use the *save "filename"* and *load "filename"* commands.

### 3.6 Can I put both commands and data into a single file?

Starting with version 5, **Gnuplot** supports named blocks of data in "here document" format:

```
gnuplot> $DATABLOCK << EOD
  cats 4 2
  dogs 1 4
EOD
gnuplot> plot $DATABLOCK using 2:3:1 with labels
```

Once the named block has been defined, it can be used as many times as you like.

Data can also be provided in-line as part of a plot command using the pseudo-file "-". In this case the data can only be used once.

```
gnuplot> plot "-"
1 1
2 4
3 9
e
```

### 3.7 How do I run my data through a filter before plotting?

If your operating system supports the `popen()` function, you can filter input data through another program or system utility as part of the *plot* command.

```
gnuplot> plot "< sort +2 file.in" # pre-sort data on column 2
```

This mechanism is particularly powerful in combination with the unix-derived command line utilities *awk*, *sort* and *grep*.

### 3.8 Can I use gnuplot routines for my own programs?

On systems supporting pipes, you can pipe commands to **gnuplot** from other programs. Many applications with gnuplot as the graphics engine, like Octave (<http://www.octave.org>), use this method. This also works from a cgi script to drive **gnuplot** from a forms-based web page.

## 4 Customizing the appearance of your plot

### 4.1 How to inspect or change the default colors, line, and point properties?

When you issue a *plot* or *splot* command with multiple components, **gnuplot** will by default cycle through a set of colors and linetypes. You can override this by providing specific color or linetype properties in the plot command or you can change the default sequence. Each of the commands below accepts many additional parameters

*test* displays the active colors, line and point properties, etc for the current terminal type

*set color* or *set monochrome* selects a pre-defined sequence.

*set linetype* changes the properties of an existing linetype or adds a new one.

*set palette* changes the color palette used for pm3d modes such as heat maps.

*set pointsize* scales all points by an additional factor

### 4.2 Hidden line or surface removal

There are two relevant commands. *set hidden3d* affects surfaces drawn using the 3D plot style *splot ... with lines*. It also clips line segments created by other 3D plot styles that are occluded by those surfaces. However it does not handle plots generated in *pm3d* mode. This includes styles *with pm3d*, *with zerror*, *with boxes*, and miscellaneous plot elements generated while *set pm3d* is in effect. Hidden surface removal for these plots is achieved instead by drawing them in order of their distance from the viewer, enabled by the command *set pm3d depthorder*.

### 4.3 How do I force exact positions for the graph borders on the page?

Specify the position of the top, bottom, left, and right borders in terms of their fractional position within the page:

```
set lmargin at screen 0.05
set bmargin at screen 0.05
set rmargin at screen 0.95
set tmargin at screen 0.95
```

## 4.4 Arranging multiple plots next to each other on a single page

The command you want is *set multiplot*. The program can place a specified number of plots on a regular grid (*set multiplot layout <rows>, <columns> ...*) or you can position them one by one using *set origin* and *set size*.

## 4.5 How to request 1:1 scaling of axes?

Try *set size square* or *set view equal xy*. In version 6 there is also a command *set isotropic*.

## 4.6 Palette that works for both color and black&white printing?

Try *set palette cubehelix*.

## 4.7 How do I skip data points?

By specifying *?* as a data value, as in

```
1 2
2 3
3 ?
4 5
```

See also *set missing*. See also *set datafile commentschars* for specifying comment characters in data files.

## 4.8 How do I plot every nth point?

This can be specified with various options for the command *plot*, for example *plot 'a.dat' every 2*. If you want to draw a line through every point but only draw a point symbol at every nth point, then try *plot 'a.dat' with linespoints pointinterval n*.

## 4.9 How do I plot a vertical line?

Depending on context, the main methods are:

- *set arrow .... nohead* where you have to compute explicitly the start and the end of the arrow.
- generate inlined datapoints and plot them

## 4.10 Y axis label in the wrong place and/or left margin too big

**Gnuplot** has trouble estimating how much space to the left of the plot will be required to hold the Y axis label if it is printed horizontally (*set ylabel norotate*). This is particularly true if there is TeX markup in the label string. To work around this problem you can place the text in a numbered label rather than in *ylabel*.

```
Y = 1001
set label Y '$\operatorname{\mathfrak{Im}} S_{21}$'
set label Y norotate at graph 0.0, 0.5 offset -6
```

## 5 Plot types that people often ask about

### 5.1 Animations

**Gnuplot** versions through release 5.4 support only one terminal type (gif) that directly outputs an animated file:

```
set terminal gif animate {delay <time>} {loop <N>} {optimize}
```

Version 6 also supports animation in webp format. Have a look at <http://gnuplot.sourceforge.net/demo/animate2.html> in the demo collection.

### 5.2 Implicit defined graphs

Implicit graphs or curves cannot be plotted directly in **gnuplot**. However there is a workaround.

```
gnuplot> # An example. Place your definition in the following line:
gnuplot> f(x,y) = y - x**2 / tan(y)
gnuplot> set contour base
gnuplot> set cntrparam levels discrete 0.0
gnuplot> unset surface
gnuplot> set table $TEMP
gnuplot> splot f(x,y)
gnuplot> unset table
gnuplot> plot $TEMP w l
```

The trick is to draw the single contour line  $z=0$  of the surface  $z=f(x,y)$ , and store the resulting contour curve to a temporary file or datablock.

### 5.3 How to fill an area between two functions

A plot with filled area between two functions  $f(x)$  and  $g(x)$  can be obtained using the pseudo file '+' with plot style *filledcurves*.

```
f(x)=cos(x); g(x)=sin(x)
set xrange [0:pi]
plot '+' using 1:(f($1):(g($1)) with filledcurves
```

Note that this code fragment fills area between the two curves regardless of which one is above the other. If you want to fill only the area satisfying  $g(x)<f(x)$  or  $f(x)<g(x)$  add an additional keyword *above* or *below* after *filledcurves*.

### 5.4 Drawing a 2D projection of 3D data

The command *set view map* adjusts the view angle and scaling such that subsequent 3D graphs made with *splot* have approximately the same layout as a 2D graph made with *plot*, with the x axis horizontal and the y axis vertical. Version 5.4 commands *set view projection xz* and *set view projection yz* similarly initialize layouts for 2D projection of the xz or yz plane, with the z axis horizontal and the x or y axis vertical.

## 5.5 How to overlay dots/points scatter plot onto a pm3d map/surface

Use the *explicit* option of the pm3d style:

```
gnuplot> set pm3d explicit
gnuplot> splot x with pm3d, x*y with points
```

## 5.6 How to produce labeled contours

Labeling individual contours in a contour plot used to require special tricks and extra processing steps in **gnuplot** version 4. Since version 5 it is much simpler. Plot the contours twice, once "with lines" and once "with labels". To make the labels stand out it may help to use

```
set style textbox opaque noborder
set contours
splot 'DATA' with lines, 'DATA' with labels boxed
```

## 5.7 Does gnuplot support bar-charts/histograms/boxes?

**Gnuplot** supports various clustered and stacked histogram styles to display pre-tabulated data. It also offers a few options for accumulating raw data into bins, which can in turn be displayed as a bar chart. See the documentation for *bins* and for *smooth frequency*.

## 5.8 Does gnuplot support pie charts? quarterly time charts?

Pie charts were difficult in older **gnuplot** versions but see <http://gnuplot.sourceforge.net/demo/circles.html>, or have a look at <http://gnuplot-tricks.blogspot.com/2009/08/pie-charts-entirely-in-gnuplot.html>. Version 6 introduced a new plot style *with sectors* that makes pie charts much easier.

The demo collection contains an example of a simple Gantt chart.

## 5.9 Does gnuplot support multiple y-axes on a single plot?

Yes. 2D plots can have separate x axes at the bottom (x1) and top (x2), and separate y axes at the left (y1) and right (y2). Version 5 offers a plot mode *with parallelaxes* that allows any number of additional y axes to be defined.

## 5.10 How to draw a solid made of triangular facets?

Plot style *with polygons* can handle polygonal faces (triangles, rectangles, octagons, ...) in either 2D or 3D plots. This style was introduced in version 5.4.

In older versions the best you can do is to describe colored quadrangles that are facets of a 3D object using a file organized like this:

```
# triangle 1
x0 y0 z0 <c0>
x1 y1 z1 <c1>

x2 y2 z2 <c2>
x2 y2 z2 <c2>
```

```
# triangle 2
x y z
...
```

Notice the single and double blank lines. Also notice that each triangle really has four vertices of which two are identical. This is because the pm3d code only knows how to deal with quadrangles. `<cN>` is an optional color.

Then plot it:

```
set pm3d
set style data pm3d
set pm3d depthorder
splot 'facets.dat' using 1:2:3 title "default coloring"
splot 'facets_with_color.dat' using 1:2:3:4 title "explicit colors"
```

Gnuplot is not a 3D modeling program. For true 3D rendering you would be probably be better off using a ray-tracing program.

## 5.11 How do I plot two functions in non-overlapping regions?

Give the desired range immediately before each function being plotted. For example, to plot experimental data and two different functional models `f1` and `f2` covering two different portions of the domain:

```
gnuplot> set autoscale x # get x range from the data
gnuplot> plot 'data', [*:0] f1(x), [0:*] f2(x)
```

## 5.12 How do I plot lines (not grids) using splot?

If the data input to `splot` is arranged such that each line contains the same number of data points (using blank lines as delimiters, as usual), `splot` will by default treat the data as describing a surface. If you want to draw individual lines instead, try some combination of *unset surface*, *set surface explicit*, *plot ... nosurface*.

## 5.13 How do I plot a function $f(x,y)$ that is bounded by other functions in the x-y plane?

Here is one way:

```
gnuplot> f(x,y) = x**2 + y **2
gnuplot> x(u) = 3*u
gnuplot> yu(x) = x**2
gnuplot> yl(x) = -x**2
gnuplot> set parametric
gnuplot> set cont
gnuplot> splot [0:1] [0:1] u,y1(x(u))+(yu(x(u)) - yl(x(u)))*v,\
>                                     f(x(u), (yu(x(u)) - yl(x(u)))*v)
```

# 6 Text formatting and special symbols

## 6.1 Text markup using "enhanced text" mode

Starting with version 5, **gnuplot** defaults to "enhanced text" mode, in which text markup is indicated by a set of special characters embedded in the text.

Enhanced Text Control Codes			
Control	Example	Result	Explanation
$\wedge$	a $\wedge$ x	$a^x$	superscript
$\_$	a $\_$ x	$a_x$	subscript
@	a@ $\wedge$ b_ $\{$ cd $\}$	$a_{cd}^b$	phantom box (occupies no width)
&	d& $\{$ space $\}$ b	d $\_ \_ \_ \_ \_ $ b	inserts space of specified length
~	~a $\{$ .8- $\}$	$\tilde{a}$	overprints '-' on 'a', raised by .8 times the current fontsize
	$\{$ /Times abc $\}$	abc	print abc in font Times at current size
	$\{$ /Times*2 abc $\}$	<b>abc</b>	print abc in font Times at twice current size
	$\{$ /Times:Italic abc $\}$	<i>abc</i>	print abc in font Times with style italic
	$\{$ /Arial:Bold=20 abc $\}$	<b>abc</b>	print abc in boldface Arial font size 20

## 6.2 How do I turn text markup on or off?

To exempt a particular text string from this processing mode use the keyword *noenhanced*. For example we don't want to interpret file names as subscripts:

```
set title 'Compare file_1.dat and file_2.dat' noenhanced
```

## 6.3 Is UTF-8 the answer to all my special character problems?

Yes.

Unfortunately there are some circumstances where it very cumbersome to use, notably in generating PostScript output. If you are working in a UTF-8 computing environment, you probably do not have to do anything special in gnuplot to use it. If you are not, then you can still tell gnuplot to use UTF-8 for output: *set encoding utf8*.

If you cannot enter UTF-8 characters on your keyboard you will have to solve that outside of gnuplot. Or you can use octal escape sequences to type them in byte-by-byte, or (since version 5.4) unicode escape sequences like  $\backslash$ U+221E ( $\infty$ ). If your keyboard does generate UTF-8 but you don't know what keystrokes would produce a specific character, there are probably suitable character selection apps for your desktop (e.g. KDE *kchselect* or GNOME *Character Map*).

## 6.4 What if I need h-bar (Planck's constant)?

The most straightforward way is to use a UTF-8 font, and type in the  $\hbar$  character (Unicode  $\backslash$ U+210F) directly.

PostScript: PostScript does not handle utf8 easily so you must use an approximation based on enhanced text markup and possibly a special Symbol font:

$\@{\/=56 -}\ {/\=24 h}$  or  $\{/\=8 \@{\textit{Symbol}=24 -}\ \_ {\/=14 h}\}$  In the latter, the "-" (a long one in /Symbol) is non-spacing and 24-pt. The 14-pt "h" is offset by an 8-pt space (which is the space preceding the "\_") but smaller, since it's written as a subscript. But these don't look too much like the hbar we're used to, since the bar is horizontal instead of sloped. Another possibility is  $\{/\=14 @^{\textit{Symbol}=10 -}\ {/\=14 h}\}$ .

The reduced Planck's constant can be set very easily by using the AMS-LaTeX PostScript fonts which are available from <http://www.ams.org/tex/amsfonts.html> (also included in many LaTeX distributions). **Gnuplot** (see `help fontpath`) and the PostScript interpreter (usually Ghostscript) have to know where the file `msbm10.pfb` (or `msbm10.pfa`) resides. Use  $\{/\textit{MSBM10} \backslash 175\}$  to produce  $\backslash$ hslash which is a "h" superimposed by a sloped bar. The standard  $\backslash$ hbar (horizontal bar) has the octal code 176. Please note that h-bar exists only as an italic type.

## 6.5 What if I need the Solar mass symbol?

As with Planck's constant, the most straightforward way is to use a UTF-8 font, and type in the ☉ character (Unicode \U+2299 ; "circled dot operator") directly. The very similar glyph at (Unicode code point \U+2609 ; "sun") may be even better, but not many fonts provide it.

## 6.6 How to use Greek letters or other special symbols?

The old-style way is to use enhanced text mode to switch to a specialize font, e.g. the Adobe "Symbol" font, that maps the characters you want onto ascii letters 'a', 'b', etc. This may still be necessary for PostScript output. However a much simpler way is to select UTF-8 encoding and enter the special characters just as you would any other text. This obviates the need to change fonts and gives you access to all unicode code points including CJK character sets. To actually print or view the files produced by **gnuplot** you still need appropriate fonts installed on your computer or output device. **Gnuplot** itself does not provide fonts.

The various L<sup>A</sup>T<sub>E</sub>X terminal types (*latex*, *epslatex*, *tikz*, *context*, *cairolatex*) hand off text generation to L<sup>A</sup>T<sub>E</sub>X. In this case you can use normal L<sup>A</sup>T<sub>E</sub>X markup like  $\alpha_3$  ( $\alpha_3$ ).

## 6.7 How do I include accented characters?

If you are stuck using a non-utf8 encoding you should use 8-bit character codes together with the appropriate encoding option to obtain accented characters like ü or ñ in your labels. You can represent the 8-bit code with an escape sequence. See the following example:

```
gnuplot> set encoding iso_8859_1
gnuplot> set title "M\374nchner Bierverbrauch \374ber die Jahre"
gnuplot> plot "bier.dat" u 1:2
```

Everyone else should use UTF-8, where these are normal characters.

## 6.8 Can I put different text sizes into my plots?

Most terminal types allow you to specify a starting font face and size. The "enhanced text" mode allows you to change fonts, text sizes, bold and italic styles within a plot.

# 7 Miscellaneous

## 7.1 Does gnuplot support a driver for graphics format XXX?

To see a list of the available graphic drivers for your installation of **gnuplot**, type *set term*.

Some graphics drivers are included in the source distribution but are not built by default. If you want to use them, you'll have to recompile from source.

## 7.2 I've found a bug, what do I do?

First, try to see whether it actually is a bug, or whether it is a feature you can turn off with some obscure *set* command.



Next, see whether you have an old version of **gnuplot** ; if so, chances are the bug has been fixed in a newer release. Before submitting a bug report, please check whether the bug in question has already been fixed in the upstream source since the release of the current version. These are marked "pending-fixed" on the bug tracker.

If, after checking these things, you still are convinced that there is a bug please report it using the bug-tracker at <http://sourceforge.net/p/gnuplot/bugs>. Be sure to include the version of **gnuplot** (including patchlevel) and the operating system you are running it on. It helps a lot if you can provide a simple script that reproduces the error.

The tracker on sourceforge is for reporting bugs and collecting bug fixes that will appear in a subsequent release. The various online forums are better places to ask about work arounds or actually solving **gnuplot** related problems.

### 7.3 Can I do heavy-duty data processing with gnuplot ? or What is beyond gnuplot ?

**gnuplot** by itself is not very well suited for heavy numerical computation. On the other hand it can easily handle very large data sets (millions of points). If you have a specific application where the limitation is gnuplot's speed in evaluating a non-trivial function, it might be worth coding this function in C or C++ and making it a loadable plugin that gnuplot can call.

Programs you might look into if you need intensive numerical computation:

**octave** (<http://www.octave.org>) is a high-level language primarily intended for numerical computations. Octave is licensed under GPL, and in principle, it is a free Matlab clone. It provides a convenient command line interface for solving linear and nonlinear problems numerically. By the way, octave uses **gnuplot** as its plotting engine, so you get a data-processing program on top of **gnuplot** .

**scilab** (<http://www.scilab.org>) is another open source alternative to **matlab**.

**julia + Gaston** (<http://github.com/mbaz/Gaston.jl>) The Julia language is designed for numerical analysis and computational science. Gaston is a julia package that provides an interface to gnuplot for graphical output.

### 7.4 What if I need special function that gnuplot doesn't have?

**Gnuplot** version 6 provides a greatly expanded set of complex special functions. If the one you need is not included, you may have to use another program or write a custom plugin for gnuplot.

### 7.5 How to use hotkeys in my interactive terminals

Most of the interactive terminals support both pre-defined and user-defined hotkeys to replot, toggle plot elements, change axis scaling, and so on. Hit *h* in an active gnuplot plot window to get list of hotkeys. Read *help mouse* and *help bind* for more information.

### 7.6 I want to help in developing the next version of gnuplot . What can I do?

Join the **gnuplot** beta test mailing list by sending a mail containing the line `subscribe gnuplot-beta` in the body (not the subject) of the mail to <mailto:Majordomo@lists.sourceforge.net>.

## 8 Common problems

### 8.1 Help! None of my fonts work.

Gnuplot does not do font handling by itself; it must necessarily leave that to the individual device support libraries. Unfortunately, this means that different terminal types need different help in finding fonts. Here are some quick hints. For more detailed information please see the gnuplot documentation for the specific terminal type you are having problems with.

**png/jpeg/gif** These terminal types use the libgd support library, which searches for fonts in the directories given in the environmental variable GDFONTPATH. Once you get libgd fontpaths sorted out, you will probably want to set a default font for gnuplot. For example: `setenv GNUPLOT_DEFAULT_GDFONT verdana`

**post** PostScript font names are not resolved until the document is printed. Gnuplot does not know what fonts are available to your printer, so it will accept any font name you give it. However, it is possible to bundle a font with the gnuplot output; please see the instructions given by gnuplot's internal command "help set term post fontfile".

**svg** Font handling is viewer-dependent.

**x11** The x11 terminal uses the normal x11 font server mechanism. The only tricky bit is that in order to use multi-byte fonts you must explicitly say so:

```
set term x11 font "mbfont:sazanami mincho,vera,20"
```

**win** Select "Choose font..." from the "Options" pull-down menu in the toolbar.

**wxt, qt** On linux systems these terminals rely on fonts provided by the system's *fontconfig* utility.

### 8.2 The first plot in a qt terminal session fails or has bad layout

You may also get an error message about "slow font initialization". This is because qt relies on a shared system font cache. If you request a font that no one else is using, it takes a while to update the cache. This mostly happens on Windows or macOS, because other programs on those systems tend to use a different font mechanism so the relevant font cache may be empty. Try invoking gnuplot with command line option *--slow*.

### 8.3 Pm3d plot from a datafile does not draw anything

You do `set pm3d; splot 'a.dat'` and no plot but colorbox appears. Perhaps there is no blank line in between two subsequent scans (isolines) in the data file? Add blank lines! If you are curious what this means, then don't hesitate to look to files like *demo/glass.dat* or *demo/triangle.dat* in the gnuplot demo directory.

You can find useful the following awk script (call it e.g. *addblanks.awk*) which adds blank lines to a data file whenever number in the first column changes:

```
/^[[[:blank:]]*#]/ {next} # ignore comments (lines starting with #)
NF < 3 {next} # ignore lines which don't have at least 3 columns
$1 != prev {printf "\n"; prev=$1} # print blank line
{print} # print the line
```

Then, either preprocess your data file by command `awk -f addblanks.awk <a.dat` or plot the datafile under a unixish platform by

```
gnuplot > splot "<awk -f addblanks.awk a.dat".
```

## 8.4 Why does gnuplot ignore my very small numbers?

For some purposes **Gnuplot** treats numbers less than 1.e-08 as being zero. Thus if you are trying to plot a collection of very small numbers, they may be plotted as zero. Worse, if you're plotting on a log scale, they will be off scale. Or, if the whole set of numbers is "zero", your range may be considered empty:

```
gnuplot> plot 'test1'
Warning: empty y range [4.047e-19:3e-11], adjusting to [-1:1]
gnuplot> set yrange [4e-19 : 3e-11]
gnuplot> plot 'test1'
^
y range is less than `zero`
```

The solution is to change gnuplot's idea of "zero":

```
gnuplot> set zero 1e-20
```

For more information, type *help set zero*.

## 8.5 When I *replot* or *resize* a multiplot I get only a fragment

**Gnuplot** versions through 5.4 retained only enough information to regenerate the most recent *plot* or *splot* command. In order to reproduce the entire multiplot you needed to save the complete sequence of commands that generated it into a script file. Then you could *load* that script into **gnuplot** as many times as needed for replotting the drawing to different terminals or output files.

The same limitation arose when you resized a multiplot displayed in an interactive terminal window, because the resize event normally triggers a replot. The *qt* and *wxt* terminals provide a toggle in the toolbox widget that suppresses this replot on resize. The *x11* terminal provides a terminal option *set term x11 noreplotonresize*.

**Gnuplot** version 6 automates regeneration of all components in a multiplot. Starting when a command *set multiplot* is seen, the program saves commands into a data block named `$GPVAL_LAST_MULTIPLOT`. Once the closing command *unset multiplot* is issued, this data block can be replayed to redraw the corresponding set of plots that make up the multiplot. This can be done explicitly using the *remultiplot* command or implicitly from *replot* commands or resizing events if the immediately preceding plot command was part of a multiplot. It is either a feature or a drawback that the result of replaying the original commands may produce a slightly different result if the state of the program (axis ranges, linetypes, fonts, etc) has changed in the interim.

## 8.6 My formulas (like 1/3) are giving me nonsense results! What's going on?

**Gnuplot** does integer, and not floating point, arithmetic on integer expressions. For example, the expression 1/3 evaluates to zero. If you want floating point expressions, supply trailing dots for your floating point numbers. Example:

```
gnuplot> print 1/3
0
gnuplot> print 1./3.
0.333333
```

This way of evaluating integer expressions is shared by both C and Fortran.

## 8.7 My output files are incomplete!

You may need to flush the output and close the file with *set output* or *unset output*).

Some output formats (postscript, pdf, latex, ...) can include several pages of plots in a single output file. For these output modes, gnuplot leaves the file open after each plot so that you can add additional plots to it. The file is not completed and made available to external applications until you issue a *set* or *unset output* command, select a different terminal type (*set term*), or exit gnuplot.

## 8.8 Calling gnuplot in a pipe or from a script doesn't show a plot on my screen!

One common cause is that **gnuplot** exits immediately after drawing, the plot window closes when gnuplot exits, and it all happens too fast for you see the plot. There are several solutions to this:

- Use the gnuplot command line option *-persist*. This leaves the plot window open after gnuplot itself exits.
- Interactive terminals also accept *persist* as an option to *set term*.
- Send a *pause mouse close* command to gnuplot before closing the pipe. This will tell gnuplot not to exit until the plot window is closed by some other action.
- Have the script close the pipe only when it is sure you don't need gnuplot and its plot window any more.

Here is a short Perl-script showing two of these fixes:

```
#!/usr/local/bin/perl -w
open (GP, "|/usr/local/bin/gnuplot -persist") or die "no gnuplot";
# force buffer to flush after each write
use FileHandle;
GP->autoflush(1);
print GP,"set term x11; plot sin(x) with lines\n";
print GP,"pause mouse close\n";
close GP
```

## 9 Credits

This FAQ was initially compiled by John Fletcher with contributions from Russell Lang, John Campbell, David Kotz, Rob Cunningham, Daniel Lewart and Alex Woo. Reworked by Thomas Koenig from a draft by Alex Woo, with corrections and additions from Alex Woo, John Campbell, Russell Lang, David Kotz and many corrections from Daniel Lewart. Again reworked for **gnuplot** 3.7 by Alexander Mai and Juergen v.Hagen with corrections by Lars Hecking, Hans-Bernhard Broecker and others. Revised for **gnuplot** version 4 by Petr Mikulík and Ethan Merritt. Revised for **gnuplot** versions 5 and 6 by Ethan Merritt.