gnuplot Plotting from the shell

START HERE

`gnuplot`
Start interactive prompt

`plot sin(x)`
Plot a function

`plot '<file>' using 1:2 with lines`
Plot two columns

`set xlabel '<text>'`
Label an axis

`set terminal png`
Render to an image

`set output '<file>'`
Write to a file

`replot`
Redraw with changes

`gnuplot <script.gp>`
Run a script

Data / function file or expression → **`set ...`** → **Settings** terminal, labels, ranges → **`plot`** → **Plot** using, with, title → **`set output`** → **Output** window, png, svg, pdf

CONCEPTS

Settings stick until changed: set options stay active for later plots. Use unset <option> or reset to clear them.

Always set terminal first, then output, then plot. Call unset output at the end so the file is flushed and closed.

RUNNING

`gnuplot`
Start interactive prompt

`gnuplot <script.gp>`
Run a script file

`gnuplot -persist -e "plot sin(x)"`
Plot and keep the window

`gnuplot -e "<cmds>" <script.gp>`
Run a command, then a script
Commands and files run in the order given; use -e to pass variables, e.g. -e "file='a.dat'".

`gnuplot -c <script.gp> <args>`
Script with arguments
Inside the script the arguments are ARG1, ARG2, ... and ARGC.

`gnuplot -`
Read commands from stdin

`load '<script.gp>'`
Run a script from the prompt

RUNNING (cont.)

`help <topic>`
Built-in help

`quit`
Leave gnuplot

`pause mouse close`
Keep window open until closed
Useful in scripts so the plot window does not vanish.

`show <setting>`
Show the current setting

PLOTTING

`plot <expr>`
Plot a function of x

`plot '<file>'`
Plot a data file (column 1 vs 2)

`plot '<file>' using 1:3`
Pick columns for x and y

`plot '<file>' using 1:($2*2)`
Compute with column values
Parentheses make an expression; $2 is column 2.

`plot '<f1>', '<f2>'`
Several datasets in one plot

`plot <a>, <b> title '<name>'`
Name a curve in the legend

`plot ... notitle`
Leave a curve out of the legend

PLOTTING (cont.)

`plot [<x0>:<x1>] [<y0>:<y1>] <expr>`
Set ranges for this plot

`plot '<file>' skip 1`
Skip header lines

`plot '<file>' using 1:2:3 with yerrorbars`
Error bars

`plot '<file>' using 1:xtic(2)`
Use column 2 as tick labels

`plot '<file>' using 1:2 smooth csplines`
Smooth the curve

`plot '<file>' using 1:2 axes x1y2`
Plot on the second y axis

`plot '< <cmd>'`
Plot output of a shell command
The text is passed to the shell; the output is read as data.

`replot`
Redraw the last plot

STYLES

`with lines`
Join points with lines

`with points`
Points only

`with linespoints`
Lines and points

`with boxes`
Bars

STYLES (cont.)

`linewidth <n>`
Line thickness (lw)

`linecolor rgb '<color>'`
Colour by name or #hex (lc)

`set style data lines`
Default style for data

`unset key`
Hide the legend

`set grid`
Show grid lines

AXES AND LABELS

`set title '<text>'`
Plot title

`set xlabel '<text>'`
X axis label

`set ylabel '<text>'`
Y axis label

`set xrange [<min>:<max>]`
Fix the x range

`set yrange [<min>:*]`
Fix min, auto max

`set logscale y`
Log scale on y

`set xdata time`
Treat x as dates

`set timefmt '%Y-%m-%d'`
Date format of input

`set datafile separator ','`
Read CSV

OUTPUT

`set terminal png size 800,600`
Raster image

`set terminal pngcairo size 800,600`
Image with cairo (nicer text)

`set terminal svg size 800,600`
Vector SVG

`set terminal pdfcairo size 6in,4in`
PDF output

`set output '<file>'`
Write output to a file

`unset output`
Close the output file

FIT AND STATS

`f(x) = a*x + b`
Define a function

`a = 1; b = 0`
Set initial parameter values
Poor start values are the main reason a fit fails to converge.

`fit f(x) '<file>' using 1:2 via a,b`
Least-squares fit

`plot '<file>', f(x)`
Show data and fit together

`stats '<file>' using 2`
Mean, min, max, sigma
Results are stored in variables such as STATS_mean and STATS_max.

MULTIPLOT AND SCRIPTING

`set multiplot layout 2,2`
Grid of plots

`unset multiplot`
Finish the multiplot

`do for [i=1:3] { <cmds> }`
Loop over commands

`reset`
Restore default settings

RECIPES

PNG from a CSV, one command

```
gnuplot -e "set datafile
separator ','; set term png
size 800,500; set output
'out.png'; plot 'data.csv'
using 1:2 with lines title
'value'"
```

Reusable script with arguments

```
# plot.gp: gnuplot -c plot.gp
data.csv out.svg
set terminal svg size 800,500
set output ARG2
set grid
plot ARG1 using 1:2 with lines
notitle
```

Fit a line and show the result

```
f(x) = a*x + b
fit f(x) 'data.dat' using 1:2
via a,b
set key top left
plot 'data.dat' title 'data',
f(x) title sprintf('%.2fx+
%.2f', a, b)
```