

Computational Astrophysics I: Introduction and basic concepts

Helge Todt

Astrophysics
Institute of Physics and Astronomy
University of Potsdam

SoSe 2025, 3.4.2025



Brief introduction to gnuplot

Starting gnuplot

- gnuplot is available for almost every platform (operating system): Linux, MacOS X, Windows, ...
- download, e.g. from <http://gnuplot.info/>
- under Linux: start interactive session in terminal via `gnuplot`



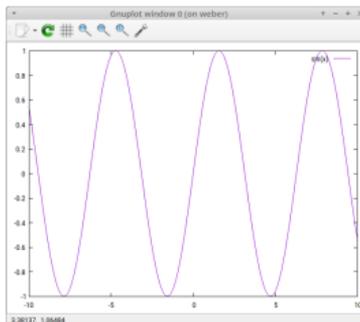
```
File Edit View Terminal Tabs Help
weber/htodt> gnuplot

G N U P L O T
Version 5.2 patchlevel 2   last modified 2017-11-15

Copyright (C) 1986-1993, 1998, 2004, 2007-2017
Thomas Williams, Colin Kelley and many others

gnuplot home:   http://www.gnuplot.info
faq, bugs, etc: type "help FAQ"
immediate help: type "help" (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> plot sin(x)
```



- quit gnuplot by command `exit`

- gnuplot can plot basic functions (independent variable / dummy variable is x) and combinations of them, default plot symbol for functions: solid line
- examples
 - `plot sin(x)`
 - `plot x**3 + 0.5*sqrt(2)`
- plotting more than one function by using comma separated list:
`plot sin(x), cos(1/x), tanh(x+2)`
- plotting only over a specific x-range (also for fitting):
`plot [-2.5:+3] cos(x)`
- plotting only over a specific y-range:
`plot [] [-1:+1] sin(x)`
- plotting only over a specific x- and y-range:
`plot [-2.5:+3] [-1:1] cos(2*x)`

- x- and y-axis labels:

```
set xlabel "d in pc"
```

```
set ylabel "t in Ga"
```

- key (legend): is automatically generated, can be written by option `title`:

```
plot "data.txt" title "observation (1998)" \  
  , f(x) t "model 17-04"
```

→ requires execution of previous plot command or just type `replot`

gnuplot plots data from files in ascii table format, i.e.

```
# this is a comment
4.5 91 -0.5
5.6 70 0.8
19 200 1.1
```

- Columns are separated by blanks. Can be changed before plotting, e.g.,
set datafile separator "," # (comma separated)
set datafile separator "\t" # (separated by tabs)
- plot "file.txt"
→ default: plots 2nd column over 1st column
- plot 'filexyz.txt' using (\$2):(\$3)
→ plots 3rd column over 2nd column
- plot 'data.txt' u (log10(\$1)):(log10(\$2))
→ plots the decadic logarithm of the data in columns 1 and 2 (double-logarithmic plot)

Histogram

is the **graphical** representation of the frequency distribution of some quantity x .

- requires the division of the data (quantity) into **bins** of a **width** Δx (can be constant or variable)
- representation usually by rectangles of width Δx and height corresponding to frequency of occurrence
- can be used to estimate the **probability density function** $p(x)$ of a continuous random variable x

Creating a histogram II

Example: normal distribution / Gaussian distribution

A data set of 10^3 random variables drawn from a Gaussian distribution with

$$N(x; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where the mean value was set to $\mu = 0$ and the variance to $\sigma = 1$, hence the distribution is

$N(x, 0, 1) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2} \cdot x^2) \rightarrow$ so-called standard normal distribution

in some file `gauss.dat` (embedded).



10-DM banknote with a Gaussian distribution

Creating a histogram III

Graphical representation with gnuplot:

- 1 Define the (constant) width of the bins (“bin width”, Δx):

```
gnuplot> bw=0.2
gnuplot> set boxwidth bw
```

... and a so-called “binning” function:

```
gnuplot> bin(x,s)=s*ceil(x/s)
```

The function `ceil(x)` rounds **up** the value of x to next larger integer

- 2 The number of data points (for normalization):

```
gnuplot> N=1000
```

one can also use

```
gnuplot> stats "gauss.dat" ; N = STATS_records
```

- The histogram is then created via:

```
gnuplot> plot "gauss.dat" u (bin($1,bw)-0.5*bw):(1./(N*bw)) \  
    smooth frequency with boxes lc rgb "blue"
```

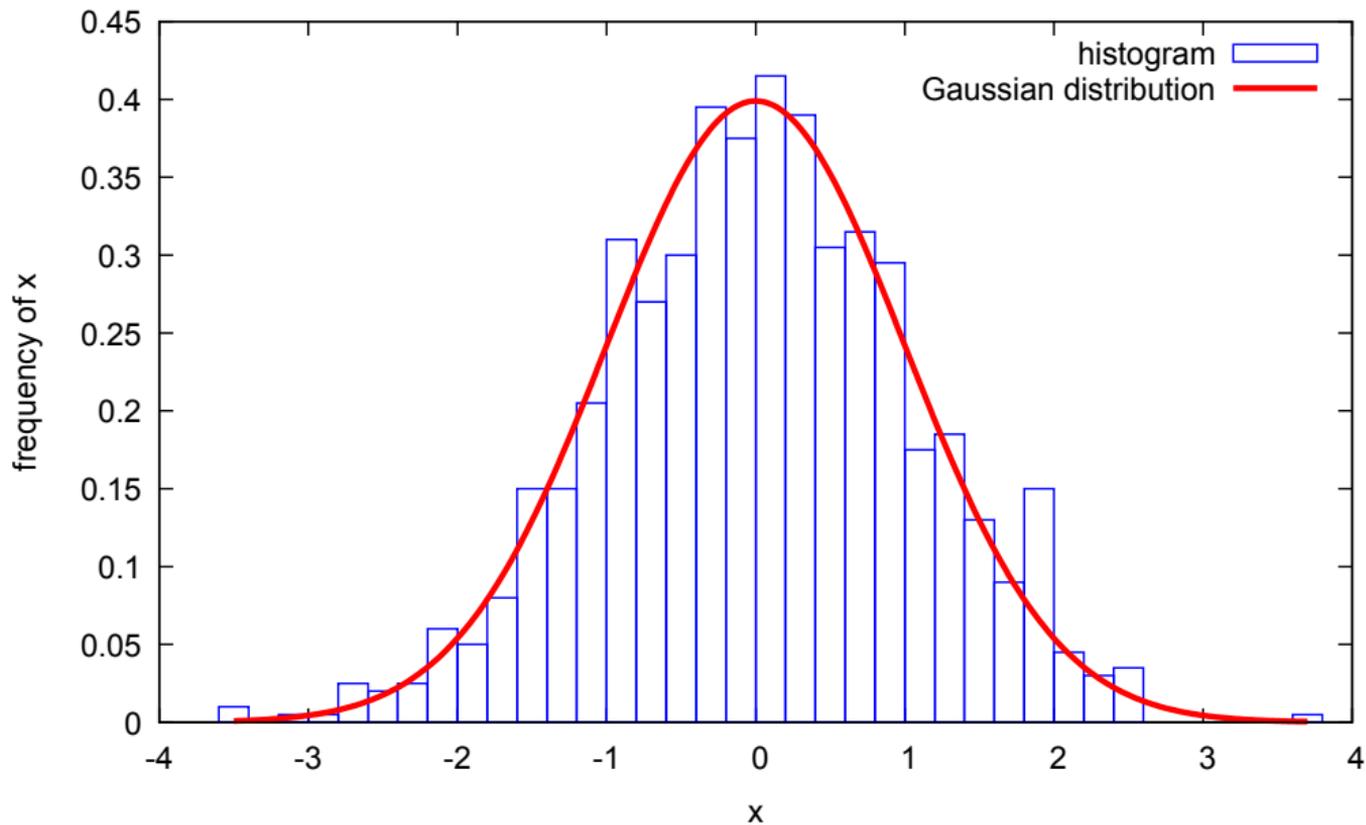
- Plot together with the underlying probability density function

```
gnuplot> gauss(x) = 1./sqrt(2*pi) * exp(-0.5*x**2)
```

in the same diagram:

```
gnuplot> replot gauss(x) with lines linewidth 3 linecolor rgb "red"
```

Creating a histogram V



With help of the Levenberg-Marquardt algorithm `gnuplot` can fit any function with free parameters to data:

- 1 define function:

$$f(x) = a * x + b$$

- 2 fitting examples:

```
fit f(x) "data.txt" via a, b
```

```
fit f(x) "data.txt" u (log10($1)):(log10($2)) via a, b
```

- 3 plotting data and function:

```
plot "data.txt", f(x)
```

If the fit should be done only for a specific x-range:

```
fit [100:300] f(x) "data.txt" via a, b
```

gnuplot supports many different output formats (see `→help terminal`)

- 1 `set terminal pdf enhanced color`
→ sets terminal (output *format*) to colored pdf with special characters
- 2 `set output "myplot.pdf"`
→ name of the file for output (don't forget it!)
- 3 `plot "data.txt", f(x)`
or: `replot`
- 4 either: `set term qt` (resetting terminal to previous output format)
or: `quit`
→ this assures that the plot is *written* to the file (otherwise: empty or incomplete PDF file)

If output is written to PDF or PS file, via option enhanced:

Input	Output in PDF/PS
<code>T_0</code>	T_0 (subscript)
<code>e^{-x}</code>	e^{-x} (superscript)
<code>{/Symbol Qp}</code>	$\Theta\pi$

In addition to the interactive mode, `gnuplot` supports also non-interactive script mode

- write all instructions into an ASCII text file (e.g., “`myplot.gplt`”)
 - comments begin with a `#` (like in makefile and shell)
 - line continuation via backslash `\`
- execute `gnuplot` script from shell:
`gnuplot myplot.gplt`

→ useful for automated PDF creation

→ easy re-use of formatting and plot instructions (labels, sizes, ...)

Example for fitting and pdf output I

```
# embedded file : enthalpie.dat

set terminal pdf enhanced color
set xlabel "1/T [100/K]"
set ylabel "ln(p/p_0)"
ln_p(x) = b + a*x
set fit errorvariables
R=8.314
p_0=1.019
fit [*:] ln_p(x) "enthalpie.dat" \
using (1./((($2)+273.15)):(log((1.019+($1))/1.019))) via a,b
set output "enthalpy.pdf"
plot "enthalpie.dat" \
using (1e2/((($2)+273.15)):(log((p_0+($1))/p_0))) \
with points ps 1 linewidth 3 title "data" \
, ln_p(1e-2*x) with lines linecolor "black" \
t sprintf("enthalpy [kJ/mol]=%5.3f +/- %5.3f",a*R*1e-3,R*a_err*1e-3)
```

Example for fitting and pdf output II

