



993SM – Laboratory of Computational Physics

notes on gnuplot

Maria Peressi

Gnuplot

A simple useful package to:

- plot your data
- save the plot
- data fitting
- movies...

<http://www.gnuplot.info/> to download the package and for details

HERE: few BASIC COMMANDS and FEATURES

Gnuplot

Data files:

```
#prima riga di commento  
#seconda riga di commento  
0.00000 1.00000  
0.0100000 0.904837  
0.0200000 0.818731  
0.0300000 0.740818  
0.0400000 0.670320  
0.0500000 0.606531  
0.0600000 0.548812  
0.0700000 0.496585  
...
```

for instance : `dati.dat`

optional comment lines;
every line with “#” at the beginning

data:
even 1 column is OK
(the progressive number line is considered as “x”)

Running:

```
host$ gnuplot
```

```
GNUPLOT  
... version ....  
.....  
Terminal type set to ...  
gnuplot
```

```
gnuplot> plot 'dati.dat'
```

ALWAYS USEFUL:

```
gnuplot> help [then RETURN]
```

OR SPECIFY SOMETHING, LIKE:

```
gnuplot> help plot using
```

Gnuplot

```
gnuplot> plot 'dati.dat' u 1:2
```

← columns to be used

```
gnuplot> set sample 2000
```

← number points to be used (if plotting a function given by an equation)

more on the same plot:

```
gnuplot> plot 'dati.dat' u 1:2,'dati.dat' u 1:3,'dati.dat' u 1:4
```

or

```
gnuplot> plot 'dati.dat' u 1:2, '' u 1:3, '' u 1:4
```

data in blocks:

separated by 2 empty lines

```
gnuplot> plot 'dati.dat' index 0 u 1:2, '' index 1 u 1:2
```

Gnuplot

Plot options:

`gnuplot> plot 'dati.dat' u 4:6 w l` ← with lines
the defaults is “w p” (with points)

points specifications: **pt** and **ps** (point type and size)

line specifications: **lt** and **lw** (line type and size)

Plot vectors:

from (x,y) to $(x+dx,y+dy)$, giving the columns with x, y, dx, dy :

`gnuplot> plot 'file.dat' using 1:2:3:4 with vectors head filled lt 2`

Plot histograms:

from (x,y) to $(x+dx,y+dy)$, giving the columns with x, y, dx, dy :

`gnuplot> set style histogram`

`gnuplot> plot 'dati.dat' u 1:2 with boxes`

Gnuplot

Action on the data:

you may modify the data before plotting, e.g.:

```
gnuplot> plot 'dati.dat' u (log($1)):(($2+0.5))
```

Use macros:

the list of commands can be prepared in a text file (a “macro.gnu”) and then you can load the macro:

```
gnuplot> plot 'macro.gnu'
```

Gnuplot

Fitting data:

gnuplot> f(x) = a * exp (-x*b)

gnuplot> fit f(x) 'dati.dat' via a,b

Final set of parameters Asymptotic Standard Error

=====
a = 1 +/- 8.276e-08 (8.276e-06%)
b = 10 +/- 1.23e-06 (1.23e-05%)

correlation matrix of the fit parameters:

a b
a 1.000
b 0.671 1.000

gnuplot> plot f(x), 'dati.dat'

Suggestions:

1) It could be useful to give an initial guess of the parameters:

gnuplot> a=1.0, b=1.0

2) it is better to fit with a straight line
(reduce the functional form to a linear form)


Gnuplot

2D plots:

e.g. of $z=f(x,y)$ varying from -1 to 1

data written in the proper way, i.e.:

values of z on an ordered (x,y) grid: keep fix y and vary x ,
then change y and vary again x , ...

every change in y , an empty line is needed: 

```
1.00000
0.904837
....
....
0.818731
0.670320
-0.765234
-0.097823
....
....
-1.000000
-0.876325
....
....
....
....
0.876324
0.876342
```

```
gnuplot> set view 0,0
```

```
gnuplot> set nosurface
```

```
gnuplot> set contour
```

```
gnuplot> set cntrparam levels auto 5
```

which makes 5 levels between min and max, or:

```
gnuplot> set cntrparam levels discrete -1,-0.8,0.4,1
```

which makes only the curves corresponding to the indicated levels; at the end, do:

```
gnuplot> splot "dati.dat"
```

For further info, type as usually:

```
gnuplot> help set cntrparam
```


Gnuplot

Plotting on different windows:

```
gnuplot> set term wxt 0 (for instance)
```

```
gnuplot> plot ... [plot with the options that you need]
```

```
gnuplot> set term wxt 1 ....
```

Saving plots:

```
gnuplot> set term png (for instance)
```

```
gnuplot> set output 'dati.png'
```

```
gnuplot> plot ... [plot with the options that you need]
```

```
gnuplot> set term x11 to be back plotting on the screen
```

Gnuplot

Doing animations:

prepare the data files divided in blocks (to be used with the **index** command)

```
gnuplot > set = 0
```

load a **macro** as following:

```
plot "filename.dat" index set u ...
```

```
pause 1 (wait 1 second)
```

```
set=set+1
```

```
if(set<[last value of set]) replot
```

Then, it is possible to save on a gif file and then use ImageMagic or anything else to make an animated gif. Alternative way:

```
gnuplot > set terminal gif animate delay 20
```

```
gnuplot > set output 'animation.gif'
```

```
gnuplot > stats 'filedat' nooutput
```

```
gnuplot > do for [i=1:100] {plot 'filedat' index (i-1)}
```