Introduction to Gnuplot

(http://gnuplot.ikir.ru)

1 - Introduction

Gnuplot is a terminal-based tool for creating graphs and charts. A user can enter commands to tell gnuplot how to format the graph, or provide a file containing a list of commands. The major advantages of gnuplot are:

• clean, publication-quality graphs are created,

• error bars are easy to display,

• input data is easy to format,

• data or commands can be piped in from stdin,

• control over every aspect of the display,

• output to a variety of pixel and vector formats.

Interface of the terminal includes general information, command line `"gnuplot> "`, and a head-line with menu through which the plot may be altered or the commands may be selected.

```
% gnuplot

        G N U P L O T
        Version 4.0 patchlevel 0
        last modified Thu Apr 15 14:44:22 CEST 2004
        System: Linux 2.4.23

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

        This is gnuplot version 4.0.  Please refer to the documentation
        for command syntax changes.  The old syntax will be accepted
        throughout the 4.0 series, but all save files use the new syntax.

        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/

        Send comments and requests for help to
                <gnuplot-info@lists.sourceforge.net>
        Send bugs, suggestions and mods to
                <gnuplot-bugs@lists.sourceforge.net>


Terminal type set to 'x11'
gnuplot>
```

All the commands including the detailed syntax may be looked up using help:

`"gnuplot> help "`.

terminal tip: use the up and down arrows to cycle through recent commands.

terminal tip: use test terminal to see the style options.

Quit, Save, Load

Commands `exit` or `quit` will close gnuplot. In order to save all the settings, use `save` with a file name:

```
gnuplot> save "savefile.plt"
```

The file "savefile.plt" is a *text* file, which may be edited by any text editor; in order to plot the file, use `load "savefile.plt":`

```
gnuplot> load "savefile.plt"
```

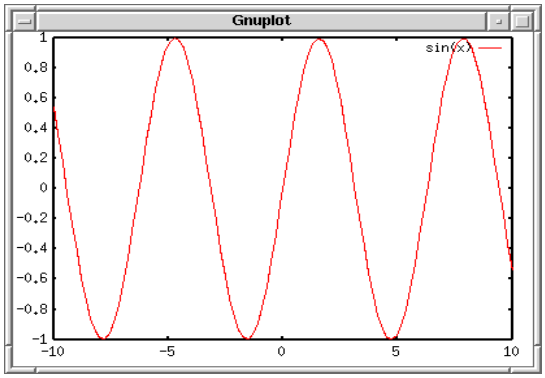
`load "plotfile.txt"` will execute commands from the *"plotfile.txt"* file.

Plotting commands

2D-plots are plotted with `plot` whereas `splot` is used to produce 3D-plots.

Example: plotting a function *y=sin (x)*.

```
gnuplot> plot sin(x)
```



Tip: specify range [0:5] executing the plot command OR use the set command:

```
gnuplot> plot [0:5] sin(x)

gnuplot> set xrange [0:5]
```

Use replot command to add another data/line to the plot.

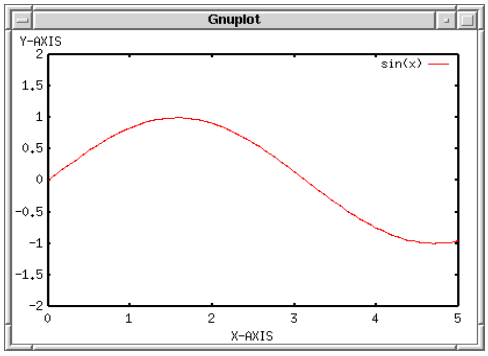
Change settings of the plot

Appearance of the plot may be tuned by the `set` command.

```
gnuplot> help set
 The `set` command can be used to alter different options.  No changes are drawn,
however, until a `plot`, `splot`, or `replot` command is given.

 The `show` command shows their settings; `show all` shows all the settings.

 If a variable contains time/date data, `show` will display it according to  the
format currently defined by `set timefmt`, even if that was not in effect when the
variable was initially defined.

Subtopics available for set:
      angles          arrow           autoscale       bar
      bmargin         border          boxwidth        clabel
      clip            cntrparam       contour         data
      dgrid3d         dummy           encoding        format
.....
      zero            zeroaxis        zlabel          zmtics
      zrange          ztics
```

Here are some examples:

```
gnuplot> set xlabel "X-AXIS"; set ylabel "Y-AXIS"
gnuplot> set xrange [0:5]; set yrange [-2:2]
gnuplot> plot sin(x)
```



The output may be specified by setting a special type of the terminal by `set terminal…`

```
gnuplot> set terminal

Available terminal types:
          unknown  Unknown terminal type - not a plotting device
            table  Dump ASCII table of X Y [Z] values to output
            linux  Linux PC with (s)vgalib
             ....
             tpic  TPIC -- LaTeX picture environment with tpic \specials
         pstricks  LaTeX picture environment with PSTricks macros
          texdraw  LaTeX texdraw environment
               mf  Metafont plotting standard

gnuplot> set terminal postscript
Terminal type set to 'postscript'
Options are 'landscape noenhanced monochrome dashed defaultplex "Helvetica" 14'
```

Command `set output` allows saving the image file with the specified format depending on the terminal chosen.

Alternatively, the standard terminal allows exporting of the figure into some common file types.

```
gnuplot> set output "plot.ps"
gnuplot> plot sin(x)
```

Use `cd "../other/dir"` to select an appropriate data folder.

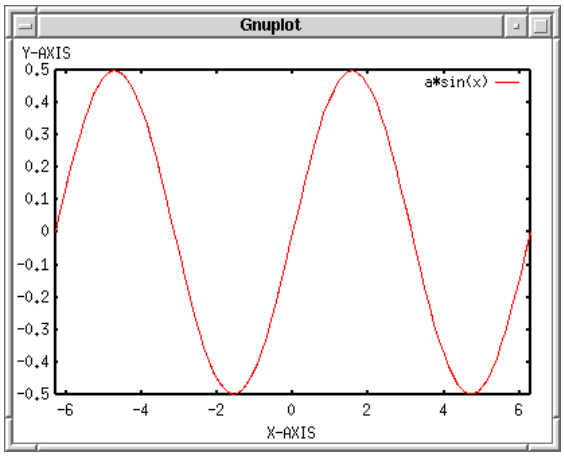
Variable assignment, definition of functions and calculations

Examples:

```
gnuplot> a=10
gnuplot> print a
10
gnuplot> a=1+2*sqrt(3)
gnuplot> print log(a)
1.49606798806764
```

Now *a* may be used for plot. For π use "pi". Example:

```
gnuplot> set xrange [-2*pi:2*pi]
gnuplot> a=0.5
gnuplot> plot a*sin(x)
```



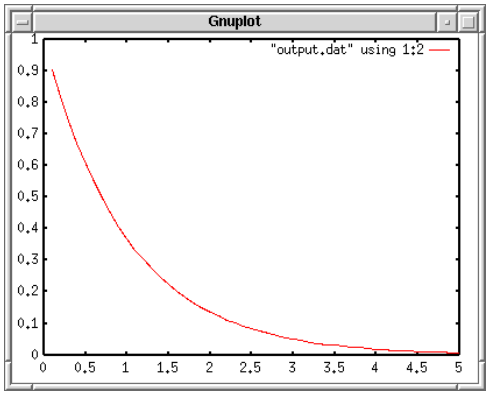
Defining a function: *f(x)=sin(x)*cos(x):*

```
gnuplot> f(x)=sin(x)*cos(x)
gnuplot> plot f(x)
```

2 – Plotting data from a file

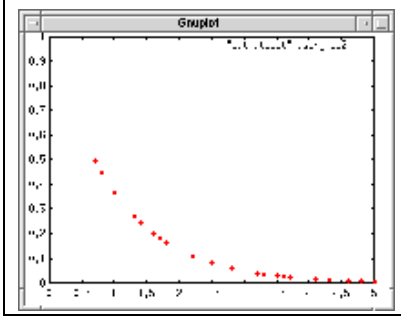
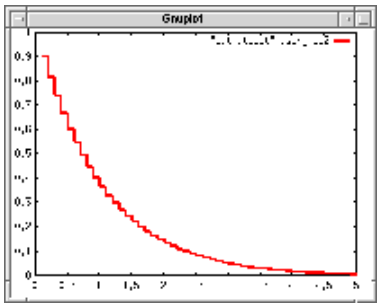
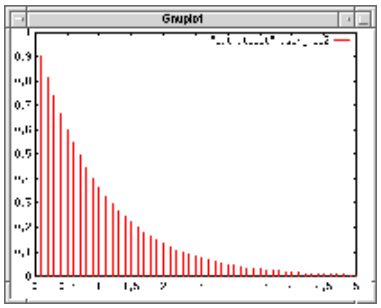
In order to plot data from a file "*output.dat*": specify in what columns the *X-axis* and *Y-axis* values are:

```
gnuplot> plot "output.dat" using 1:2 with lines
```



Modifier `with` is used to specify style options: *lines, points, linespoints,…, linewidth, linecolor, pointsize, pointtype*.

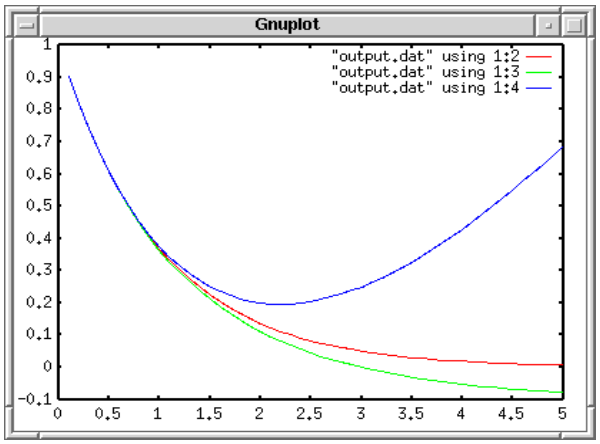
Examples:

points	steps	pulses
		

Plotting of several lines

Use comma to specify a few lines, use "\" to visually split the long line, use replot to repeat the command: `plot "A" using 1:2 with line, "B" using 1:2 with points, ...`

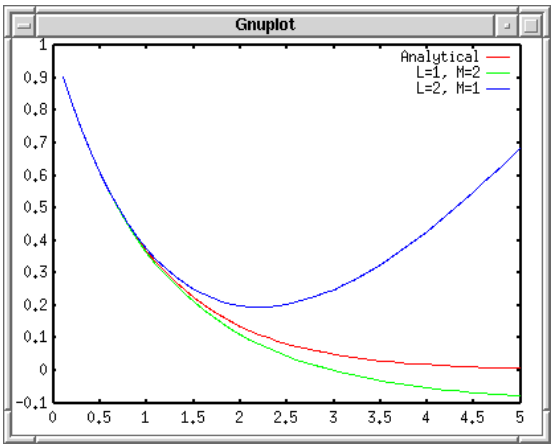
```
gnuplot> plot "output.dat" using 1:2 with lines, \
> "output.dat" using 1:3 with lines
gnuplot> replot "output.dat" using 1:($4*10) with lines
```



Terminal tip: Abbreviation of the commonly used commands available in Gnuplot: e.g., "w" - "with", "i" - "index", etc.

Use title to give a desired name to each line, and lc, lw to specify the *color* and *width* of the lines:

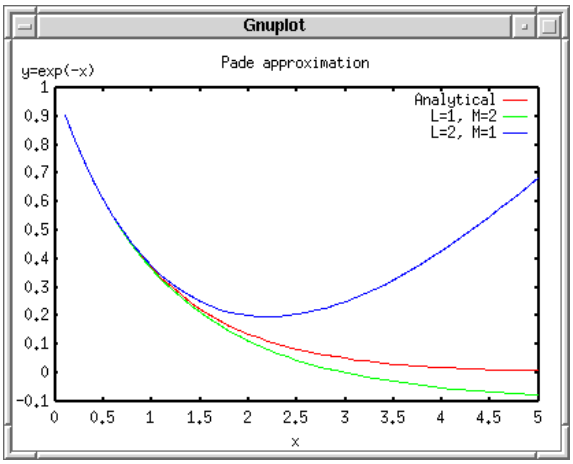
```
gnuplot> plot "output.dat" using 1:2 title "Analytical" with lines lc 1 lw 1, \
> "output.dat" using 1:3 title "L=1, M=2" with lines lc 2 lw 1
gnuplot> replot "output.dat" using 1:($4*10) title "L=2, M=1" with lines lc 3 lw 1
```



Giving names to axes and the plot

Use `set xlabel` and `set ylabel` . Use `set title` to place a head-title on the plot:

```
gnuplot> set xlabel "x"
gnuplot> set ylabel "y=exp(-x)"
gnuplot> set title "Pade approximation"
gnuplot> replot
```



Range of axes, tics, minor tics

Example:

```
gnuplot> set xrange [0:2]
gnuplot> set yrange [0:1]
gnuplot> replot
```

Use `set {x|y}tics`. For example, `set xtics 10`, sets the increment of the scale division to 10.

The minor tics are set with `set m{x|y}tics` *n*, with *n* being the number of the intervals.

```
gnuplot> set xtics 1
gnuplot> set mxtics 5
gnuplot> set ytics 0.5
gnuplot> set mytics 5
gnuplot> replot
```

3 – Plotting data sets from a file

Data file

In the data file (see example below) "#" comments out the line, and the sets of data are split by double Enter.

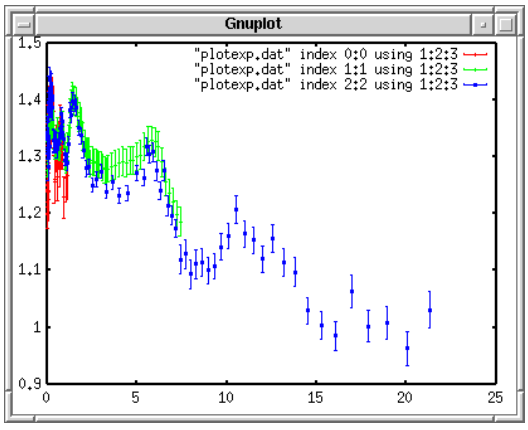
```
# Data No. 1
 2.1500E-02 1.3060E+00 5.3098E-02
 2.3900E-02 1.2220E+00 4.7043E-02
                ....
 1.0350E+00 1.2630E+00 4.1449E-02
 1.1330E+00 1.2670E+00 4.2289E-02
¶
¶
# Data No. 2
 2.4000E-02 1.2970E+00 3.1387E-02
                ....
 7.0000E+00 1.2210E+00 2.5031E-02
 7.4000E+00 1.1860E+00 2.5618E-02
¶
¶
# Data No.3
 2.2500E-02 1.3310E+00 3.4606E-02
 3.5000E-02 1.3440E+00 2.6880E-02
                ....
  2.1296E+01 1.0310E+00 3.1961E-02
```

Plotting, fitting and smoothing the data

With use of command "index X" with *X* being the number of the data set -1 each of the data sets may be addressed. To combine the sets, use `index 0:1`.

To demonstrate Y-axis errorbars: "plot using 1:2:3 with yerrorbars :

```
gnuplot> plot "plotexp.dat" index 0:0 using 1:2:3 with yerrorbars,\
> "plotexp.dat" index 1:1 using 1:2:3 with yerrorbars,\
> "plotexp.dat" index 2:2 using 1:2:3 with yerrorbars
```



Fitting and smoothing the data

The syntax is provided in the help menu. Use the following commands:

Fit

Via

Smooth

4 – Standard operators and functions

Binary operators

Symbol	Example	Result
**	5**3	Power
*	5*3	Multiplication
/	5/3	Division
%	a%b	Reminder of division (for integer numbers only)
+	a+b	Summation
-	a-b	Subtraction
==	a==b	Comparison of two numbers by equality
!=	a!=b	Comparison of two numbers by the inequality
<	a<b	Less than
<=	a<=b	Less than or equal
>	a>b	Greater than
>=	a>=b	Greater than or equal
&	a&b	Bitwise AND
^	a^b	Bitwise OR
|	a|b	Bitwise exclusive OR (XOR)
&&	a&&b	Logic AND
||	a||b	Logic OR

Unary operators

Symbol	Example	Result
-	-a	Unary minus
+	+a	Unary plus
~	~a	One's complement
!	!a	Logic negation
!	a!	Factorial
$	$3	Choice of column

Standard functions

Function	Result
abs(x)	absolute value of x, |x|
acos(x)	arc-cosine of x
asin(x)	arc-sine of x
atan(x)	arc-tangent of x
cos(x)	cosine of x, x is in radians.
cosh(x)	hyperbolic cosine of x, x is in radians
erf(x)	error function of x
exp(x)	exponential function of x, base e
inverf(x)	inverse error function of x
invnorm(x)	inverse normal distribution of x
log(x)	log of x, base e
log10(x)	log of x, base 10
norm(x)	normal Gaussian distribution function
rand(x)	pseudo-random number generator
sgn(x)	1 if x > 0, -1 if x < 0, 0 if x = 0
sin(x)	sine of x, x is in radians
sinh(x)	hyperbolic sine of x, x is in radians
sqrt(x)	the square root of x
tan(x)	tangent of x, x is in radians
tanh(x)	hyperbolic tangent of x, x is in radians