Gnuplot

Gnuplot is a free program that plots 2D and 3D data. It is what I use now on my linux machines to make publication quality scientific plots. Along the learning curve, I made some notes for myself. Perhaps they will also be useful to you. You might also try the very nice Los Alamos site by Kawano, or the demo scripts

```
# RUNNING FROM A FILE OF COMMANDS
# Probably the easiest way to work. I edit the command file in one window, have a second
# window running gnuplot where I simply keep loading in the new command file and it makes a new
# eps file each time I load, and then a third window running ghostview to view the eps file.
# To load a command file:
  load 'junk.gnu'

# LIMITS
# the noautoscale, xrange, yrange change the x- and y-limits
  set noautoscale
  set xrange [0.5:10]
  set yrange [300:480]
# or plot ranges directly
  plot 'junk.dat' [0.5:10][300:480]
# or choose autoscale for one axis only
  set autoscale x

# STYLE OF PLOT
# choice of points, lines, linespoints, steps, boxes, errorbars, impulses, etc.
# for errorbar options type 'help plot errorbars'
  set data style linespoints

# POINT SIZE AND TYPE
# pointsize is to expand points
  set pointsize 2.5
# type 'test' to see the colors and point types available
# lt is for color of the points: -1=black 1=red 2=grn 3=blue 4=purple 5=aqua 6=brn 7=orange 8=light-brn
# pt gives a particular point type: 1=diamond 2=+ 3=square 4=X 5=triangle 6=*
# postscipt: 1=+, 2=X, 3=*, 4=square, 5=filled square, 6=circle,
#            7=filled circle, 8=triangle, 9=filled triangle, etc.

# LINE COLORS, STYLES
# type 'test' to see the colors and point types available.
# Differs from x11 to postscript
# lt chooses a particular line type: -1=black 1=red 2=grn 3=blue 4=purple 5=aqua 6=brn 7=orange 8=light-br
# lt must be specified before pt for colored points
# for postscipt -1=normal, 1=grey, 2=dashed, 3=hashed, 4=dot, 5=dot-dash
# lw chooses a line width 1=normal, can use 0.8, 0.3, 1.5, 3, etc.
# ls chooses a line style
  plot sin(x)k with linespoints lt 2 pt 4

# DRAW A LINE OR ARROW BETWEEN TWO POINTS
# use the arrow command with nohead to draw just a line:
  set arrow from 1,2 to 4,8.4 nohead lt -1 lw 1.2

# STORING AND USING A LINE TYPE AND POINT TYPE
# The linestyle command is a shortcut to changing the points and line types each time
# to store the current setup
  set linestyle
```

```
# to view the stored linestyles
  show linestyle
# to use it
  plot sin(x) ls 1
# to erase the styles
  set nolinestyle

# VECTORS, PLOTTING DATA FROM A FILE, PLOTTING SINGLE POINTS
# Instead of loading variables into a vector and multiplying
# vectors together, I usually just make a table where the values
# of each vector are represented by a column. Then follow the description
# below to do arithmetic on the columns with awk or the 'plot using' command
# To make a table that has column1 = 0.05, 0.1, ... 1.0; column2 = 2*column1+14
  yes | head -20 | awk '{print NR/20., NR/10.+14}' >! table.dat

# If you are using Solaris, you may not have 'yes', so code it (fortran) via
        open(unit=0,file="/dev/null")
 10     write(6,'("y")')
        goto 10
        end

# The 1:2 means plot x from column 1, y from column 2
  plot 'table.dat' using 1:2

# Note, to do SM-like arithmetic on columns,
# where $n means column n
  plot 'table.dat' using ($3/$1):($2*134.44)

# to plot lines 4-20 from a file with filled square points and double-weight solid lines
# 'w lp lt 1 lw 2 pt 5' means 'with linespoints linetype 1 lineweight 2 pointtype 5'
  plot "< awk 'NR>3 && NR<20 {print $1,$2}' axis.dat" w lp lt 1 lw 2 pt 5

# to place a single 'X' point at x=4, y=5 (useful for keys)
  plot "< echo 4 5" w p pt 2

# Comments in a file begin with '#' and are ignored
# For multiple graphs, a blank line in a file is interpreted as
# 'lift the pen'. Two blank lines indicate a new 'index'. To plot
# only the data from indices 4 through 6 in a file, where index 0 is
# the first index,
  plot 'file' index 4:6

# PLOTTING A FUNCTION
# first define the function, then plot
  plot f(x) = sin(x*a), a=0.2, f(x) with points
# Make 2 plots, first with points and second with a line, note abbreviations w, l, p
  plot f(x) = sin(x*a), a=0.2, f(x) w p, a=0.4, f(x) w l

# MAKING A SQUARE OUTPUT GRAPH
  set size square

# MULTIPLE GRAPHS AND LINES
  set multiplot # Do at beginning, after 'set term' and 'set output' commands
                # then do 'set origin', 'set tmargin', 'set size' etc. for each
                # separate plot
  set nomultiplot #The last line in the command file
# location and sizes of smaller plots, viewport is [0:1,0:1]
  set size 0.5, 0.5
  set origin 0.0, 0.5
# to eliminate offsets between plots (units are character sizes)
  set tmargin 0
  set bmargin 0
# losing the labels?  include them before the last plot command
#
# All labels apply to all subsequent graphs in multiplot mode.
# To relabel each graph individually, shut off the labels after 'set origin' via
  set nolabel
# You can also name, and then shut off an individual label, e.g.
  set label 2 "string" at 3,4
  set nolabel 2
# By default labels are numbered sequentially throughout the command file.

# LATEX OF MULTIPLE PLOTS
# Either make a single plot with multiplot as above, or make each plot
# individually and combine in the latex document. The following from the
#  Kawano site:
\documentclass{article}
\usepackage{graphics}
\begin{document}
\begin{figure}
  \begin{center}
    \begin{tabular}{cc}
      \resizebox{60mm}{!}{\includegraphics{test1.eps}} &
      \resizebox{60mm}{!}{\includegraphics{test2.eps}} \\
      \resizebox{60mm}{!}{\includegraphics{test3.eps}} &
      \resizebox{60mm}{!}{\includegraphics{test4.eps}} \\
    \end{tabular}
    \caption{This is sample figures.}
    \label{test4}
  \end{center}
\end{figure}
\end{document}

# FITS TO DATA
  f1(x) = a1*tanh(x/b1)           # define the function to be fit
  a1 = 300; b1 = 0.005;           # initial guess for a1 and b1
  fit f1(x) 'force.dat' using 1:2 via a1, b1

  f(x) = a*x + b                          #plot the data and a linear fit
  fit f(x) 'try.dat' using 1:2 via a,b
  plot 'try.dat' using 1:2 w p pt 3, f(x)

# TIC MARK CONTROL
# mytics is number of small intervals marked between major tics in y
# if you want 4 minor tics between each major tic in y
  set mytics 4
# ytics controls major tics on y-axis
# the logscale is to plot logs
# a typical log plot will have
  set logscale x
  set xrange [1:21]
  set xtics 1,10,100
  set mxtics 10
# to shut off tics:
  set noxtics
  set noytics
```

```
# For tic marks every 0.02, when x increases to left
set xtics 3.64,-0.02,3.40
# Various specific tic marks and their labels
  set xtics ("low" 0, "medium" 50, "high" 100)
  set xtics (1,2,4,8,16,32,64,128,256,512,1024)
  set ytics ("bottom" 0, "" 10, "top" 20)
# size of tic marks
# first number is for major tics, second for minor
# defaults are 1 and 0.5, negative numbers put tics on outside
  set ticscale 2 1

# BOX, AXIS LABEL CONTROLS
# The default labels on the graph are often too small. To make the default size bigger
# define it in the terminal command, as below. For some inexplicable reason here
# the fontname is in quotes, then a space, then the number, as opposed to a quote,
# fontname, comma, number, endquote, the way it normally behaves.
# The terminal command is the first line in the command file
  set terminal postscript eps enhanced "Helvetica" 30
# axis labels
  set xlabel "Distance (AU)" font "Helvetica,24"
# To turn off x-labels of tic marks
  set format x ""
# In a log plot, make x-axis labels 10, 100, 1000, and then 1e4, 1e5...
# The '3' controls where the format swtiches to exponential
  set format x "%.3g"
# To not draw the box when you plot
  set noborder
# to eliminate offsets between plots (units are character sizes)
# see multiple plots
  set tmargin 2
  set bmargin 2

# LABELS
# Example of a label on the x-axis (the 22 is point size; Bold is another font to try)
  set label "3.52" at 3.52,-1.65 center font "Helvetica,22"
# Example of a y-axis label
  set label "log (L/L_\o)" at 3.66,-0.5 center rotate font "Helvetica,24"
# to put label "y=x" at location (1,2), rotated by 43 degrees:
  set label "y=x" at 1,2 rotate by 43 font "Helvetica,20"
# same as above but going to the right of (1,2):
  set label "y=x" at 1,2 right
# to put label "y=x" at center of graph:
  set label "y=x" at graph 0.5,0.5
# Inserting special symbols, greek letters and italics, subscripts, superscripts:
# These only look right on the final .eps file if you include the
# enhanced option: set term postscript eps enhanced -- add this to the fig.gnu file
# along with set output 'fig.eps', then run gnuplot fig.gnu in another window and
# gv fig.eps in a third window
  set xlabel 'Flux, F_{/Symbol L} '
  set ylabel 'Intensity, {/Times-Italic I}_o'
  set xlabel "N_{e} (cm^{-3})"
# {/*0.75 K} is a K at three-quarters of whatever fontsize is currently in effect.
# {/Symbol=20 G} is a 20-point GAMMA
# For special symbols specify \char-code (in octal)
#  e.g., {/Symbol \245} is the symbol for infinity.
#       {/Symbol \305} is the Earth
#       {/Symbol \360} is the degree symbol
#       {\247} is the section symbol
# There does not appear to be a sign for the Sun
# These are all stored in /usr/local/gnuplot-3.7/docs/ps
# For Angstroms, set encoding iso_8859_1, {\305}; then set encoding default
#
# To get a space the size of a string type &{string} [enhanced ps only]
#
# If you are losing labels on multiple plots, see MULTIPLE GRAPHS AND LINES section above

# ERASE SCREEN
  clear

# GRID
# this does a grid
  set grid

# MINIMUM VALUES
# set smaller values than this to zero
  set zero 1e-30

# DEFAULT LABEL AND KEY FOR GRAPH
# to turn off the annoying label in the upper right corner
  set nokey
# this key has a title
  set key 0.018,150 title "F(x) = A tanh (x/B)"
# when you plot with a 'title' this goes into the key
plot    "force.dat" using 1:2 title 'Column data' with points 3, \
        "force.dat" using 1:3 title 'Beam data'   with points 4, \
        a1 * tanh( x / b1 ) title 'Column-fit: A=309, B=0.00227', \
        a2 * tanh( x / b2 ) title 'Beam-fit: A=260, B=0.00415'

# TITLE
# multiple lines to the title
set title "Force Deflection Data \n and curve fit"

# STORE A POSTSCRIPT OUTPUT FILE
# use the 'set output' command, the second command in the file
  gnuplot> set terminal postscript eps enhanced
  Terminal type set to 'postscript'
  Options are 'eps enhanced monochrome dashed defaultplex "Helvetica" 14'
  gnuplot> set output "file.eps"
  gnuplot> replot
  gnuplot> set output              # set output back to default
  gnuplot> set terminal x11        # ditto for terminal type

# HARDCOPIES: LANDSCAPE PLOTS, THICKER LINES, LOCATION OF PLOTS, ETC.
  The postscript commands used in gnuplot are described in /usr/local/gnuplot/docs/ps/ps_file.doc
  You can edit this file in several places easily. For example, '/gnulinewidth 5.000 def' sets
  the line thicknesses, and '50 50 translate' '0.050 0.050 scale' set the location and scale of the plot.
  The postscript commands used in gnuplot are described in
```

/usr/local/gnuplot/docs/ps/ps_file.doc

```
  You can edit this file in several places easily. For example, '/gnulinewidth 5.000 def' sets
  the line thicknesses, and '50 50 translate' '0.050 0.050 scale' set the location and scale of the plot.
  The default size seems to come out better without the 'eps', i.e., 'set terminal postscript enhanced'.
  Try '90 rotate' down next to the translate commands to spin the graph 90 degrees.
```

Good luck!!

Back to Hartigan's Home Page

Patrick Hartigan
hartigan@sparky.rice.edu