

C2110 *UNIX and Programming*

10th Lesson

gnuplot, bash

Petr Kulhánek

kulhanek@chemi.muni.cz

National Centre for Biomolecular Research, Faculty of Science,
Masaryk University, Kamenice 5, CZ-62500 Brno

Contents

➤ **Presentation of Scientific Data**

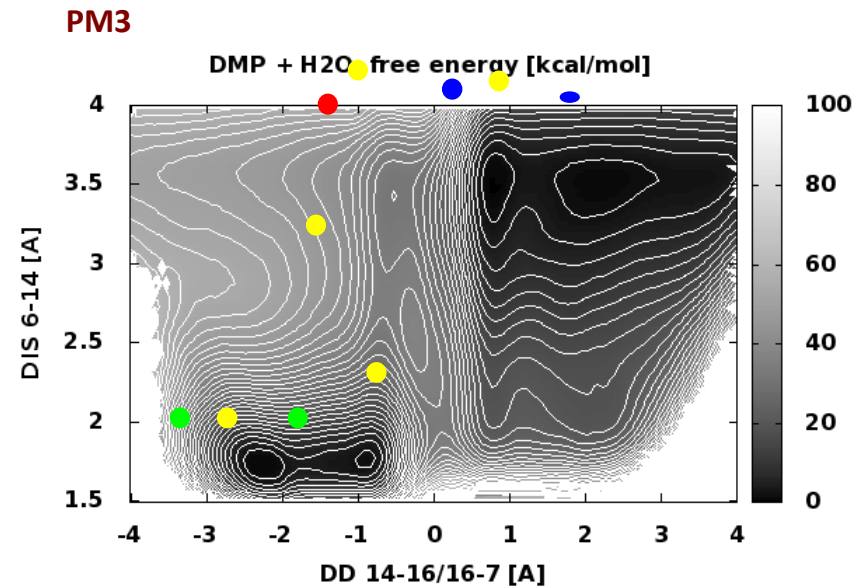
➤ **Gnuplot**

- **overview of the Gnuplot language**
- **command plot, setting and description of axes**
- **terminals**
- **splot command**

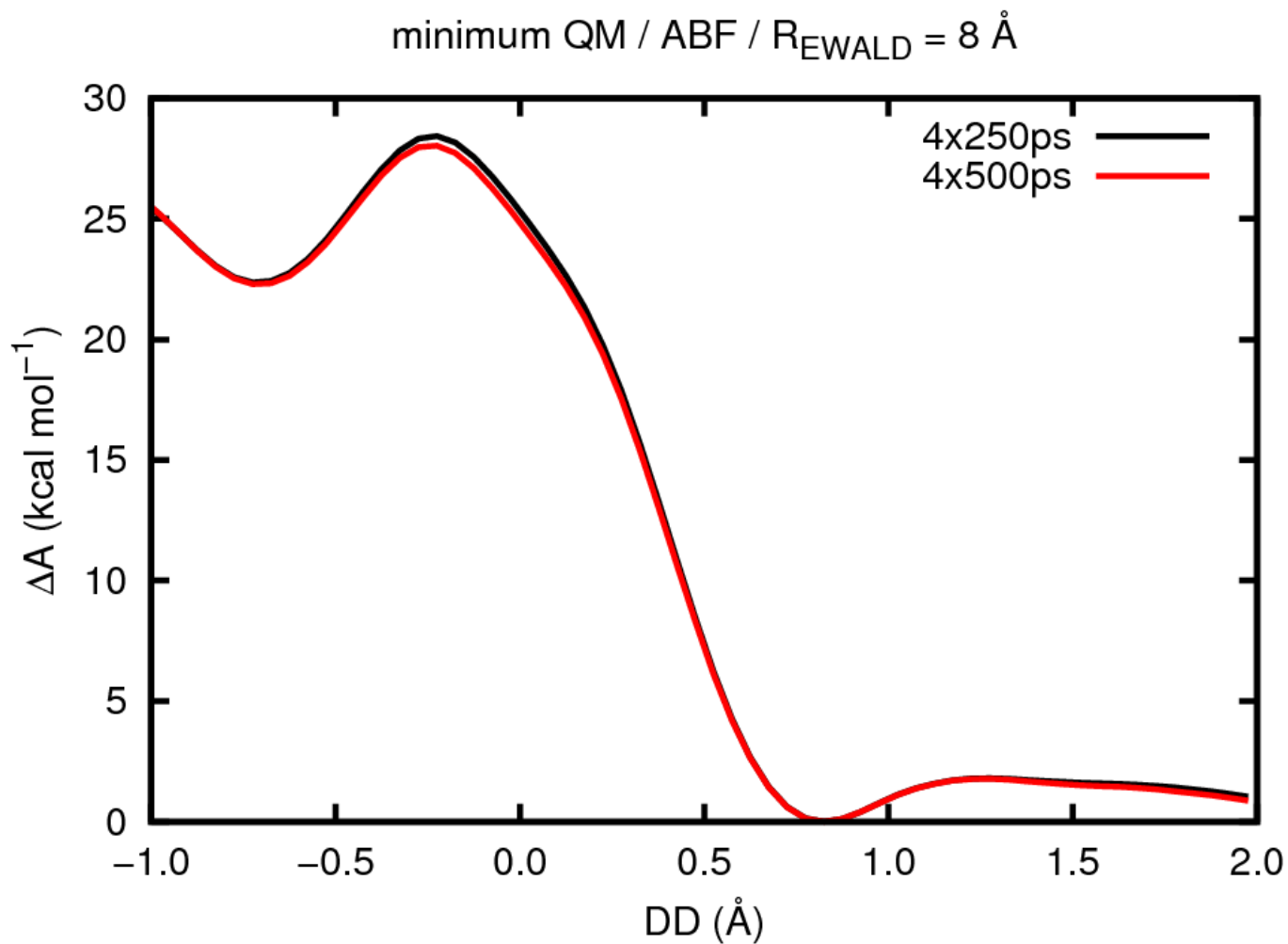
Gnuplot

<http://www.gnuplot.info/>

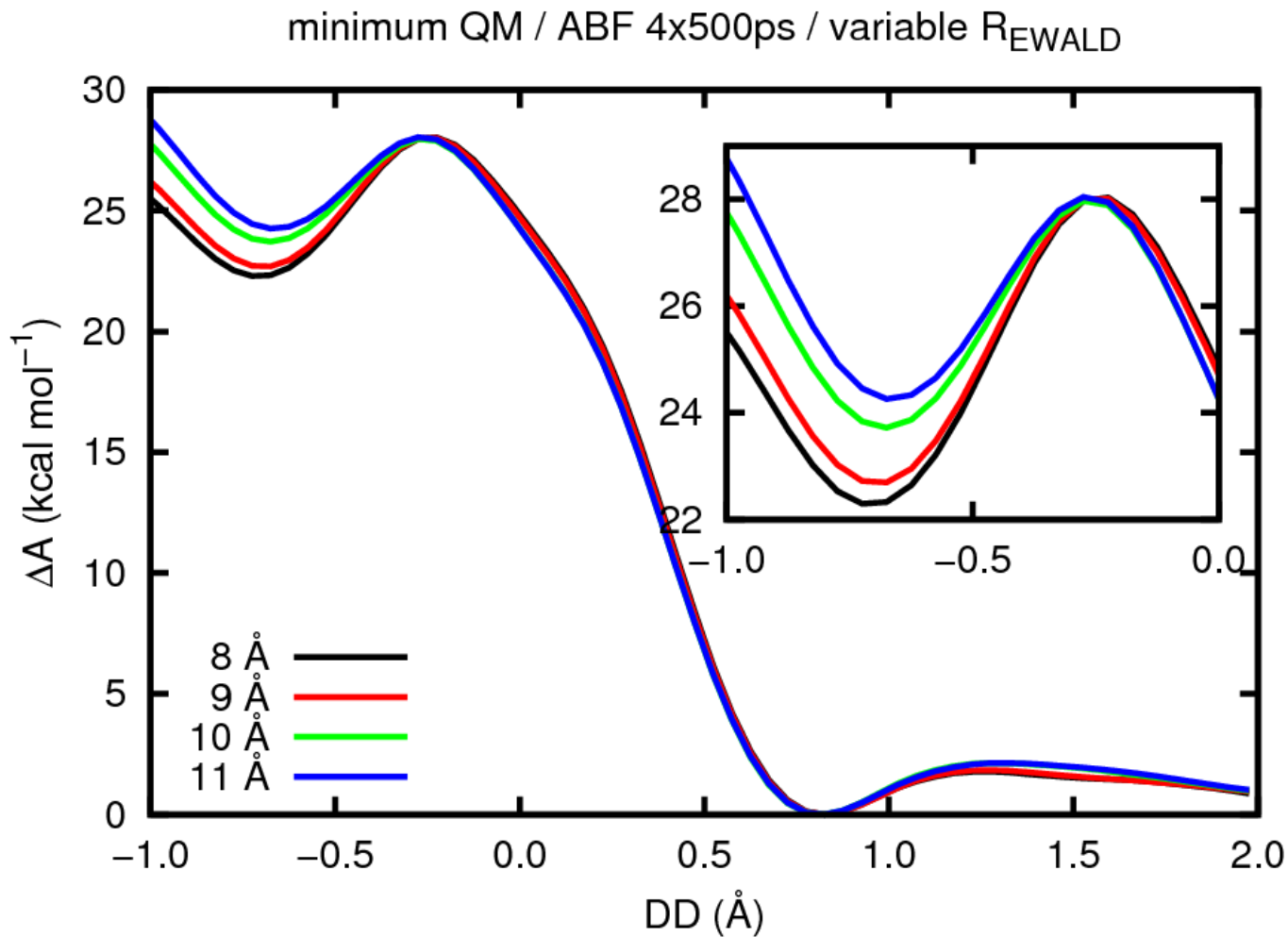
(documentation, tutorials, source codes)



Example



Example



Interactive mode

Gnuplot is used for rendering of 2D and 3D graphs and enabled work in interactive or scripting regime.

Interactive mode

```
[kulhanek@wolf ~]$ gnuplot
```

```
G N U P L O T
Version 4.4 patchlevel 3
last modified March 2011
System: Linux 3.2.0-31-generic
```

```
Copyright (C) 1986-1993, 1998, 2004, 2007-2010
Thomas Williams, Colin Kelley and many others
```

```
gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:   type "help seeking-assistance"
immediate help:   type "help"
plot window:      hit 'h'
```

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

shell Bash command line



gnuplot command line



Non-interactive mode

1) Indirect launch

We start language interpreter and state the name of the script as an argument.

```
$ gnuplot my_gnuplot_script
```

Scripts do not have to have set flag **x** (**executable**).

2) Direct launch

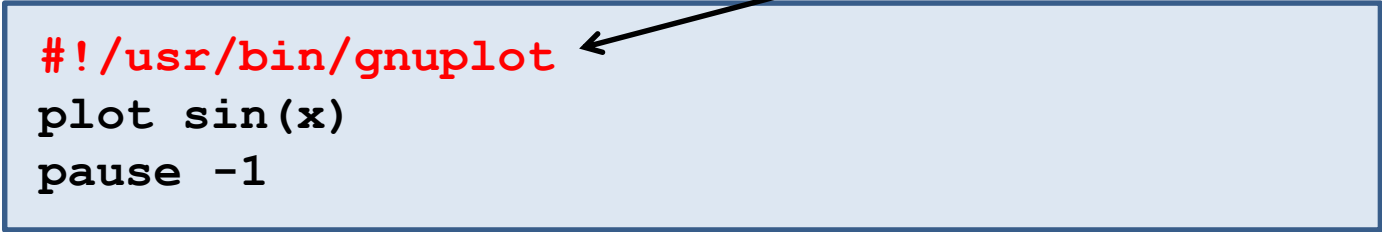
We start the script directly (shell automatically starts the interpreter).

```
$ chmod u+x my_gnuplot_script
```

```
$ ./my_gnuplot_script
```

Scripts must have flag **x** (**executable**) and contain **interpreter** (part of the script).

```
#!/usr/bin/gnuplot  
plot sin(x)  
pause -1
```



Command - plot

```
> plot function/file [display_settings] [, function/file ...]
```

Displays XY graph of a function or data series contained in the file.

Examples:

lines, points, linespoints, dots

color of line

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

```
> plot cos(5.7*x+3.4) with points linecolor rgb "red" \  
linewidth 2 title "cos"
```

data file name

line width

key

```
> plot "input.txt" using 1:2 with lines
```

the second column contains the y values

the first column contains the x values

```
> plot sin(x) , cos(x)
```

displays the functions sin and cos in one graph

Exercise I

1. Plot the graph of $y = x^2$ function
2. Plot the graph from previous task with blue color
3. Display dependence of temperature on time contained in the file:
`/home/kulhanek/Documents/C2110/Lesson10/temp.txt`

Time is in the first column, temperature is in the second column.

4. Display function $\sin(x)$ using red line and $\cos(x)$ using orange line and points into one graph.

Perform tasks in interactive mode

Other commands

- > **set title** "description" # graph header
- > **set xrange** [min_value:max_value] # sets range of x axis
- > **set xlabel** "description" # sets description of x axis
- > **set yrange** [min_value:max_value] # sets range of y axis
- > **set ylabel** "describtion" # sets description of y axis
- > **set nokey** # does not display key for all data series
- > **pause -1** # waits for pressing of a key

Exercise II

1. Write a script, which will plot function $y = x^2$ in the range of 0-10 for the x-value. Launch the script indirectly using gnuplot interpreter.
2. Write a script that will display dependence of temperature on time contained in the file `/home/kulhanek/Documents/C2110/Lesson10/temp.txt`

Put labels on axis including units, time is in picoseconds, temperature is in kelvin.

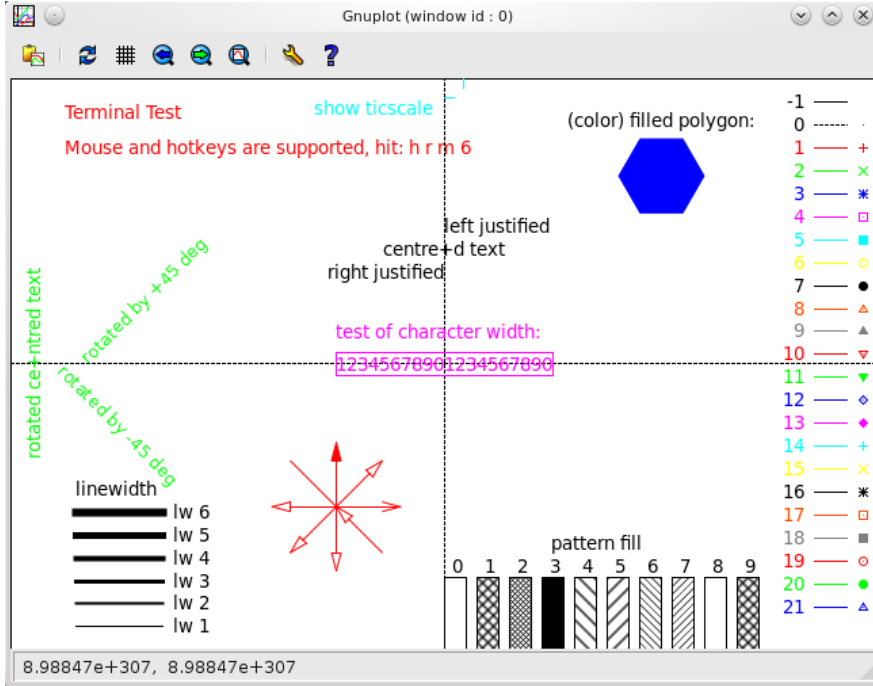
Terminals

Terminal determines where the graph is going to be plotted.

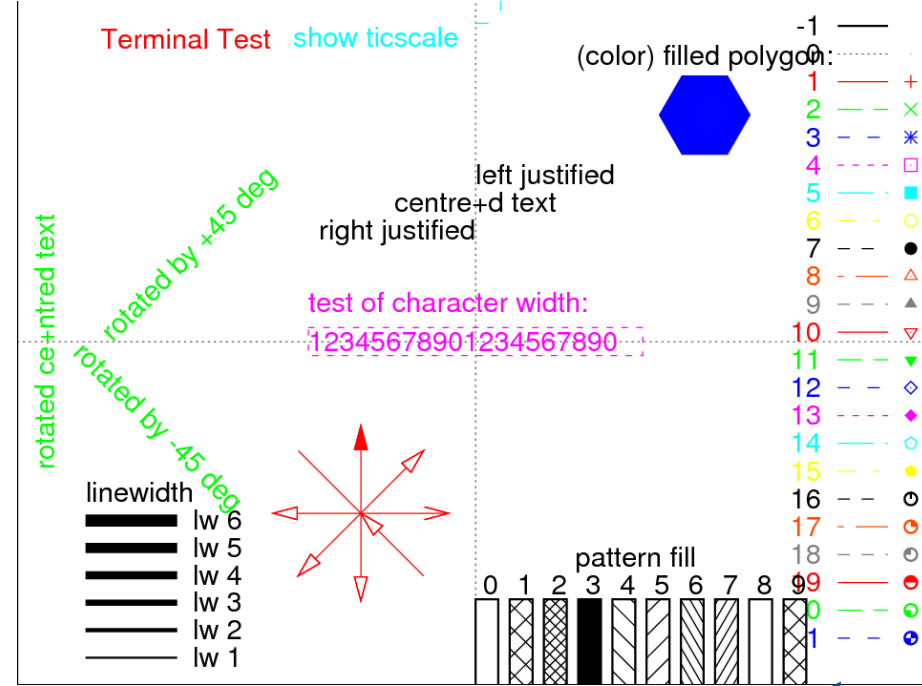
- > **set term x11** # output is rendered into a window
- > **set term wxt** # output is rendered into a window (better properties)
- > **set term png size 800,600**
output is rendered as a png image
- > **set output "output.png"** # output is saved to the "output.png" file
- > **test** # prints page demonstrating properties of given terminal (not all terminals have the same output options)

Output from different terminals

wxt



postscript/eps



Support of dashed lines



Exercise III

1. What are properties provided by terminals x11 and wxt. Work in interactive mode and use command test
2. Write a script, which will plot function $y = 5x^3 + 6x^2 - 7$ in the range from -10 to 5 for x-axis. Run the script directly specifying the interpreter in the header of the script.
3. Modify the previous script so that the graph is rendered as a png image into a file. Size of the image will be 640x480. Display the image file using command display.
4. Display the result of the command test for the terminal png and postscript.
5. What are type of terminal supported by gnuplot (set terminal command without an argument)?

spot command

spot command can be used to display functions of two variables.

```
> spot function/file [display_settings] [, function/file ...]
```

Displays XYZ graph of a function or data series contained in a file.

View is set by the command **set view a, b**, where **a** and **b** are directional angles. Top view can be set by **set view map**

While visualizing functions, density of the sampling for x-axis and y-axis can be set by command **set isosamples a, b**, where **a** and **b** indicate the number of samples in the given direction.

Surface can be colored based on the functional value by using display option **pm3d**, e.g.:

```
> spot x*x+y*y with pm3d
```

Exercise IV

1. Display function $x^2 + y^2$
2. Set view from the top (command set view)
3. Disable view from the top (unset view)
4. Increase the density of points for the visualization of the function (set isosamples).
Use values 10,20; 20,10 and 20,20
5. Use pm3d display option
6. Set view from the top (set view)

Work in the interactive mode

Bash

- **Redirection of the input from a script**

Redirection of input from script

Redirection of standard input of program `my_command` from the script file.

```
.....  
./my_command << EOF  
first line of text  
second line of text  
third line of text  
EOF  
.....
```

sign indicates the end of input
(user selectable)

text, that is loaded as input

end of input, *sign must not be surrounded by spaces*

This type of redirection is particularly advantageous in scripts but works also in the command line. The advantage is the expansion of variables in read text.

Examples

```
#!/bin/bash

for ((I=1;$I<=10;I++)); do
    NAME=`printf "%02d.txt" $I`
    cat << EOF > $NAME
    This is a file number: $I
EOF
done
```

Result of commands in graves `` is assigned to variable NAME.

Highlighted text is sent to the **standard input** of cat command, variables are expanded before sending the output, then cat saves the output to the file \$NAME.

```
#!/bin/bash

gnuplot << EOF
plot sin(x)
EOF
```

This construction allows to automatically create scripts for gnuplot.

Exercise V

1. Create a script that will create 360 images with size 800x600 of function $\sin(x + \text{offset})$ for x in the interval from 0 to 2π , where the constant for offset will vary gradually from 1 to 360° .
2. Create a script that will create ten files. The file name will be in format XX.txt, where XX is the file number. If the file number is less than ten, use 0 as the first digit. Each file will contain the following text (where X is file number):

`Automatically generated text file`

`File number is: X`