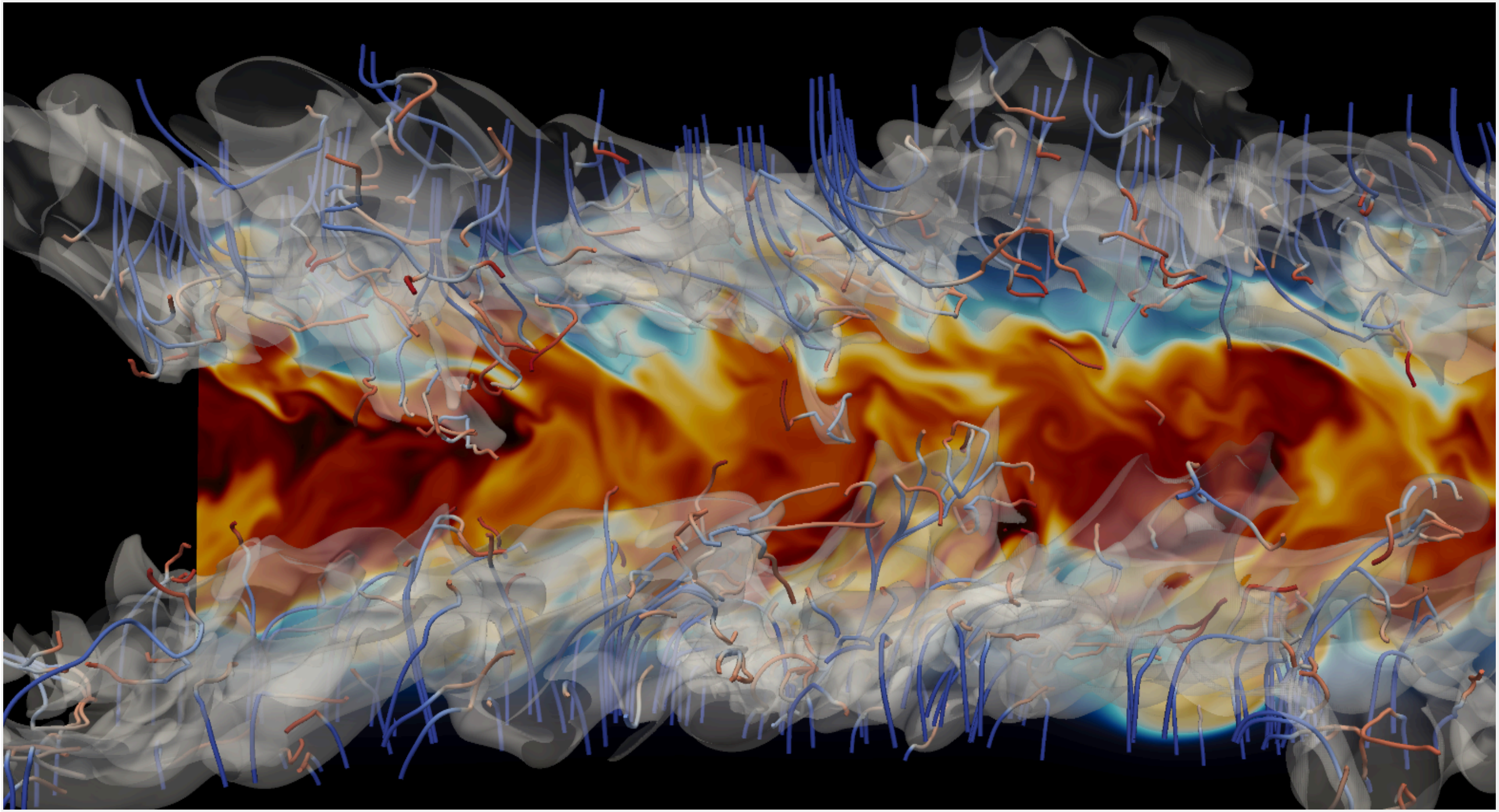
Software Tools for UNIX/Linux Systems

Part 9: Visualization and Documentation

C. Hasse

TECHNISCHE
UNIVERSITÄT
DARMSTADT

STFS



Visualization using gnuplot

Overview

1. Basics
    1. gnuplot
    2. Basic Commands
    3. Local – Global Definitions

2. Plotting Data from Files
    1. File Structure
    2. Plot Settings

3. Layout
    1. Axis, grid
    2. Multiplots
    3. Linestyles

Overview

1. Basics
    1. gnuplot
    2. Basic Commands
    3. Local – Global Definitions

2. Plotting Data from Files
    1. File Structure
    2. Plot Settings

3. Layout
    1. Axis, grid
    2. Multiplots
    3. Linestyles

Basics

General Information about gnuplot

- Plotting program (gnu *not* from GNU-Licence)
- Probably widest used open source tool
  dates back to 1986
- able to plot functions and data 2D and 3D
- Interactive or using input Files

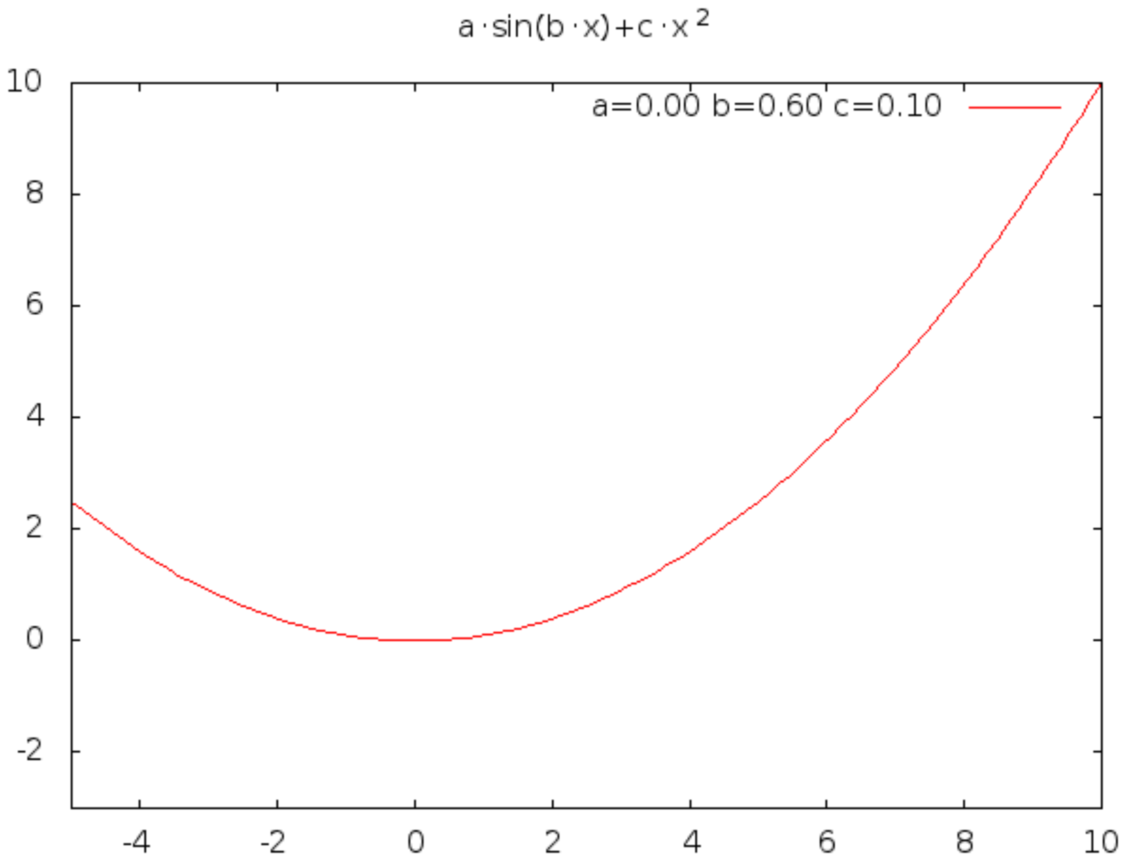
1.1 General Information about gnuplot

- Advantages:
  - Open Source (close to)
  - platform variable
  - robust to input data
  - allows handling/modification of input data

- Disadvantages:
  - 3D Data (→ better Paraview etc.)

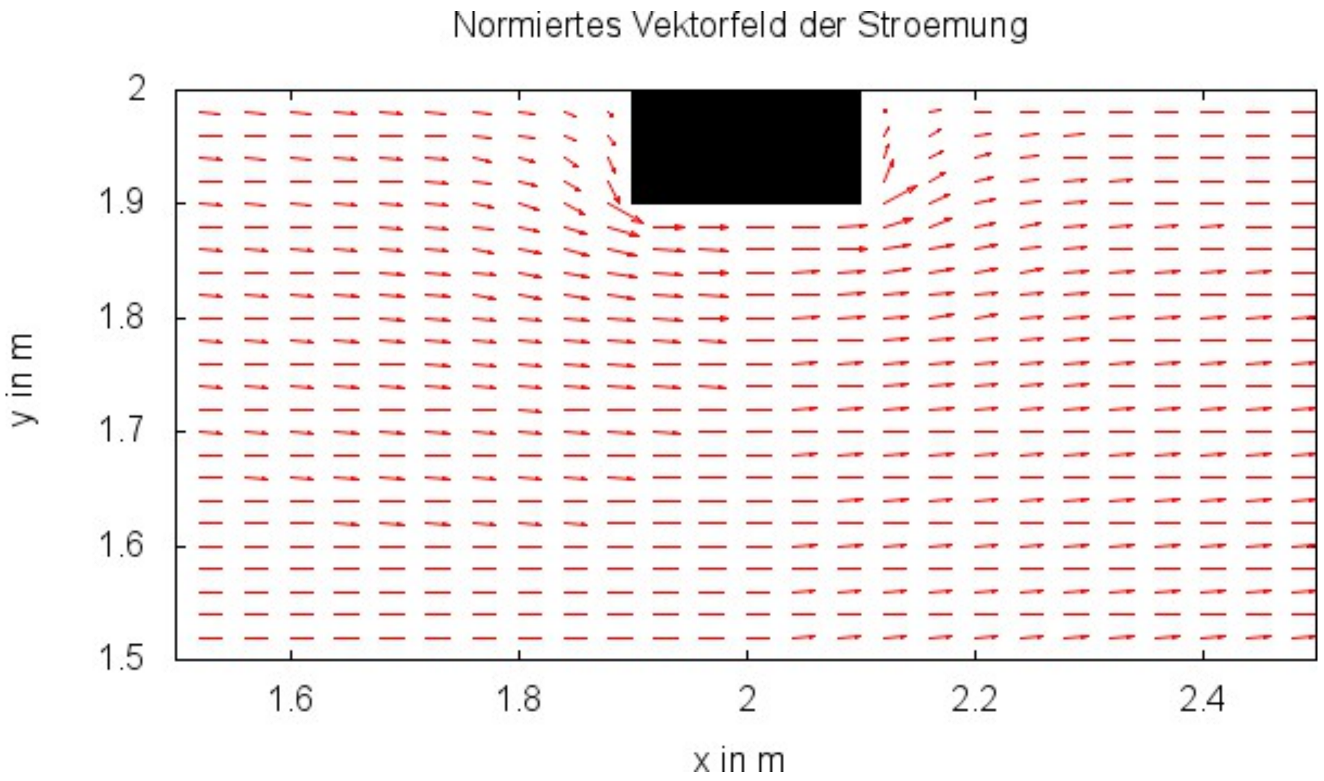
Examples:

- animated plots:



Examples:

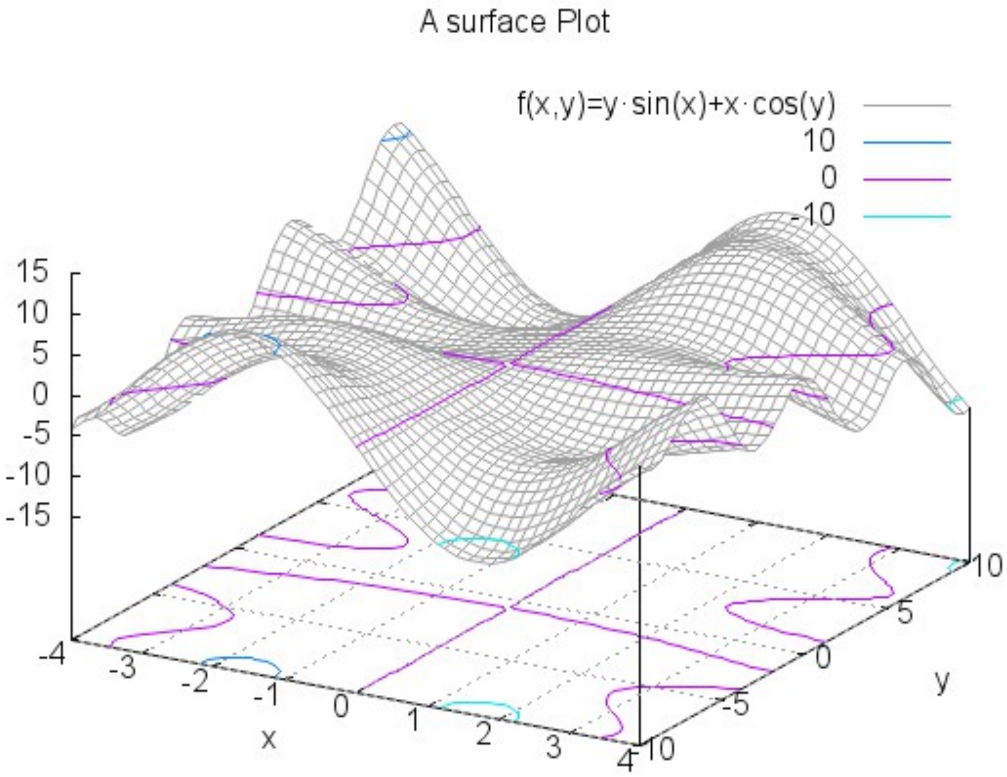
- animated plots
- vector plots



Introduction

Examples:

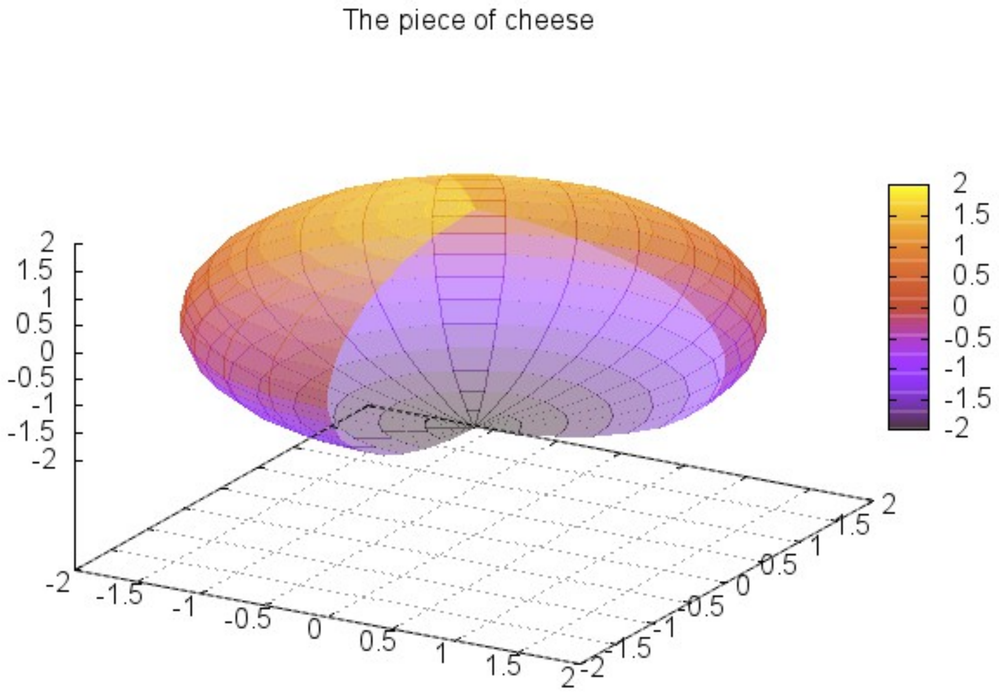
- animated plots
- vector plots
- surface plots



Introduction

Examples:

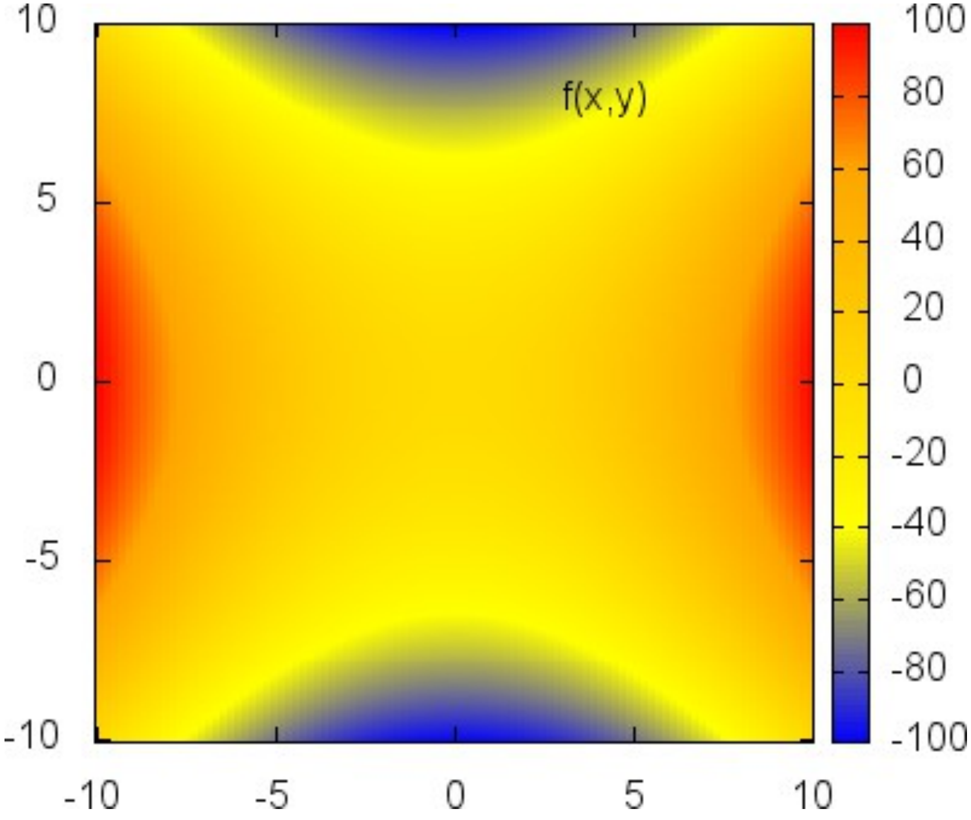
- animated plots
- vector plots
- surface plots
- parametric plots



The piece of cheese

Examples:

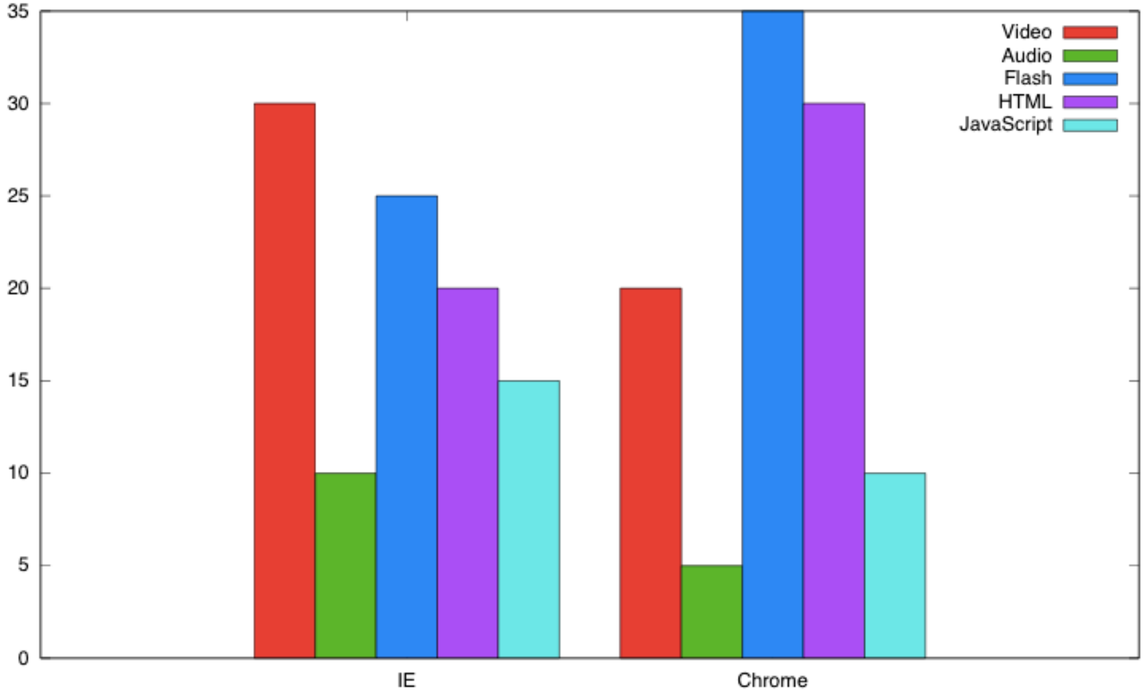
- animated plots
- vector plots
- surface plots
- parametric plots
- contour plots



Introduction

Examples:

- animated plots
- vector plots
- surface plots
- parametric plots
- contour plots
- bardiagrams
- …

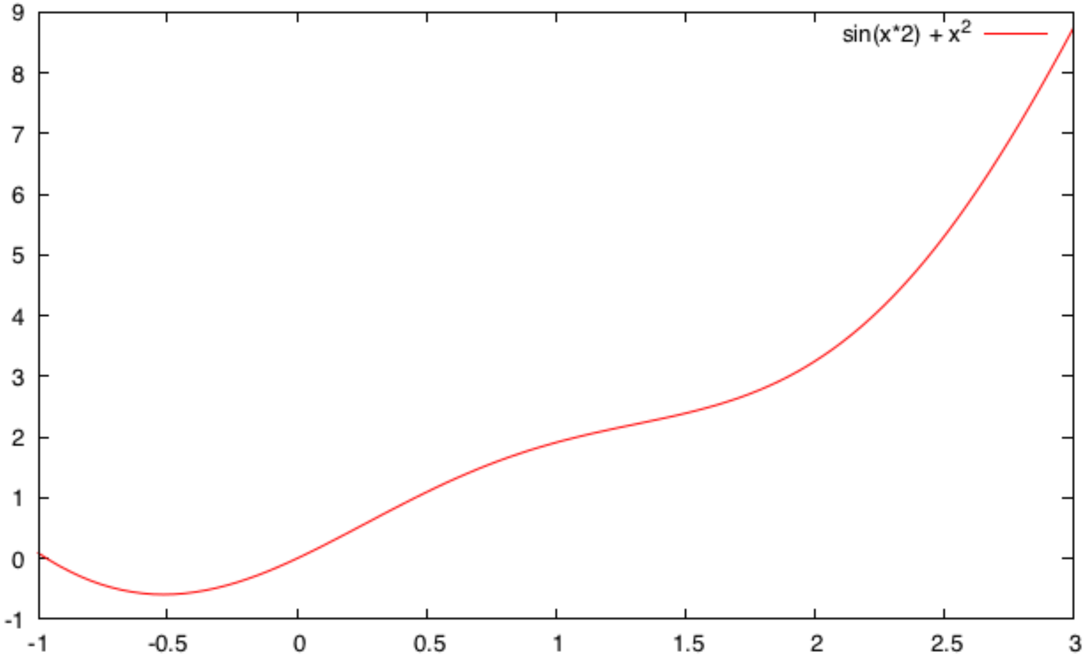


Basics

1.2 Basic Commands:

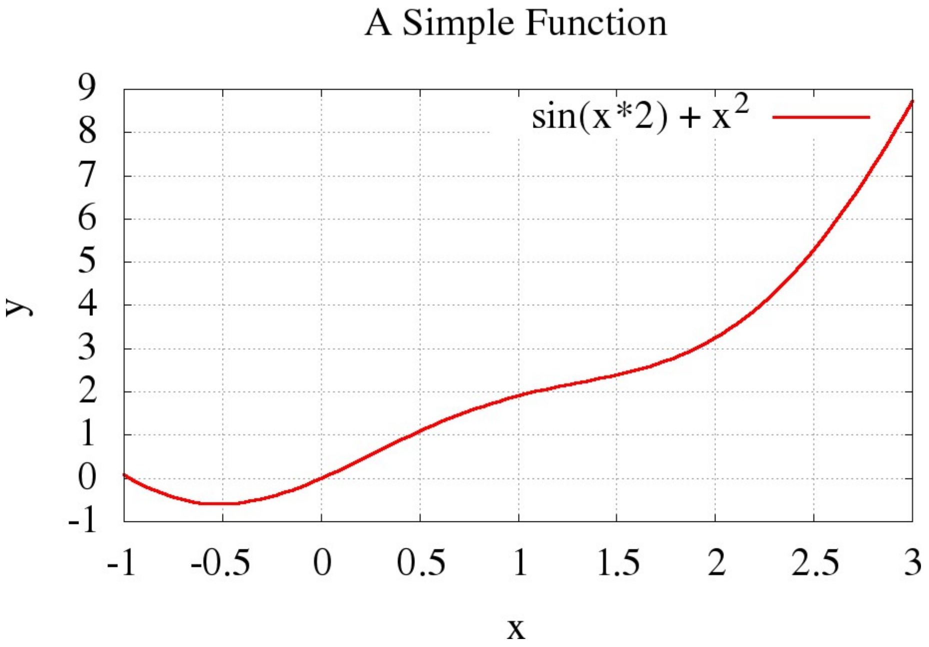
Example:

```
$> gnuplot
gnuplot> plot [-1:3][]  sin(2*x) + x**2 with lines title "sin(2*x) + x^2"
```



1.2 Basic Commands:

Example (pretty up):



1. reset
2. set title "A Simple Function"
3. set xlabel "x"
4. set ylabel "y"
5. set grid
6. plot [-1:3][] sin(2*x) + x**2 with lines lw 3 title "sin(x*2) + x^2"

Basics

Basic Commands:

- *set <object> <option/parameter/value>*
- define settings for an object
- set xlabel "x"

- unset *<object>*
- sets object to standard setting
- unset xlabel

- reset
- sets everything back to standard setting

Basics

Basic Commands:

- plot [*xrange*][*yrange*] {*fct / "File"*} axes {*x1y1*} title ""
- plots the function 2D

- replot
- repeat the previous plot command

- splot [*xrange*][*yrange*][*zrange*] {*fct / "File"*} title ""
- Surface plots (3D)

Basics

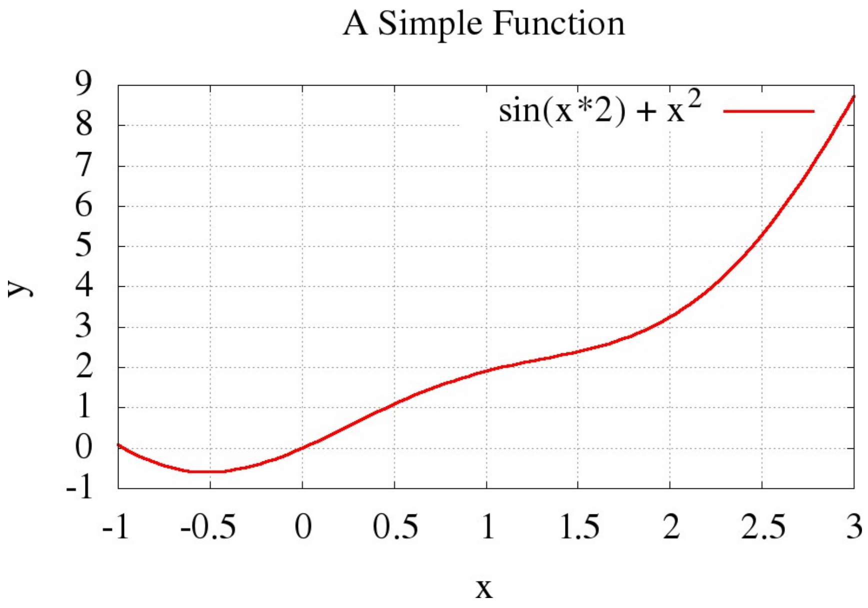
Example (recap):

```
reset
# make global definitions
set terminal jpeg enhanced
set output "1_Sin.jpeg"

set title "A Simple Function"
set xlabel "x"
set ylabel "y"

set grid

plot [-1:3][]  sin(2*x) + x**2 with lines lw 3 title
"sin(x*2) + x^2"
```



Basics

define the output:

- set terminal <*format*> <*options*>
- set output "filename"

selection of Output formats <format>

x11, wxt (Linux); windows (MS)	screen
postscript	*.ps, *.eps
latex, epslatex	*.tex , and *.eps
svg	Scalable Vector Graphics
jpeg, gif, png	raster graphics image

Basics

define the output:

- set terminal <*format*> <*options*>
- set output "filename"

selection of Output formats options <options>	
enhanced	improved scripts
crop	fill whole area, no outer space
size <*int pixel width*> <*int pixel hight*>	resolution (for jpeg, png, gif)
size <*int cm/inch*> <*int cm/inch*>	resolution (for vector graphics)
font "type" <*letter size*>	sets global font option, e.g. font "Times" 30
color	Color (otherwise black-white in many formats)

Basics

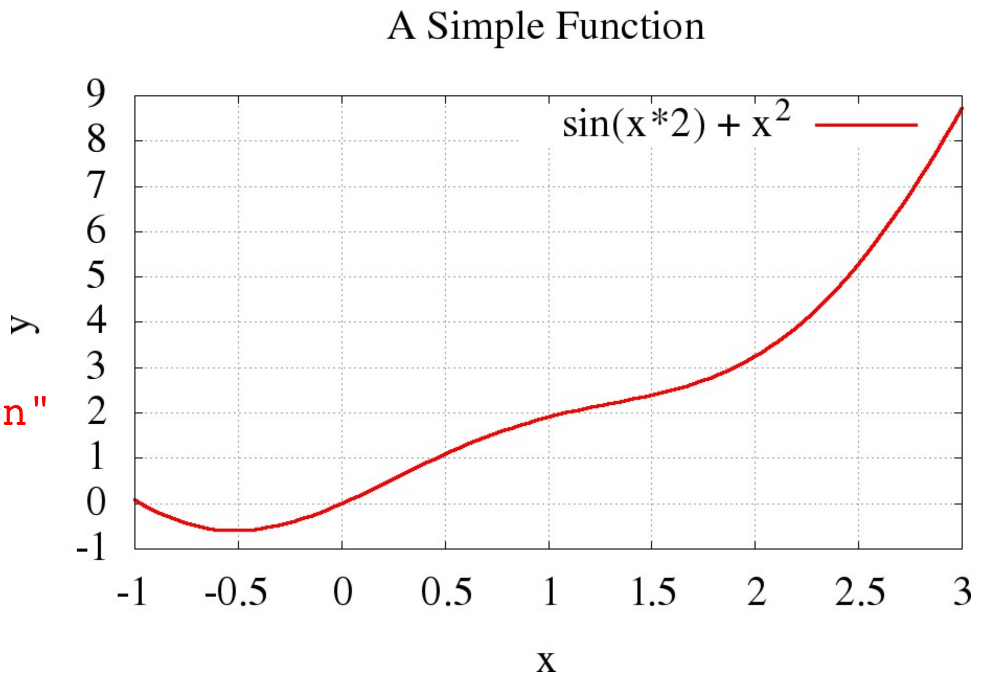
Example (recap):

```
reset
# make global definitions
set terminal jpeg enhanced
set output "1_Sin.jpeg"

set title "A Simple Function"
set xlabel "x"
set ylabel "y"

set grid

plot [-1:3][]  sin(2*x) + x**2 with lines lw 3 title
"sin({x}*{2}) + x^2"
```



Basics

A (small !) selection of layout definitions

1. set title “text” font “fontname” <*fontsize*> <*color*>

2. set _label “text” font “fontname*”* <*fontsize*> “color” rotate by <*int:degrees*>

3. set grid [no]_tics [no]m_tics linewidth <*float linewidth*>

Basics

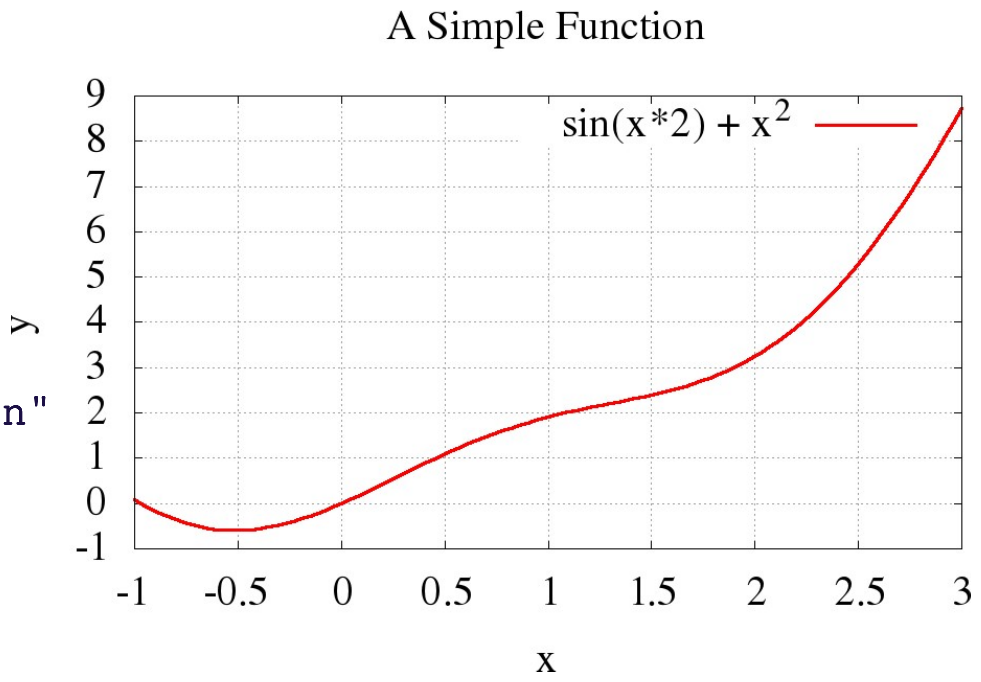
Example (recap):

```
reset
# make global definitions
set terminal jpeg enhanced
set output "1_Sin.jpeg"

set title "A Simple Function"
set xlabel "x"
set ylabel "y"

set grid

plot [-1:3][]  sin(2*x) + x**2 with lines lw 3 title
"sin({x}*{2}) + x^2"
```



Basics

Basic Commands:

- # comment the string behind in the line
- selection of operators:

Operators (numbers)	
**	Exponentation (see Fortran, Python)
%	Modulo
<, <=, ==	less, less equal, equal

Operators	
!	Logical not
a ? b : c	Ternary (see C)
eq	Equality (strings)

Basics

Basic Commands:

- implemented Functions (selection):

Mathmatical Functions	
exp	exponential function
log, log10	logarithm (basis e, 10)
sin, cos, asinh	trigonometic functions
erf, erfc	error fct, complementary ~
abs	absolute value

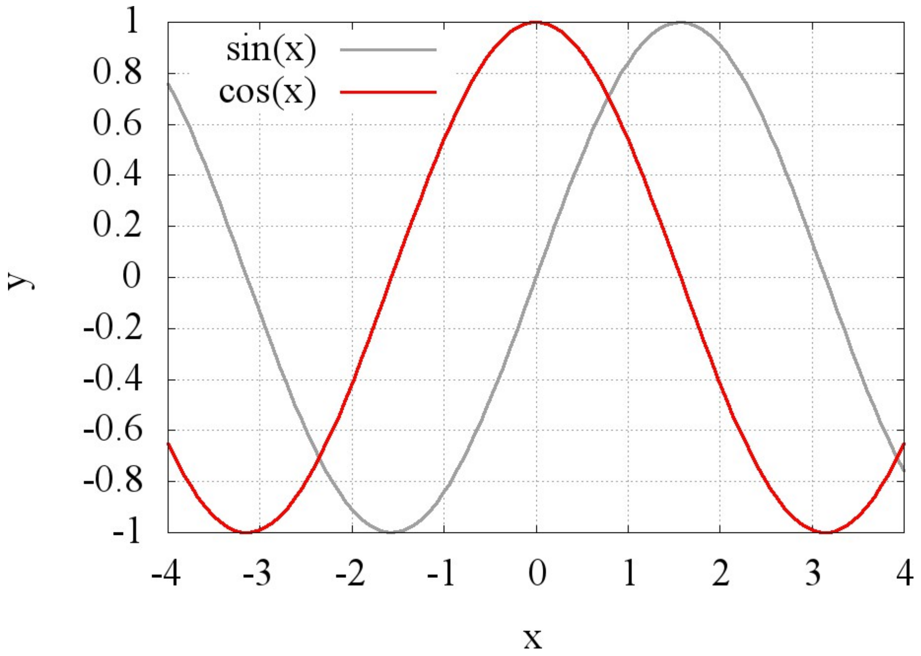
Program Functions	
sprintf	return formatted str, (see C)
system	executes input str in subshell

1.3 Local – Global Definitions:

Example global definitions:

```
set style line 1 linewidth 3 linecolor 0
set style line 2 lw 3        lc 1
f(x) = sin(x)
g(x) = cos(x)
title1 = "sin(x)"
title2 = "cos(x)"
set xrange[-4:4]

plot f(x) with lines linestyle 1 title title1, \
     g(x) w l        ls 2        t title2
```



Global Definitions:

- set <…> valid until:
  - reset
  - unset <…>
  - set <…> # re-definition

Local – Global Definitions:

Example local definitions:

```
plot [-4:4] sin(x) with lines linewidth 3 linecolor 0 title "sin(x)" ,\
            cos(x) w l        lw 3        lc 1        t  "cos(x)"
```

Overview

1. Basics
    1. gnuplot
    2. Basic Commands
    3. Local – Global Definitions

2. Plotting Data from Files
    1. File Structure
    2. Plot Settings

3. Layout
    1. Axis, grid
    2. Multiplots
    3. Linestyles

Plotting Data from Files

2.1 Structure of data files:

- # Comments, not considered by gnuplot
  - set datafile commentschar “!”
- Files structured *columnwise*:
  - selectable by using col1:col2:…
  - space separated
    - set datafile separator “,”
  - index starting at 1

- *single* blank line:
  - discontinuity in data
  - line keeps style but not connected

Plotting Data from Files

- *double* blank line:
  - different sets of data
  - selectable by index
    - starts at 0,
      so index 2 selects the third set in a file
    - further possible parameters:
      index <*int start*> : <*int end*> : <*int step*>

- every allows to select lines interval (e.g. if data is mixed)

Plotting Data from Files

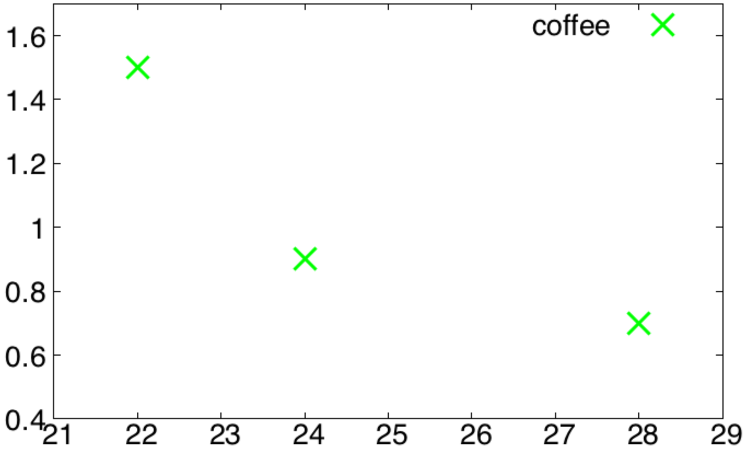
- using allows to access to columns, e.g. using 2:4
- calculate with columns in files:
  - () implies values, e.g. plots using 1:(2.1) a constant value of y=2.1 over x
  - using 1:(2*$3) plots the doubled value of column 3 over column 1
  - special columns indicies:
    - 0 is the line number in file
    - -2 the index

Plotting Data from Files

```
$> cat food.dat
12:22   22     19    #Number of Cookies
12:22   22     1.5   #liters of coffee
12:22   22     3.0   #liters of water


12:24   24     14    #Number of Cookies
12:24   24     0.9   #liters of coffee
12:24   24     2.8   #liters of water

…
$>gnuplot
gnuplot> plot “food.dat" index 0:4:1 every 4::1 u 2:3 ls 2 t
"coffee”
```



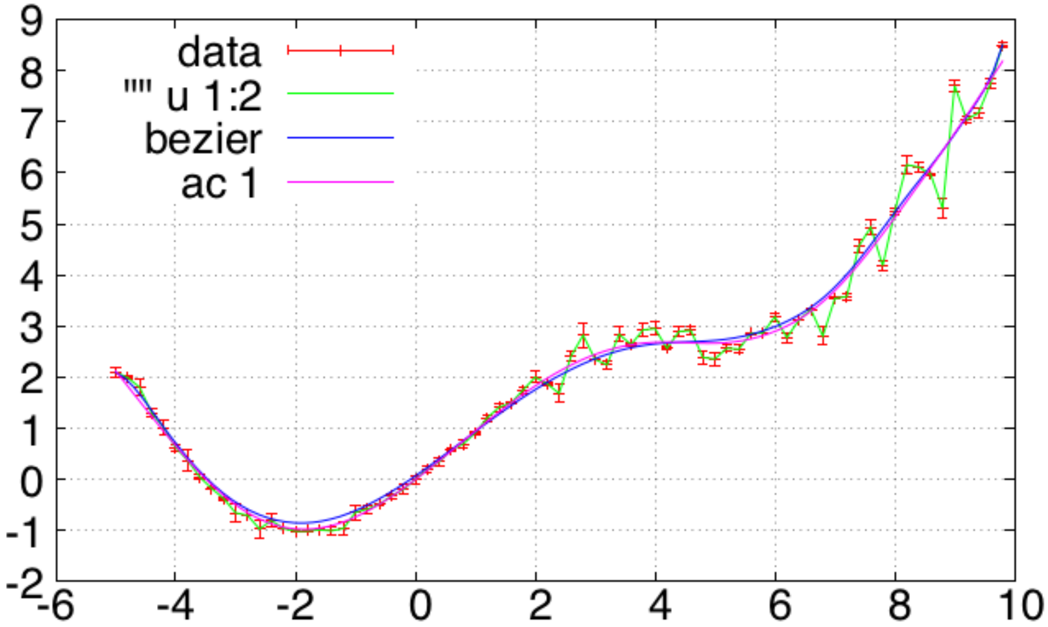
Plotting Data from Files

2.2 options for data plotting:

- smooth
  - unique
    - sorts the elements in a column
    - points are directly linear connected
  - bezier
    - Bézier curve through all points
  - acsplines
    - weighted cubic spline approximation
    - weights defined in third column
- $>cat norm.dat
- x   y   error
- -5.0000000e+00   2.0922396e+00 -8.5687486e-02
- -4.8000000e+00   1.9912158e+00   3.9218173e-02
- .....

Plotting Data from Files

```
$> gnuplot
gnuplot> plot "norm.dat" using 1:2:3 w yerrorbars t "data" ,\
>                  "" u 1:2 w l,\
>            "" u 1:2 smooth bezier t "bezier" ,\
>            "" u 1:2:(1) s acs t "ac 1"
```



Plotting Data from Files

options for data input:

- fit <fct> “datafile” {index …} {every …} {using …} via param1,…
  - fits data file using least square

- use file data headers as title:
  - plot for [col=2:3] ’norm.dat’ u 1:col with lines title columnheader
  - plot “norm.dat” u (column(' x')):(column(' y')) w l

Plotting Data from Files

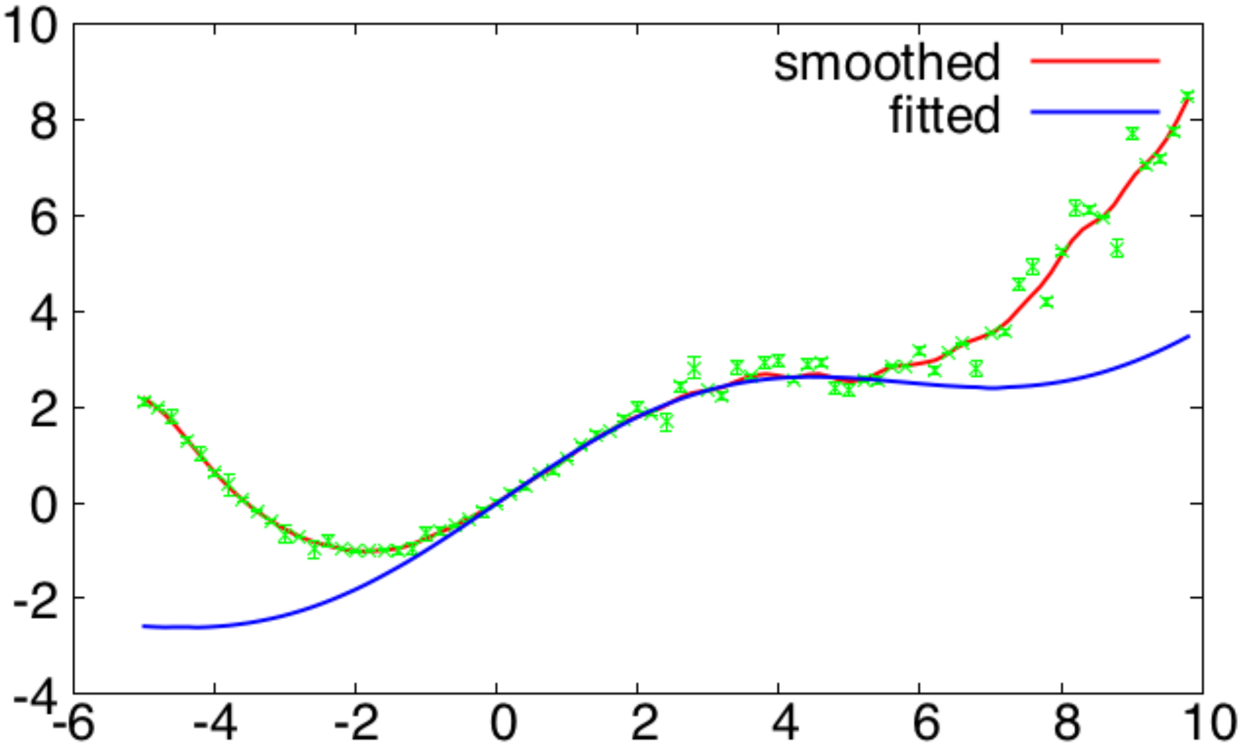
```
$>cat fit.gplt
# fitting function
fitfct(x) = p1*cos(p2*x + p3) + p4*x

# initial guess
p1 = 1.
p2 = 0.5
p3 = pi/4.
p4 = 0.2

fit [0:5][] fitfct(x) "norm.dat" u 1:2:3 via p1,p2,p3,p4    # third col treated as std deviation
plot "norm.dat" u 1:2:(1./column("error")**2) smooth acsplines t "smoothed" ,\
        "" u "x":(column("y")):3 w yerrorbars notitle ,\
        fitfct(x) t "fitted"
```

Plotting Data from Files

```
$> gnuplot
gnuplot> load ‘fit.gplt’
```



Overview

1. Basics
   1. gnuplot
   2. Basic Commands
   3. Local – Global Definitions

2. Plotting Data from Files
   1. File Structure
   2. Plot Settings

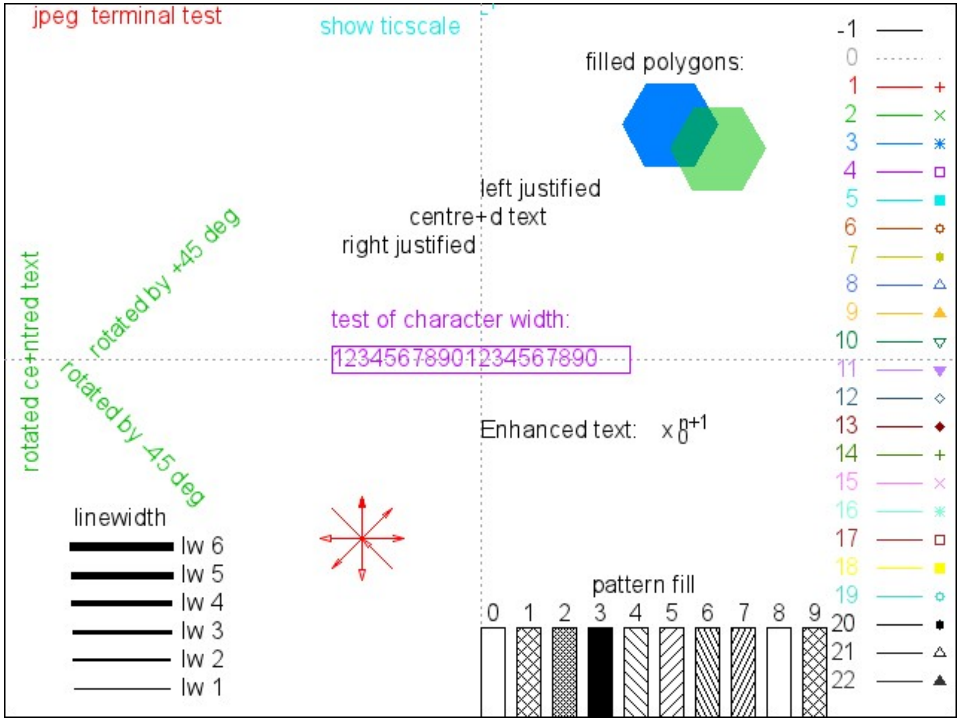
3. Layout
   1. Axis, grid
   2. Multiplots
   3. Linestyles

Layout

TECHNISCHE
UNIVERSITÄT
DARMSTADT

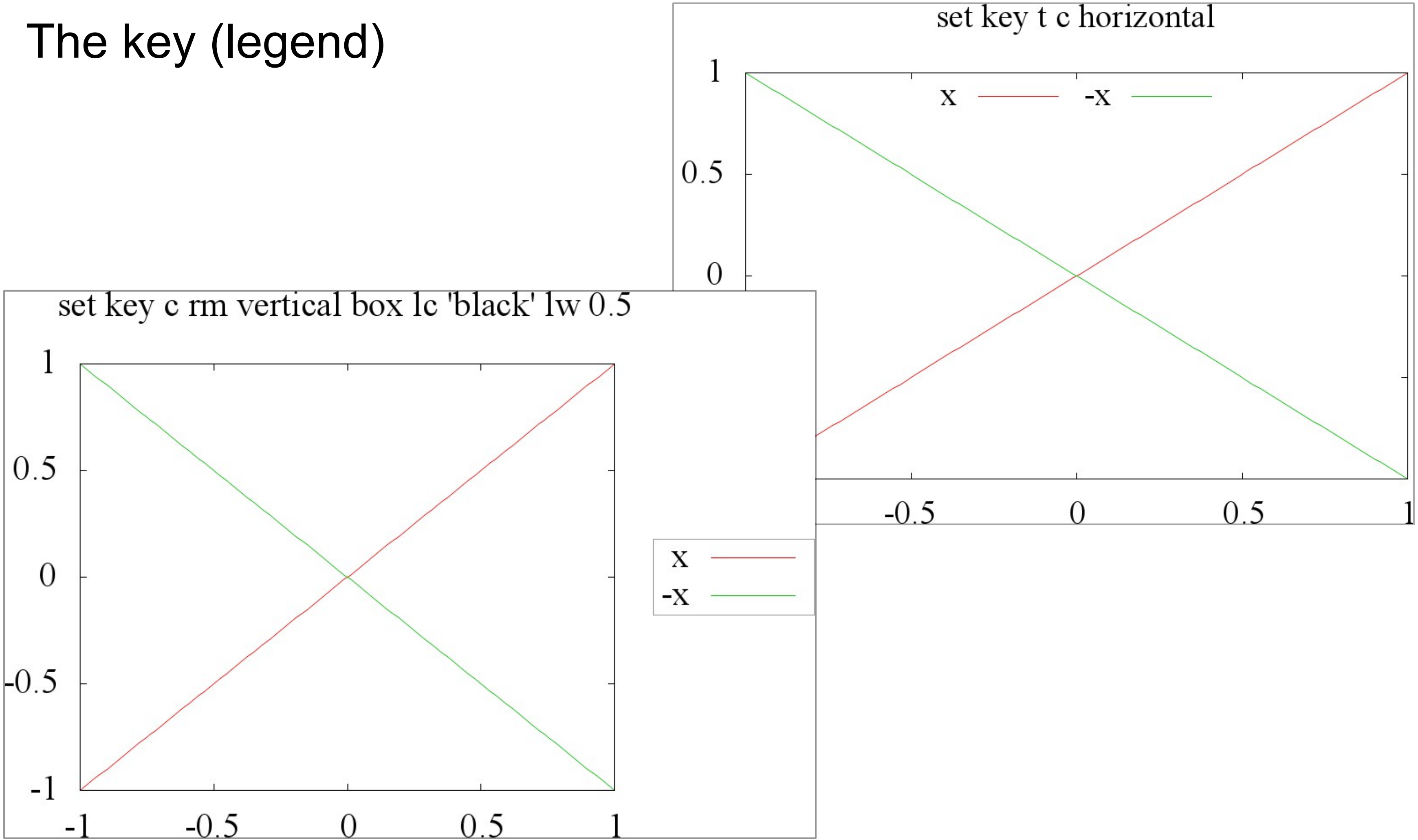
1. Layout

- General:
  - output style depends on terminal
  - command *test* in gnuplot outputs overview of layout options:



Layout

The key (legend)



```
set key t c horizontal
```

```
set key c rm vertical box lc 'black' lw 0.5
```

Layout

3.1 Axis

- logarithmic axes:

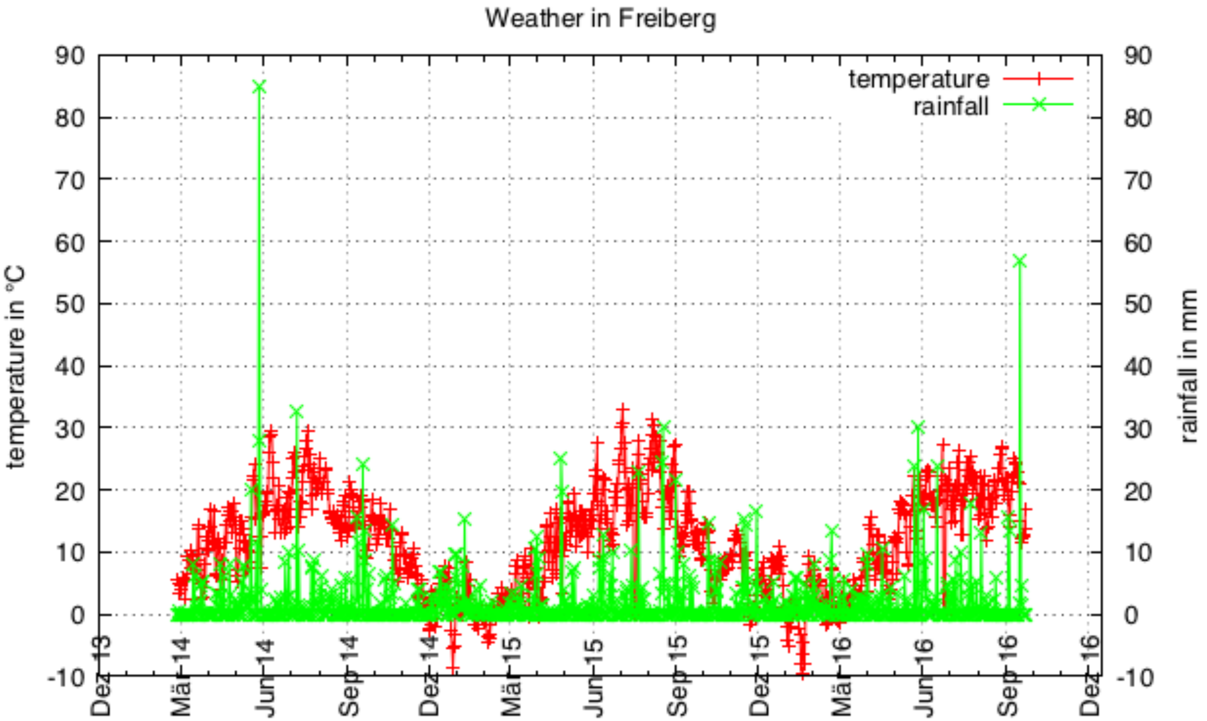
  ```
  set logscale     # double logarithmic plots
  set logscale x   # only x axis
  set logscale y   # only y axis
  ```

- for multiple axes

  ```
  plot "" u 1:2 axes x1y1
  ```

Layout

```
$> gnuplot
gnuplot> load “WeatherTrendFG.gplt”
```



source of data: *http://www.chemie.tu-freiberg.de/wetterdaten/ddaten_anzeige.php*

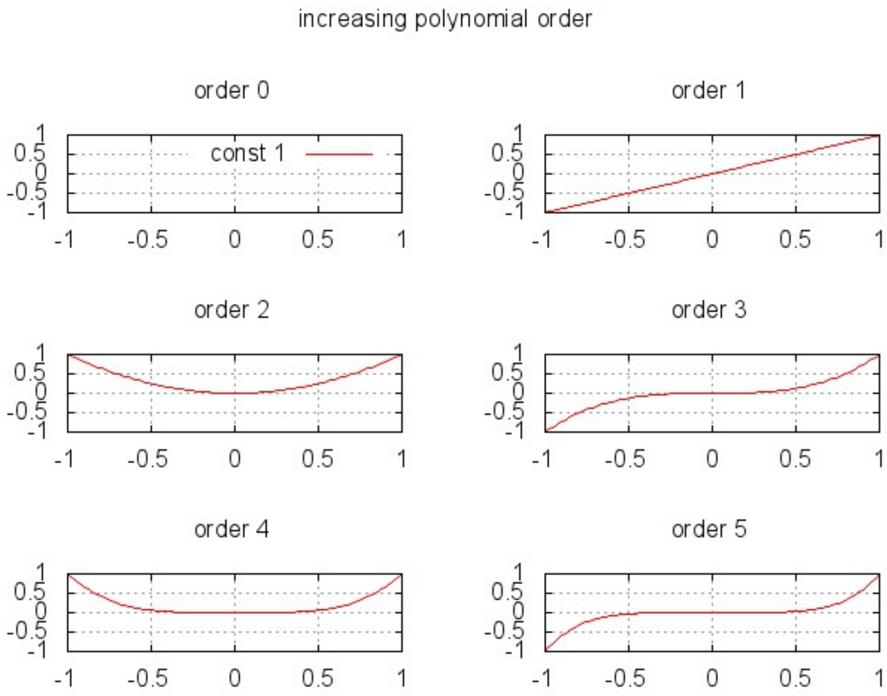
Layout

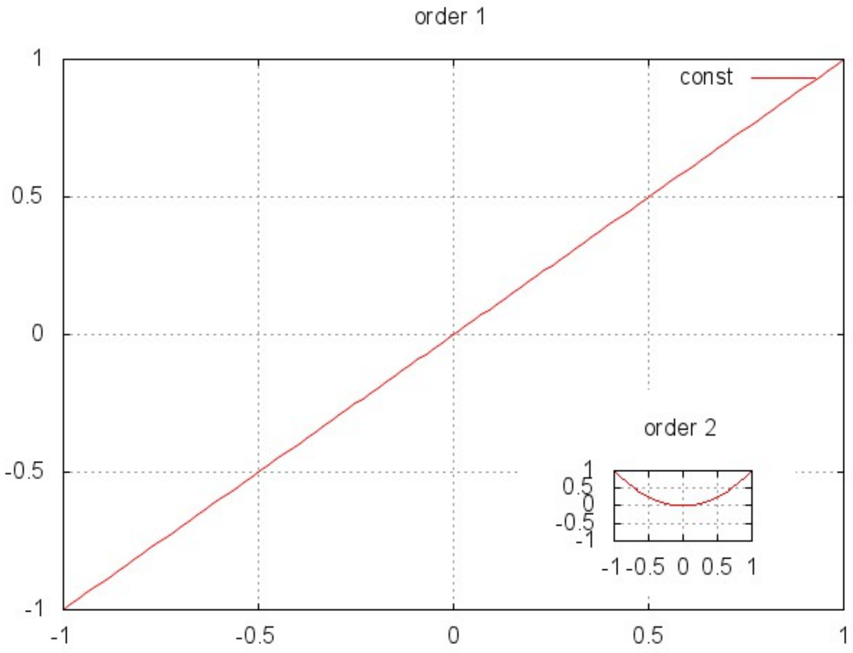
3.2 multiplots

- if many coordinate systems
  - two main options:

(1) array

(2) insets





Layout

3.3.Linestyles

plot with <type> <linestyle> <linewidth> <linecolor>

Types

- Lines
- Points
- Linepoints
- Impulses
- boxes

Links and Literature

- gnuplot homepage: (see *demos*)
  - http://gnuplot.sourceforge.net/

- very detailed documentation:
  gnuplot> help [command]

- *Gnuplot in Action, Understanding Data with Graphs*; Philipp K. Janert; Manning Publications; 2009

- list with commands and corresponding examples:
  http://www.chemie.fu-berlin.de/chemnet/use/info/gnuplot/gnuplot_toc.html

- many (advanced) examples:
  http://gnuplot-tricks.blogspot.de/