
Basics of Linux II

The Linux Command Line Interface

Some examples about the target knowledge

- `for i in $(seq -f "%02g" 1 40); do ii=$(echo "43.00+($i-1)*0.5" | bc -l) ; cp -r sampletest test-$i; sed -i 's/#2#/'$ii'/g' test-$i/check; sed -i 's/#2#/'$ii'/g' test-$i/killer; sed -i 's/#1#/'$i'/g' test-$i/check; sed -i 's/#1#/'$i'/g' test-$i/job-submit; done;`
- `cd /home2; for user in *; do quota 2>&1 -wp -u $user | awk -v user=$user '{if(NR == 3) {print $6,user}}' ; done | sort -n -r > ~/home2-number-of-files.txt`
- `for i in $(seq 70 77); do if ! ping -c1 10.100.163.$i 1>/dev/null ; then echo "NucleusC$i IB is NOT OK"; fi; done`
- `for i in $(seq -w 010 195); do ssh Nucleus$i 'hname=`/bin/hostname`; core=`lscpu | grep ^CPU\(| cut -f2- -d" "`; ht=`lscpu | grep ^Thread| cut -f4- -d" "`; mem=`free -g | grep ^Mem | cut -f2- -d" "`; echo $hname $mem $core $ht'; done > node-information.txt`
- `awk '{print $4/3600, $15}' 2020_01_15_1901.log | grep -v "e-05" | sort -n -r | head -10 | sed 's/\/home2\/\\\\/g' | gnuplot -p -e 'set terminal png size 1200,800; set output "~/home2-backup-time.png"; set boxwidth 0.5; set style fill solid; plot "/dev/stdin" using 1:xtic(2) with boxes'`

Shell commands and tools:

- bash scripting
 - Commands
 - Outputs
 - errors
 - forwarding
 - numerical operations
 - loops
 - conditional tools
- Shell tools
 - grep
 - sed
 - awk
 - gnuplot

Forwarding, Arrays, Numerical operations

```
python hello.py > output.txt # stdout to (file)
python hello.py >> output.txt # stdout to (file), append
python hello.py 2> error.log # stderr to (file)
python hello.py 2>&1 # stderr to stdout
python hello.py 2>/dev/null # stderr to (null)
python hello.py &>/dev/null # stdout and stderr to (null)

python hello.py < foo.txt
```

```
Fruits=('Apple' 'Banana' 'Orange')

Fruits[0]="Apple"
Fruits[1]="Banana"
Fruits[2]="Orange"

echo ${Fruits[0]} # Element #0
echo ${Fruits[@]} # All elements, space-separated
echo ${#Fruits[@]} # Number of elements
echo ${#Fruits} # String length of the 1st element
echo ${#Fruits[3]} # String length of the Nth element
echo ${Fruits[@]:3:2} # Range (from position 3, length 2)
```

```
a=20
var1=$((a + 200))
echo $var1 220

echo $(( 20 / 5 )) 4
echo $(( 20 * 5 )) 100
echo $(( 20 % 3 )) 2

x=5
echo $(( x++ )) 5
echo $(( x++ )) 6
echo $(( x-- )) 7
echo $(( x-- )) 6

x=2
y=3
echo $(( x ** y )) 8
```

Loops

Basic for loop

```
for i in /etc/rc.*; do
  echo $i
done
```

Ranges

```
for i in {1..5}; do
  echo "Welcome $i"
done
```

With step size

```
for i in {5..50..5}; do
  echo "Welcome $i"
done
```

Reading lines

```
cat file.txt | while read line; do
  echo $line
done
```

Forever

```
while true; do
  ...
done
```

```
for i in $(seq -w 001 081); do echo "Nucleus$i"; done
```

Conditional Tools

Conditions

```
[ -z STRING ]      Empty string
[ -n STRING ]      Not empty string
[ NUM -eq NUM ]    Equal
[ NUM -ne NUM ]    Not equal
[ NUM -lt NUM ]    Less than
[ NUM -le NUM ]    Less than or equal
[ NUM -gt NUM ]    Greater than
[ NUM -ge NUM ]    Greater than or equal
[[ STRING =~ STRING ]] Regexp
(( NUM < NUM ))    Numeric conditions
[ -o noclobber ]   If OPTIONNAME is enabled
[ ! EXPR ]         Not
[ X ] && [ Y ]      And
[ X ] || [ Y ]     Or
```

File conditions

```
[ -e FILE ]        Exists
[ -r FILE ]        Readable
[ -h FILE ]        Symlink
[ -d FILE ]        Directory
[ -w FILE ]        Writable
[ -s FILE ]        Size is > 0 bytes
[ -f FILE ]        File
[ -x FILE ]        Executable
[ FILE1 -nt FILE2 ] 1 is more recent than 2
[ FILE1 -ot FILE2 ] 2 is more recent than 1
[ FILE1 -ef FILE2 ] Same files
```

```
#!/bin/bash
if [ -r $1 ]; then
    content=$(cat $1)
elif [ -f $1 ]; then
    echo "$1 not readable"
else
    echo "$1 doesn't exist"
fi
```

Example

```
# String
if [ -z "$string" ]; then
    echo "String is empty"
elif [ -n "$string" ]; then
    echo "String is not empty"
fi

# Combinations
if [ X ] && [ Y ]; then
    ...
fi

# Regexp
if [[ "A" =~ "." ]]

if (( $a < $b ))

if [ -e "file.txt" ]; then
    echo "file exists"
fi
```

Sed is a stream editor.

▪ Syntax:

▪ sed OPTIONS... [SCRIPT] [INPUTFILE...]

- sed 's/unix/linux/' geekfile.txt # change first occurrence
- sed 's/unix/linux/2' geekfile.txt # change second occurrence
- sed 's/unix/linux/g' geekfile.txt # change all occurrence
- sed '3 s/unix/linux/' geekfile.txt # for line 3
- sed 's/unix/linux/p' geekfile.txt # Duplicating the replaced line
- sed -n 's/unix/linux/p' geekfile.txt # Printing only the replaced lines
- sed '1,3 s/unix/linux/' geekfile.txt # Replacing string on a range of lines
- sed '2,\$ s/unix/linux/' geekfile.txt # 2 to end
- sed '5d' filename.txt # Deleting lines from file
- sed '\$d' filename.txt # to delete last line
- sed '3,6d' filename.txt # delete 3-6 lines
- sed '/pattern/d' filename.txt # To Delete pattern matching line

▪ echo "Welcome To The Geek Stuff" | sed 's^\(b[A-Z]\)\(1\)/g' # (W)elcome (T)o (T)he (G)eek (S)tuff

```
unix is great os. unix is opensource. unix is free os.
learn operating system. u
learn operating system. un
learn operating system. uni
learn operating system. unix
unix linux which one you choose.
unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
```

grep

- `grep "linux" index.html` # Search for a string
- `grep -i "linux" index.html` # Insensitive case search
- `grep "linux" file*.*` # Searching multiple files
- `grep "fast.*host" index.html` # Specifying the search string
- `grep -n "word*" file.txt` # Displaying the line numbers
- `grep -color "linux" index.html` # Highlighting the search
- `grep -v linux /etc/passwd` # Print the line excluding the pattern
- `grep ^root /etc/passwd` # lines that starts with specified pattern
- `grep bash$ /etc/passwd` # lines that ends with specified pattern
- `grep -r linux /etc/` # Search the pattern recursively
- `grep -c 'test' /home/example/test.txt` # Counting the lines when words match

Awk

AWK is a [programming language](#) designed for text processing and typically used as a [data extraction](#) and reporting tool.

Basic Syntax : `awk '/search_pattern/ { action_to_take_on_matches; another_action; }' file_to_parse`

- `awk '{print}' /etc/fstab`
- `awk '/UUID/' /etc/fstab`
- `awk '/^UUID/' /etc/fstab`
- `awk '/^UUID/ {print $1;}' /etc/fstab`

- `awk '/sa/' favorite_food.txt` # print lines includes sa in anywhere
- `awk '$2 ~ /^sa/' favorite_food.txt` # print lines if its second column starts with sa
- `awk '$2 !~ /^sa/' favorite_food.txt` # print lines if its second column not starts with sa

- awk script files:
`awk 'BEGIN { action; }
/search/ { action; }
END { action; }' input_file`
- `awk 'BEGIN { FS=":"; print "User\t\tUID\t\tGID\t\tHome\t\tShell\n-----"; } {print $1,"\t\t",$3,"\t\t",$4,"\t\t",$6,"\t\t",$7;} END { print "-----\nFile Complete" }' /etc/passwd`

```
# cat favorite_food.txt
```

```
1 carrot sandy
2 wasabi luke
3 sandwich brian
4 salad ryan
5 spaghetti jessica
```

1) <https://www.digitalocean.com/community/tutorials/how-to-use-the-awk-language-to-manipulate-text-in-linux>

```
# /etc/fstab: static file system information.
```

```
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
proc /proc proc nodev,noexec,nosuid 0 0
# / was on /dev/vda1 during installation
UUID=b96601ba-7d51-4c5f-bfe2-63815708aabd / ext4
noatime,errors=remount-ro 0 1
```

Awk Built-in Variables – FS, OFS, RS, ORS, NR, NF, FILENAME, FNR

FS = input separator, use `-F` in command line

Syntax:

```
$ awk -F 'FS' 'commands' inputfilename
```

(or)

```
$ awk 'BEGIN{FS="FS"}'
```

OFS = output separator, use `-F` in command line

Syntax:

```
$ awk 'BEGIN{OFS=":"}'
```

RS = input Record separator (default line by line)

Syntax:

```
$ awk 'BEGIN{RS="\n\n"}'
```

NR = Line number at the time of processing

NF = Number of column at the line of processing

FNR = Line number for the current file

FILENAME = the name of the file being read

```
$ awk '{print FILENAME, FNR;}' student-marks bookdetails
```

```
student-marks 1
```

```
student-marks 2
```

```
student-marks 3
```

```
student-marks 4
```

```
student-marks 5
```

```
bookdetails 1
```

```
bookdetails 2
```

```
bookdetails 3
```

```
bookdetails 4
```

```
bookdetails 5
```

```
$ cat student.txt
```

```
Jones
```

```
2143
```

```
78
```

```
84
```

```
77
```

```
Gondrol
```

```
2321
```

```
56
```

```
58
```

```
45
```

```
RinRao
```

```
2122
```

```
38
```

```
37
```

```
65
```

```
Edwin
```

```
2537
```

```
78
```

```
67
```

```
45
```

```
Dayan
```

```
2415
```

```
30
```

```
47
```

```
20
```

```
$ cat student.awk
```

```
BEGIN {
```

```
    RS="\n\n";
```

```
    FS="\n";
```

```
}
```

```
{
```

```
    print $1,$2;
```

```
}
```

```
$ awk -f student.awk student.txt > student.mark
```

```
Jones 2143
```

```
Gondrol 2321
```

```
RinRao 2122
```

```
Edwin 2537
```

```
Dayan 2415
```

```
$ awk 'BEGIN{FS="\n";RS="\n\n";ORS="=";} {print $1,$2;}' student.txt
```

```
Jones 2143=Gondrol 2321=RinRao 2122=Edwin 2537=Dayan 2415=
```

```
$ awk '{print FILENAME, NR, FNR;}' student.mark
```

```
student.txt
```

```
student.mark 1 1
```

```
student.mark 2 2
```

```
student.mark 3 3
```

```
student.mark 4 4
```

```
student.mark 5 5
```

```
student.txt 6 1
```

```
student.txt 7 2
```

```
student.txt 8 3
```

Example: script for awk

```
$ ls -l /usr/bin | head -7
total 436832
-rwxr-xr-x 1 root root 41544 Mar 15 2019 [
-rwxr-xr-x 1 root root 107904 Jan 7 2019 a2p
-rwxr-xr-x 1 root root 11336 Mar 20 2019 abrt-action-analyze-backtrace
-rwxr-xr-x 1 root root 11320 Mar 20 2019 abrt-action-analyze-c
-rwxr-xr-x 1 root root 1345 Mar 20 2019 abrt-action-analyze-ccpp-local
-rwxr-xr-x 1 root root 6821 Mar 20 2019 abrt-action-analyze-core
```

```
ls -l /usr/bin | awk '
BEGIN {
    print "Directory Report"
    print "======"
}
NF > 9 {
    print $9, "is a symbolic link to", $NF
}
END {
    print "======"
    print "End Of Report"
}'
```

```
Directory Report
=====
apropos is a symbolic link to whatis
arecord is a symbolic link to aplay
atq is a symbolic link to at
...
ypdomainname is a symbolic link to hostname
zsoelim is a symbolic link to soelim
=====
End Of Report
```

```
$ ls -l /usr/bin | awk '{print $1}' | head -5
total
-rwxr-xr-x
-rwxr-xr-x
-rwxr-xr-x
-rwxr-xr-x
```

```
$ ls -l /usr/bin | awk '{print $9}' | head -5
[
a2p
abrt-action-analyze-backtrace
abrt-action-analyze-c
```

```
$ ls -l /usr/bin | awk 'NF > 9 {print $0}' | head -5
lrwxrwxrwx 1 root root 6 Feb 24 2020 apropos -> whatis
lrwxrwxrwx 1 root root 5 Feb 24 2020 arecord -> aplay
lrwxrwxrwx 1 root root 2 Feb 24 2020 atq -> at
lrwxrwxrwx 1 root root 2 Feb 24 2020 atrm -> at
lrwxrwxrwx 1 root root 11 Feb 24 2020 audit2why -> audit2allow
```

```
$ ls -l /usr/bin | awk 'BEGIN {OFS = ","} NF == 9 {print
$1,$2,$3,$4,$5,$6,$7,$8,$9}' | head -5
-rwxr-xr-x,1,root,root,41544,Mar,15,2019,[
-rwxr-xr-x,1,root,root,107904,Jan,7,2019,a2p
-rwxr-xr-x,1,root,root,11336,Mar,20,2019,abrt-action-analyze-backtrace
-rwxr-xr-x,1,root,root,11320,Mar,20,2019,abrt-action-analyze-c
-rwxr-xr-x,1,root,root,1345,Mar,20,2019,abrt-action-analyze-ccpp-local
```

```
$ ls -l /usr/bin | awk 'BEGIN {ORS = "\n\n"} {print}' | head -10
total 436832

-rwxr-xr-x 1 root root 41544 Mar 15 2019 [

-rwxr-xr-x 1 root root 107904 Jan 7 2019 a2p

-rwxr-xr-x 1 root root 11336 Mar 20 2019 abrt-action-analyze-backtrace

-rwxr-xr-x 1 root root 11320 Mar 20 2019 abrt-action-analyze-c
```

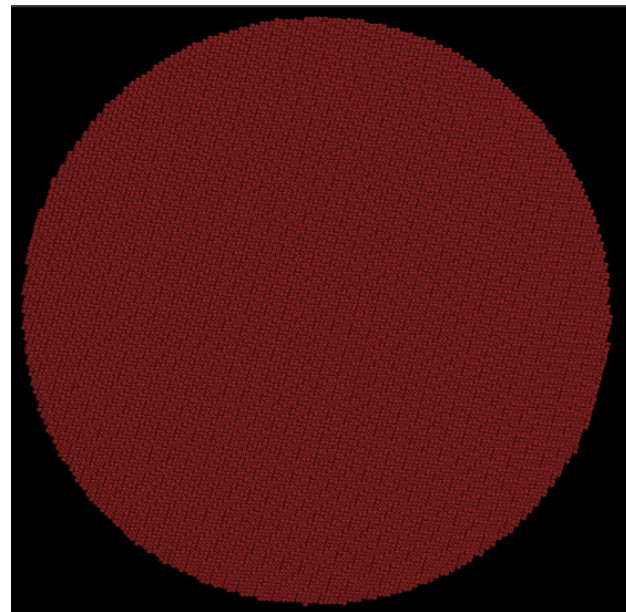
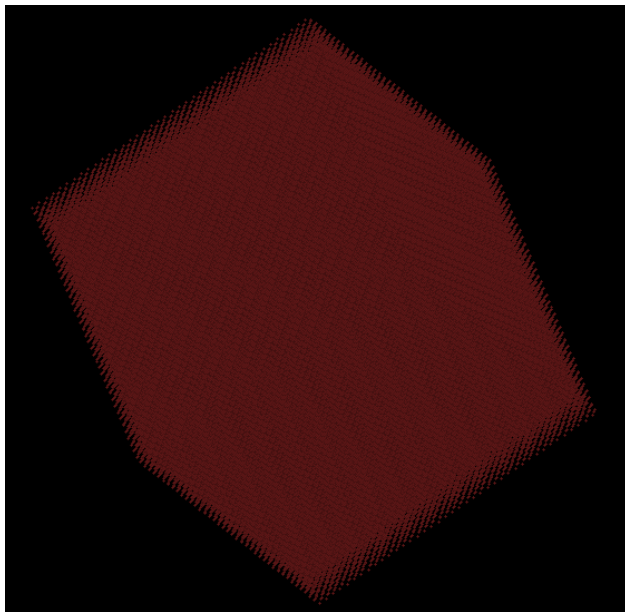
Example: script for awk

```
#!/bin/awk -f
awk '
BEGIN {
  while (getline > 0) {
    i++;
    el[i]=$1; x[i]=$2; y[i]=$3; z[i]=$4;
    if(NR == 1) {
      mx=$2; my=$3; mz=$4
    }
    if($2>mx) {mx=$2}
    if($2>my) {mx=$2}
    if($2>mz) {mx=$2}
    }
    str=i
  }
END {
  for (i=1; i <= str; i++) {
    print el[i], x[i]+0.0, y[i]+26.0, z[i]+0.0
    print el[i], x[i]-0.0, y[i]-26.0, z[i]+0.0
  }
}'
```

```
# Check for file space on /tmp
space=`df | grep "/tmp" | awk '{ print $5+0.0; }'`
echo "Space on /tmp:" $space "%"
if test $space -ge 80
then
  echo "slurm script check: File space on tmp on node" `hostname`
  echo "is dangerously low. Aborting job."
  echo "Please ask an administrator to clean up /tmp on that node"
  echo `df | grep "/tmp"`
  exit
fi
```

Example: script for awk

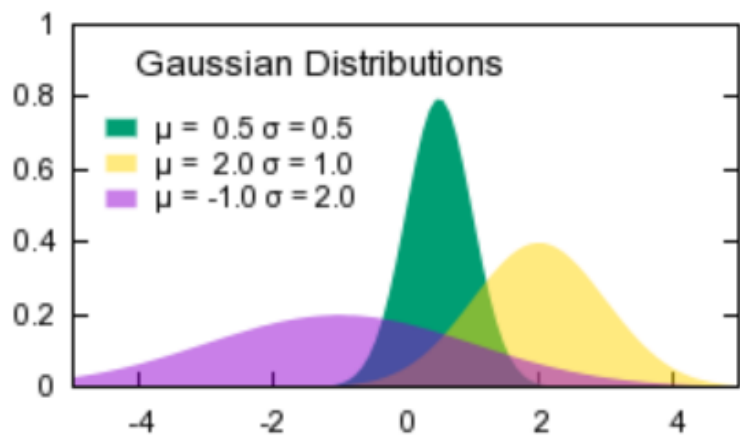
```
256000
Normal fcc (001) cell. Boxsize      144.8000    144.8000    144.8000
Cu  -72.399998    -72.399998    -72.399998     1
Cu  -70.589998    -70.589998    -72.399998     1
Cu  -70.589998    -72.399998    -70.589998     1
Cu  -72.399998    -70.589998    -70.589998     1
Cu  -72.399998    -72.399998    -68.779998     1
Cu  -70.589998    -70.589998    -68.779998     1
Cu  -70.589998    -72.399998    -66.969998     1
Cu  -72.399998    -70.589998    -66.969998     1
Cu  -72.399998    -72.399998    -65.159998     1
Cu  -70.589998    -70.589998    -65.159998     1
```



```
awk '
BEGIN {rmax=70;
}
{if(NR == 2) line=$0;
  if(NR > 2) {i++;
    if(NR == 3) {xmax=$2; ymax=$3; zmax=$4;
                xmin=$2; ymin=$3; zmin=$4;}
    el[i]=$1; x[i]=$2; y[i]=$3; z[i]=$4;
    if(xmax < $2) xmax=$2;
    if(ymax < $3) ymax=$3;
    if(zmax < $4) zmax=$4;

    if(xmin > $2) xmin=$2;
    if(ymin > $3) ymin=$3;
    if(zmin > $4) zmin=$4;
  }
  str=i;
}
END {
  xcenter=(xmax+xmin)/2;
  ycenter=(ymax+ymin)/2;
  zcenter=(zmax+zmin)/2;
  for(i=1; i<= str; i++) {
    x[i]=x[i]-xcenter;
    y[i]=y[i]-ycenter;
    z[i]=z[i]-zcenter;
    r=sqrt(x[i]*x[i] + y[i]*y[i] + z[i]*z[i]);
    if(r<rmax) {j++;}
  }
  natom=j; print natom; print line;
  for(i=1; i<= str; i++) {
    r=sqrt(x[i]*x[i] + y[i]*y[i] + z[i]*z[i]);
    if(r<rmax) {print el[i],x[i],y[i],z[i]}
  }
}'
```

gnuplot



[FAQ](#)

[Documentation](#)

[Demos](#)

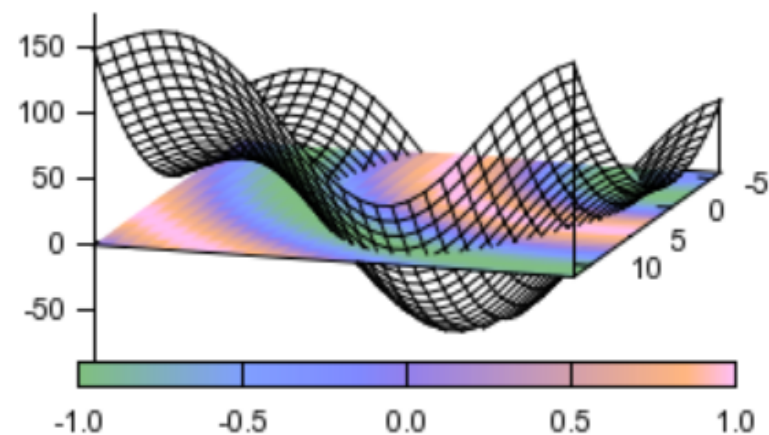
[Download](#)

[Contributed scripts](#)

[External Links](#)

[Tutorials and guides](#)

[Books](#)



- **Gnuplot** is a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms.

Example1 script for gnuplot

```
set terminal postscript portrait
set output "d1_plot.ps"
set size .75,.75
set title "Energy vs. Time for Sample Data"
set xlabel "Time"
set ylabel "Energy"
plot "d1.dat" with lines
pause -1 "Hit any key to continue"
```

```
• d1.dat
1 10
2 20
3 30
4 25
5 40
```

Example2 script for gnuplot

```
set term x11 enhanced font "terminal-14"  
#set output "data_p.ps"  
#set terminal postscript portrait  
set size .75,.75  
set grid  
set title 'BP and Heartrate'  
set yrange [50:160]  
set xlabel 'time (military)'  
set label 'finished walk' at 15, 140  
unset label  
set label 'finished walk' at 15, 105  
plot 'bp-hr.dat' u 1:2 w lp t 'systolic', 'bp-hr.dat' u 1:3 w lp t 'diastolic', 'bp-hr.dat' u 1:4 w lp t 'heartrate'  
pause -1 "Hit any key to continue"  
replot
```

• 'bp-hr.dat'			
8.5	112	60	52
9	116	73	59
10.5	127	71	58
11	124	69	62
11.5	117	68	60
12	122	73	60
13	121	67	62
15	120	78	68
15.5	134	70	96
16	120	72	73
16.5	114	68	72
22	119	69	61

Example3 gnuplot multiplot

```
#
# Set overall margins for the combined set of plots and size them
# to generate a requested inter-plot spacing
#
if (!exists("MP_LEFT"))    MP_LEFT = .1
if (!exists("MP_RIGHT"))  MP_RIGHT = .95
if (!exists("MP_BOTTOM")) MP_BOTTOM = .1
if (!exists("MP_TOP"))    MP_TOP = .9
if (!exists("MP_GAP"))    MP_GAP = 0.05

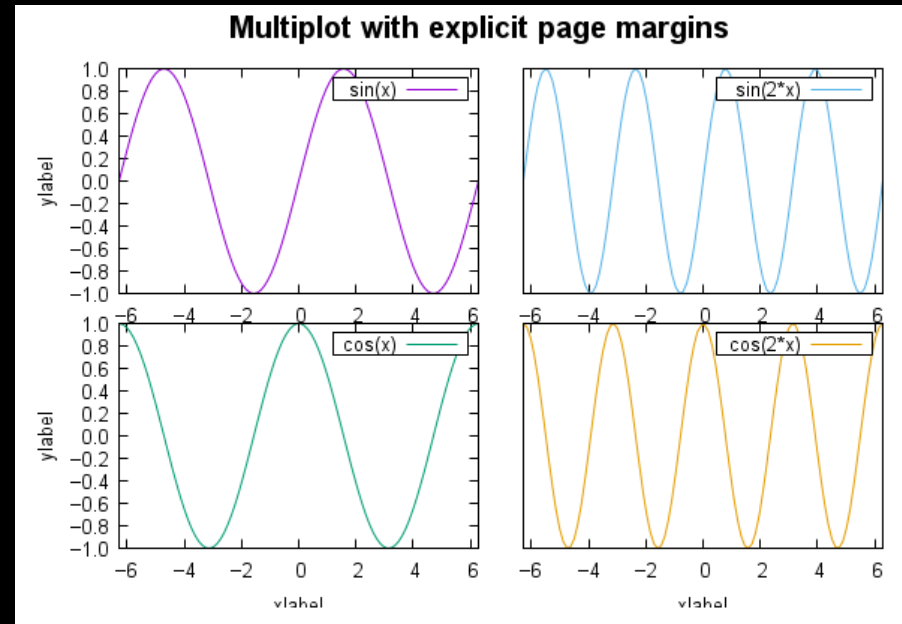
set multiplot layout 2,2 columns first title "{/:Bold=15 Multiplot with explicit page margins}"
      margins screen MP_LEFT, MP_RIGHT, MP_BOTTOM, MP_TOP spacing screen MP_GAP

set format y "%.1f"
set key box opaque
set ylabel 'ylabel'
set xrange [-2*pi:2*pi]

plot sin(x) lt 1
set xlabel 'xlabel'
plot cos(x) lt 2

unset ylabel
unset ytics

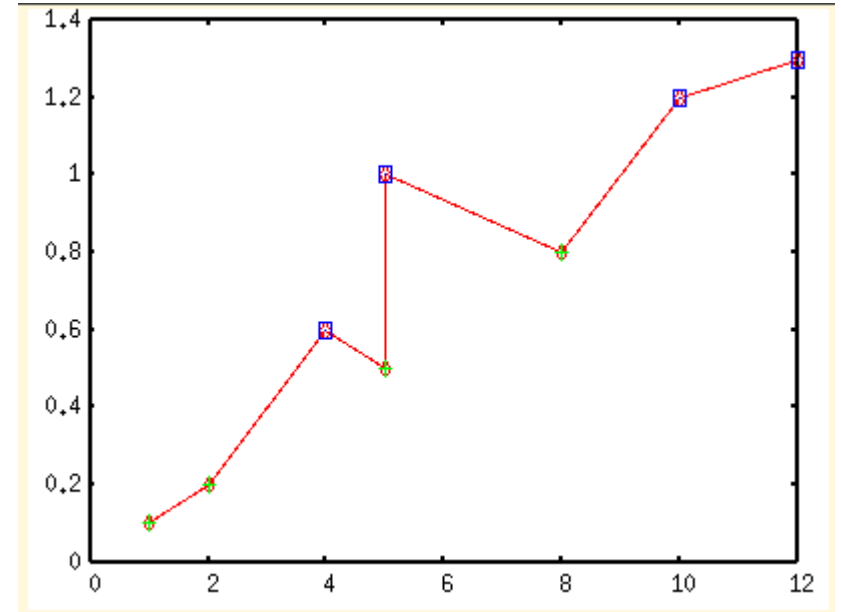
unset xlabel
plot sin(2*x) lt 3
set xlabel 'xlabel'
plot cos(2*x) lt 4
unset multiplot
```



Example gnuplot & bash

file1.dat	file2.dat
1.0 0.1	4.0 0.6
2.0 0.2	5.0 1.0
5.0 0.5	10.0 1.2
8.0 0.8	12.0 1.3

```
gnuplot> plot "< cat -n file1.dat file2.dat" using 1:2 w l,\n>           "file1.dat" u 1:2 w p,\n>           "file2.dat" u 1:2 w p
```



```
gnuplot> plot "<awk '{print $1,($1<=3 && $1>=1) ? $2*5 : $2}' file1.dat" with lines
```

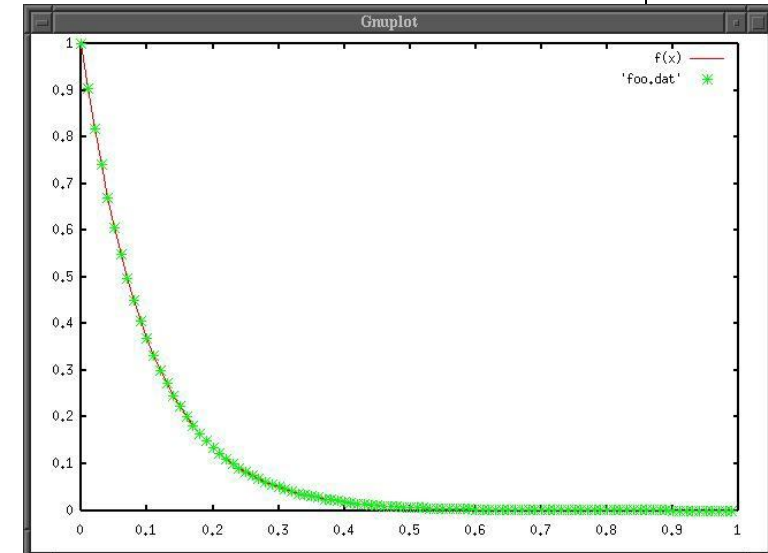
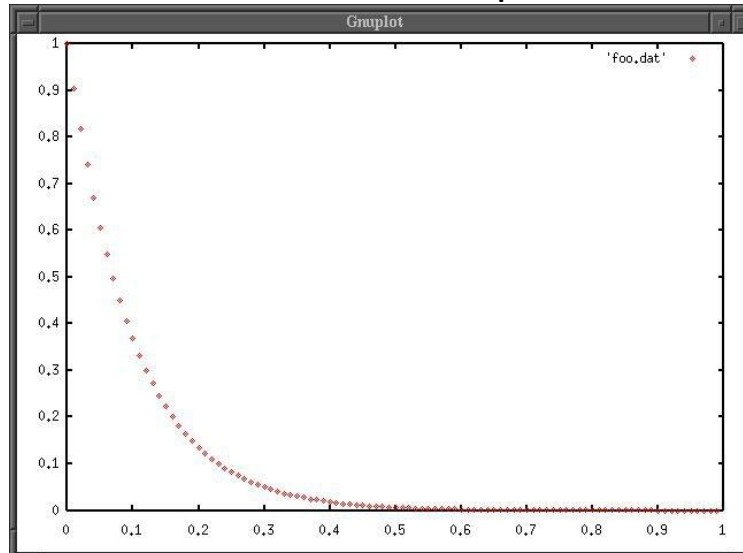
Regression with gnuplot

- gnuplot> plot 'foo.dat'
- gnuplot> f(x) = a * exp (-x*b)
- gnuplot> fit f(x) 'foo.dat' via a,b
- gnuplot> plot f(x), 'foo.dat'

- gnuplot> set term post eps
- Options are 'eps noenhanced monochrome dashed defaultplex "Helvetica" 14'
- gnuplot> set output 'foo.eps'
- gnuplot> plot f(x), 'foo.dat'

- # reset to default
- gnuplot> set term x11
- gnuplot> set output

```
• foo.dat
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
0.0800000 0.449329
0.0900000 0.406570
0.100000 0.367879
0.110000 0.332871
0.120000 0.301194
0.130000 0.272532
0.140000 0.246597
```



Regression with gnuplot 2

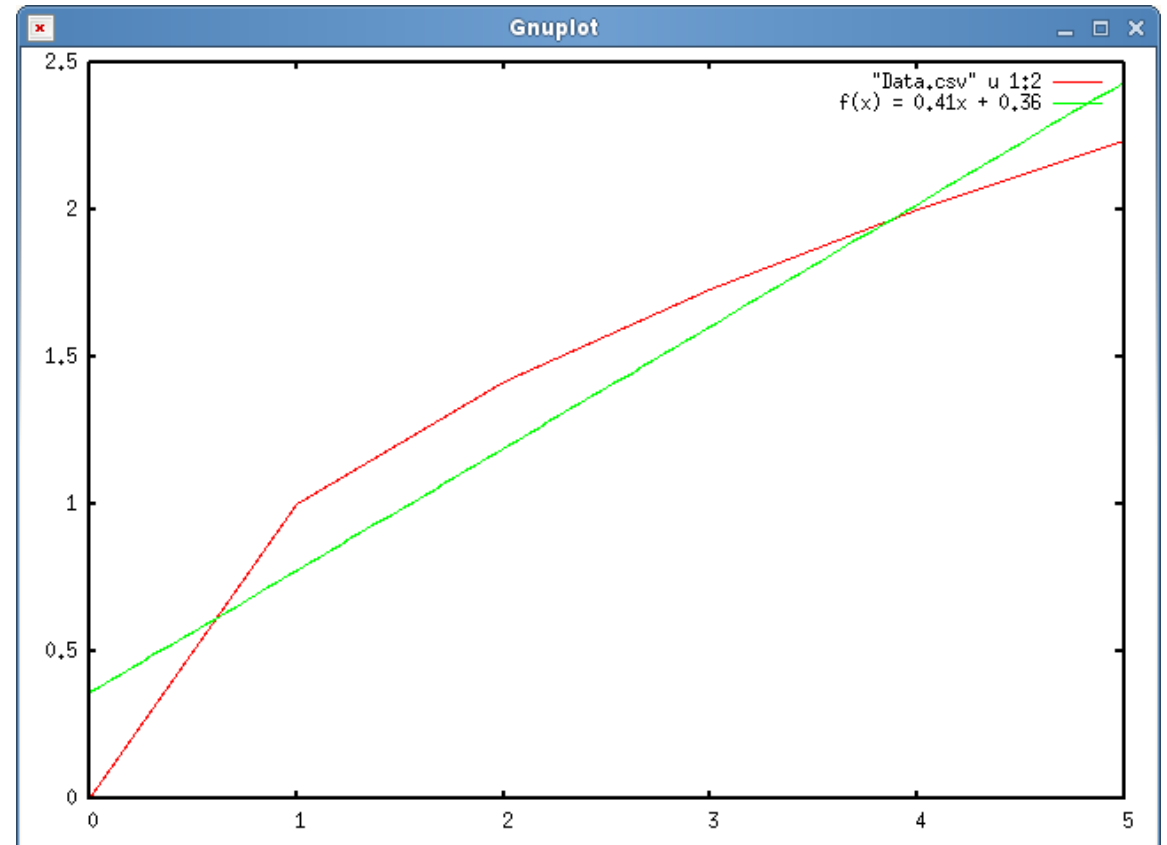
$$f(x) = a*x + b$$

fit f(x) 'Data.csv' u 1:2 via a, b

title_f(a,b) = sprintf('f(x) = %.2fx + %.2f', a, b)

plot "Data.csv" u 1:2 w l, f(x) t title_f(a,b)

```
• foo.dat
0 0.00000
1 1.00000
2 1.41421
3 1.73205
4 2.00000
5 2.23607
```



Thanks !