
IntroToCommandLine Documentation

Release 1

Grainne Kerr, Holger Dinkel, Matt Betts, Jon Fuller

September 18, 2013

1	Introduction to the Linux Commandline	3
1.1	Why Use the Commandline	3
1.2	General Remarks Regarding Using UNIX/Linux Systems	3
1.3	General Structure of Linux Commands	5
1.4	Getting Help	6
1.5	Who am I, where am I	6
2	Exercises	17
2.1	Misc. file tools	17
2.2	Searching	17
2.3	Misc. terminal	17
2.4	Permissions	17
2.5	Remote access	18
2.6	IO and Redirections	18
3	File Formats	19
3.1	Presentation Task	19
4	Files	21
5	Quality Control	23
5.1	In this section you will need to use the following three files:	23
5.2	SAMstat	24
6	Aligning RNASeq Reads	25
6.1	Example 1 - arm.Xsubset.fastq	25
6.2	Example 2 - smo.2Lsubset.fastq	26
7	Samtools	27
7.1	Getting bored? Good to know:	29
8	Estimating Expression	31
8.1	Example - Expression in the arm gene knockdown sample	31
8.2	Review Questions:	31
8.3	Task	32
9	Estimating differential Expression	33
9.1	Example - Estimating significance of differential expression in the ARM gene knockdown compared to control.	33
9.2	Use your linux know how	33
9.3	Task	34

Contents:

INTRODUCTION TO THE LINUX COMMANDLINE

1.1 Why Use the Commandline

- It's **fast**. Productivity is a word that gets tossed around a lot by so-called power users, but the command line can really streamline your computer use, assuming you learn to use it right.
- It's **easier to get help**. The command line may not be the easiest thing to use, but it makes life a whole lot easier for people trying to help you and for yourself when looking for help, especially over the internet. Many times it's as simple as the helper posting a few commands and some instructions and the recipient copying and pasting those commands. Anyone who has spent hours listening to someone from tech support say something like, "OK, now click this, then this, then select this menu command" knows how frustrating the GUI alternative can be.
- It's nearly **universal**. There are hundreds of Linux distros out there, each with a slightly different graphical environment. Thankfully, the various distros do have one common element: the command line. There are distro-specific commands, but the bulk of commands will work on any Linux system.
- It's **powerful**. The companies behind those other operating systems try their best to stop a user from accidentally screwing up their computer. Doing this involves hiding a lot of the components and tools that could harm a computer away from novices. Linux is more of an open book, which is due in part to its prominent use of the command line.

1.2 General Remarks Regarding Using UNIX/Linux Systems

- **Test before run**. Anything written here has to be taken with a grain of salt. On another system – be it a different Linux distribution or another UNIXoid operating system – you might find the same command but without the support of some of the options taught here. It is even possible, that the same option has a different meaning on another system. With this in mind always make sure to test your commands (specially the "dangerous" ones which remove or modify files) when switching from one system to the other.
- **The Linux/UNIX environment**. The behaviour of many commands is influenced or controlled by the so-called "environment". This environment is the sum of all your environment variables. Some of these environment variables will be shown towards the end of this course.
- **UPPERCASE, lowercase**. Don't forget that everything is case-sensitive.
- **The Filesystem**. Linux filesystems start on top at the root directory (sic!) "/" which hierarchically broadens towards the ground. The separator between directories or directories and files in Linux is the slash ("/").

..note:: The terms "directory" and "folder" are used interchangeably in this document.

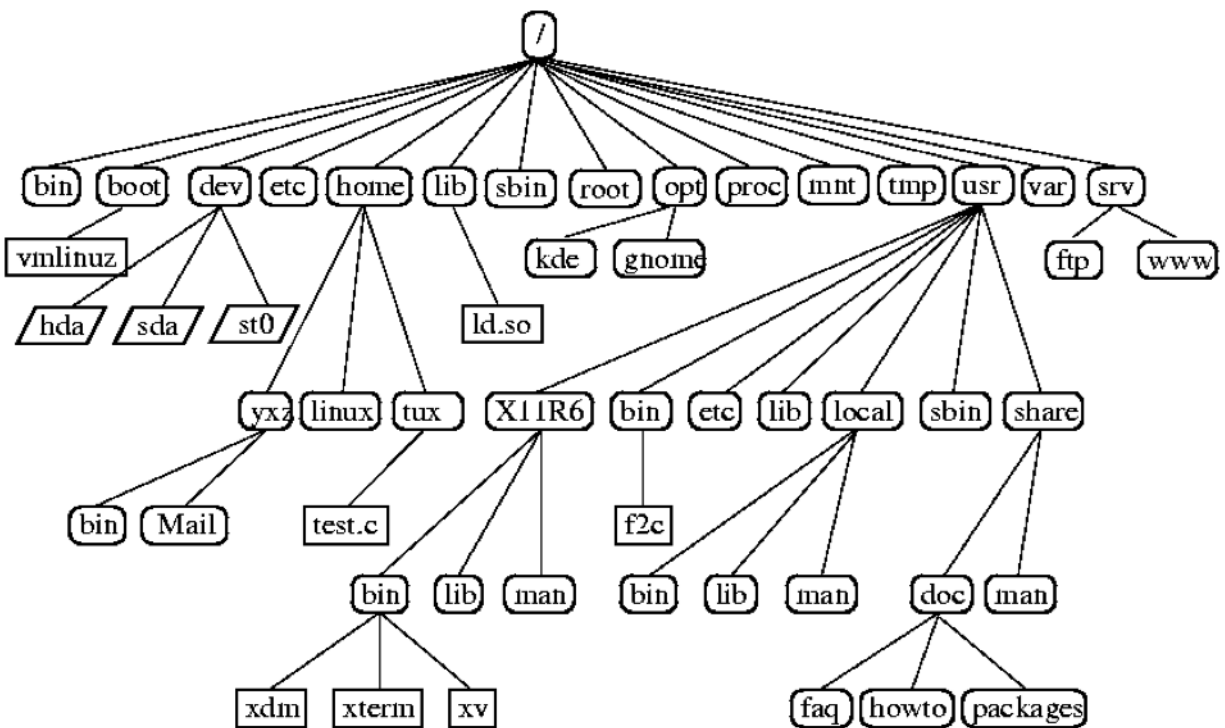
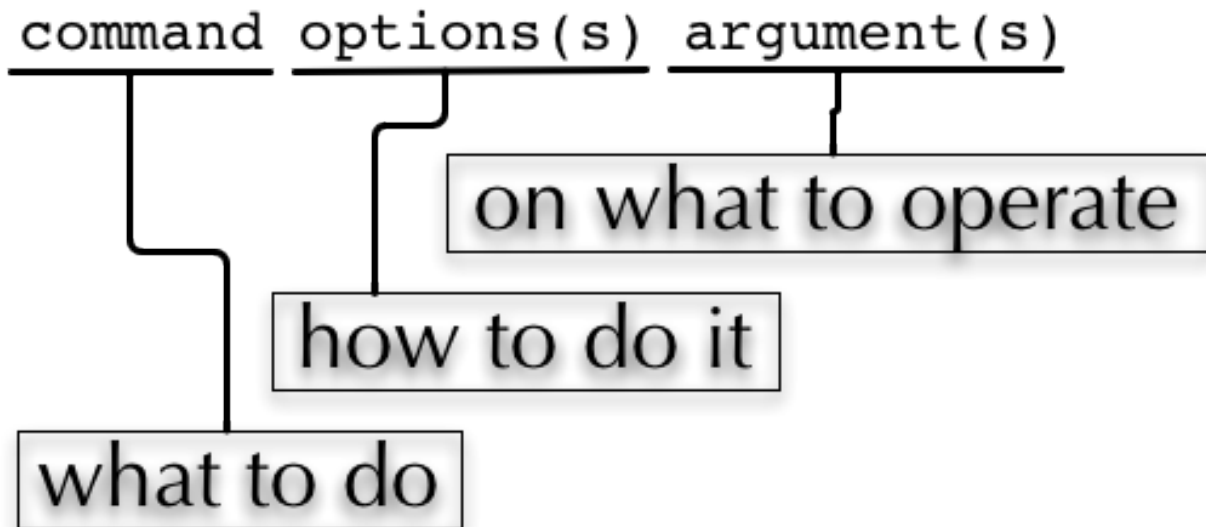


Figure 1.1: Depending on the Linux distribution you might or might not find all of above directories. Most important directories for you are `/bin` and `/usr/bin` (sometimes also `/usr/local/bin`) which contain the user software, `/home` which usually contains the users' homedirectories and `/tmp` which can be used to store temporary data (beware: Its content is regularly removed!).

1.3 General Structure of Linux Commands

Linux commands have the following general structure:



commandline options (sometimes called comandline switches) commonly have one of the two following forms: The short form `-character` or the long form `--string`. E.g.

```
> man -h
> man --help
```

Short options are usually – though not always – concatenable:

```
> ls -l -A -h
> ls -lAh
```

Some options require an additional argument, which is added with a blank to the short form and with an equal sign to the long form:

```
> ls -I "*.pdf"
> ls --ignore="*.pdf"
```

Since Linux incorporates commands from different sources, options can be available in one or both forms and you'll also encounter options with no dash at all and all kinds of mixtures:

```
> tar cf file.tar -C .. file/
> ps auxgww
```

1.3.1 A Journey Through the Commands

Please note that all examples and usage instructions below are just a glimpse of what you can do and reflect our opinion on what's important and what's not. Most of these commands support many more options and different usages. Consult the manpages to find them. Typographical conventions: Commands and examples are written in Courier. User Input is written in Courier bold and placeholders are generally written in *italic*.

1.4 Getting Help

`-h/--help` option, no parameters

Many commands support a “help” option, either through `-h` or through `--help`. Other commands will show a help page or at least a short usage overview if you provide wrong commandline options

Usage: `man command or file`

```
> man man
man(1)

NAME
  man - format and display the on-line manual pages

SYNOPSIS
  man [-acdfFhkKtwW] [--path] [-m system] [-p string] [-C config_file]
  [...]
```

For the navigation within a man-page see the chapter regarding less below.

Note: The behaviour of `man` is dependent of the `$PAGER` environment variable

Usage: `apropos keyword`

```
> apropos who
[...]
> who                (1)  - show who is logged on
> who                (lp) - display who is on the system
> whoami             (1)  - print effective userid
```

Use `apropos` to find candidates for specific tasks

The `/usr/share/doc` directory in some Linux distributions contains additional documentation of installed software packages

1.5 Who am I, where am I

Usage: `whoami`

```
> whoami
fthommen
```

Usage: `hostname`

```
> hostname
pc-teach01
```

Usage: `pwd`

```
> pwd
/home/fthommen
```

Usage: `date`

```
> date
Tue Sep 25 19:57:50 CEST 2012
```

Note: The command time does something completely different than date and is not used to show the current time.

Usage: cd [new_directory]

```
# pwd
/home/fthommen
# cd /usr/bin
# pwd
/usr/bin
```

Special directories:

- “.”: The current working directory
- “..”: The parent directory of the current working directory
- “~”: Your homedirectory

Note: Using cd without a directory is equivalent to “cd ~” and changes into the users’s homedirectory

Note: Please note the difference between absolute pathes (starting with “/”) and relative pathes (starting with a directory name)

```
$ pwd
/usr
$ cd /bin
$ pwd
/bin
```

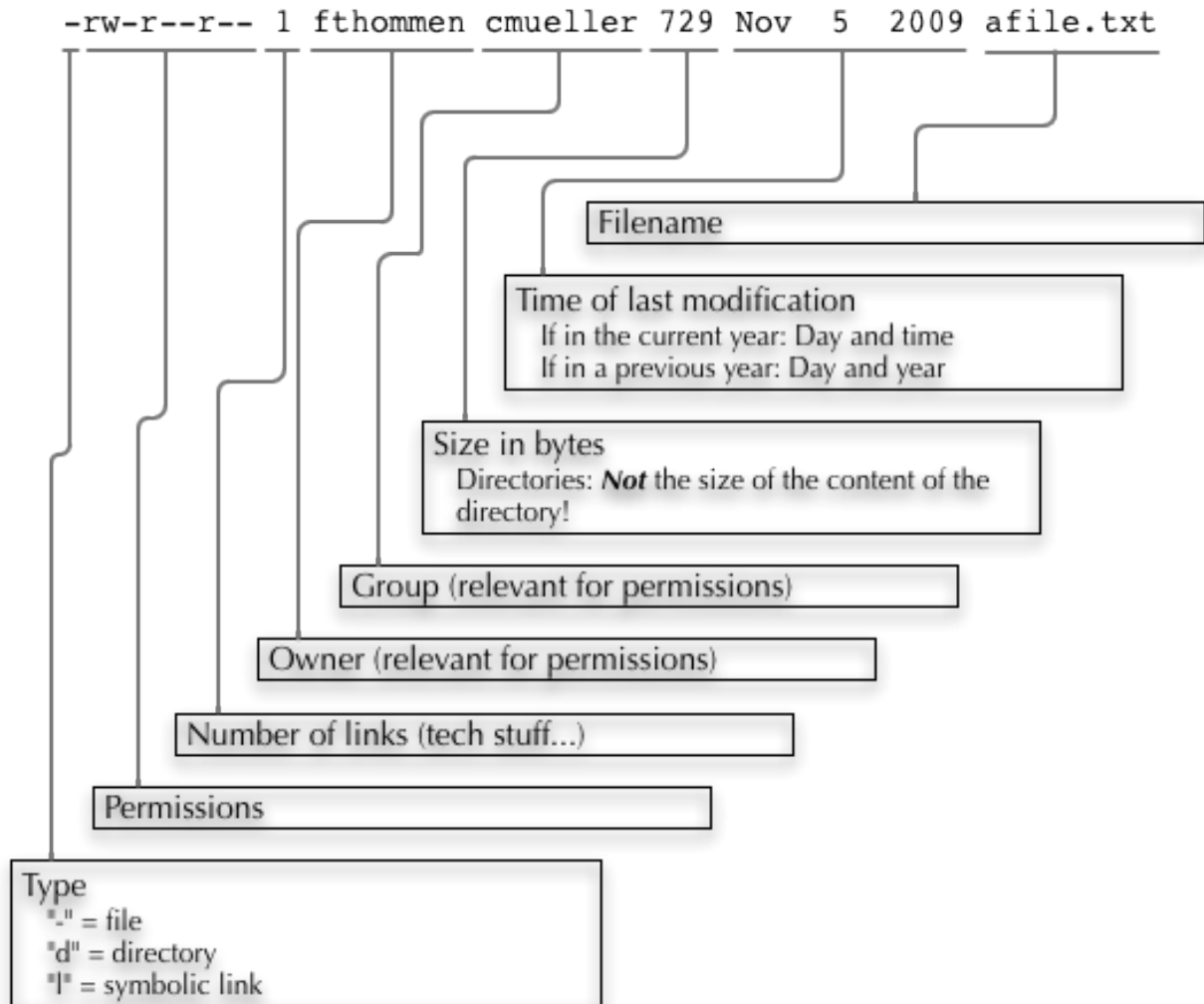
```
> pwd
/usr
> cd bin
> pwd
/usr/bin
```

Usage: ls [options] [file(s) or directory/ies]

```
> ls
/home/fthommen
> ls -l aa.pdf
-rw-r--r-- 1 fthommen cmueller 0 Sep 24 10:59 aa.pdf
```

Useful options:

-l	Long listing with permissions, user, group and last modification date
-1	Print listing in one column only
-a	Show all files (hidden, “.” and “..”)
-A	Show almost all files (hidden, but not “.” and “..”)
-F	Show filetypes (nothing = regular file, “/” = directory, “*” = executable file, “@” = symbolic link)
-d	Show directory information instead of directory content
-t	Sort listing by modification time (most recent on top)



Files and folders can't only be referred to with their full name, but also with so-called "Shell Globs", which are a kind of simple pattern to address groups of files and folders. Instead of explicit names you can use the following placeholders:

- `?`: Any single character
- `*`: Any number of any character (including no character at all)
- `[...]`: One of the characters included in the brackets. Use `--` to define ranges of characters

Examples:

- `*.pdf`: All files having the extension `".pdf"`
- `? .jpg`: Jpeg file consisting of only one character
- `[0-9]*.txt`: All files starting with a number and having the extension `".txt"`
- `*.???`: All files having a three-character extension

Note: The special directory `"~"` mentioned above is a shell glob, too.

Usage: `touch file(s) or directory/ies`

```
> ls afile
ls: afile: No such file or directory
> touch afile
> ls afile
afile

> ls -l aa.pdf
-rw-r--r-- 1 fthommen cmueller 0 Sep 24 10:59 aa.pdf
> touch aa.pdf
> ls -l aa.pdf
-rw-r--r-- 1 fthommen cmueller 0 Sep 25 22:01 aa.pdf
```

Usage: `rm [options] file(s)`

```
rm -r [options] directory/ies
> ls afile
afile
> rm afile
> ls afile
ls: afile: No such file or directory
```

Useful options:

- | | |
|-----------|---------------------------------------------------------------------|
| -i | Ask for confirmation of each removal |
| -r | Remove recursively |
| -f | Force the removal (no questions, no errors if a file doesn't exist) |

Note: `rm` without the `-i` option will usually not ask you if you really want to remove the file or directory

Usage: `mv [options] sourcefile destinationfile`

```
mv [options] sourcefile(s) destinationdirectory
> ls *.txt
a.txt
```

```
> mv a.txt b.txt
> ls *.txt
b.txt
```

Useful options:

-i Ask for confirmation of each removal

Note: You cannot overwrite an existing directory by another one with mv

Usage: mkdir [options] directory

```
> ls adir/
ls: adir/: No such file or directory
> mkdir adir
> ls adir
```

Useful options:

-p Create parent directories (when creating nested directories)

```
> mkdir adir/bdir
mkdir: cannot create directory 'adir/bdir': No such file or directory
> mkdir -p adir/bdir
```

Usage: rmdir directory

```
> rmdir adir/
```

Note: If the directory is not empty, rmdir will complain and not remove it

Usage: cp [options] sourcefile destinationfile .. note:: cp [options] sourcefile(s) destinationdirectory

```
> cp P12931.fasta backup_of_P12931.fasta
```

Useful options:

-r Copy recursively

-i Interactive operation, ask before overwriting an existing file

-p Preserve owner, permissions and timestamp

Usage: cat [options] file(s)

```
> cat P12931.fasta backup_of_P12931.fasta
[...]
```

Note: cat only makes sense for short files or for e.g. combining several files into one. See the redirection examples later

Usage: less [options] file(s)

```
> less P12931.fasta backup_of_P12931.fasta
[...]
```

Note: This is the default “pager” for manpages under Linux unless you redefine your \$PAGER environment variable

Navigation within less:

Key(s):	Effect:
up, down, right, left:	use cursor keys
top of document:	g
bottom of document:	G
search:	"/" + search-term
find next match:	n
find previous match:	N
quit:	q

Grep is a command-line utility for searching plain-text data sets for lines matching a regular expression.

Usage: `grep [options] pattern file(s)`

```
> grep -i ensembl P04637.txt
DR   Ensembl; ENST00000269305; ENSP00000269305; ENSG00000141510.
DR   Ensembl; ENST00000359597; ENSP00000352610; ENSG00000141510.
DR   Ensembl; ENST00000419024; ENSP00000402130; ENSG00000141510.
DR   Ensembl; ENST00000420246; ENSP00000391127; ENSG00000141510.
DR   Ensembl; ENST00000445888; ENSP00000391478; ENSG00000141510.
DR   Ensembl; ENST00000455263; ENSP00000398846; ENSG00000141510.
```

Useful options:

- v** Print lines that do not match
- i** Search case-insensitive
- l** List files with matching lines, not the lines itself
- L** List files without matches
- c** Print count of matching lines for each file

Head is a program on Unix and Unix-like systems used to display the beginning of a text file or piped data.

Usage: `head [options] file(s)`

```
> head /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
```

Useful options:

- n num** Print num lines (default is 10)

Usage: `tail [options] file(s)`

```
> tail -n 3 /etc/passwd
xfs:x:43:43:X Font Server:/etc/X11/fs:/sbin/nologin
gdm:x:42:42:./var/gdm:/sbin/nologin
sabayon:x:86:86:Sabayon user:/home/sabayon:/sbin/nologin
```

Useful options:

- n num** Print num lines (default is 10)

-f “Follow” a file (print new lines as they are written to the file)

Usage: `file [options] file(s)`

```
> file /bin/date
/bin/date: ELF 32-bit LSB executable
> file /bin
/bin: directory
> file SRC_HUMAN.fasta
SRC_HUMAN.fasta: ASCII text
```

Note: The command `file` uses certain tests and some magic to determine the type of a file

Usage: `which [options] command(s)`

```
> which date
/bin/date
> which eclipse
/usr/bin/eclipse
>
```

Usage: `find [starting path(es)] [search filter]`

```
> find /etc
/etc
/etc/printcap
/etc/protocols
/etc/xinetd.d
/etc/xinetd.d/ktalk
[...]
>
```

`find` is a powerful command with lots of possible search filters. Refer to the manpage for a complete list.

Examples:

- Find by name:

```
> find . -name SRC_HUMAN.fasta
./SRC_HUMAN.fasta
```

- Find by size: (List those entries in the directory `/usr/bin` that are bigger than 500kBytes)

```
> find /usr/bin -size +500k
/usr/bin/oparchive
/usr/bin/kiconedit
/usr/bin/opjitconv
[...]
```

- Find by type (d=directory, f=file, l=link)

```
> find . -type d
.
./adir
```

Usage: `clear`

```
> clear
```

In case the output of the terminal/screen gets cluttered, you can use `clear` to clear the screen...

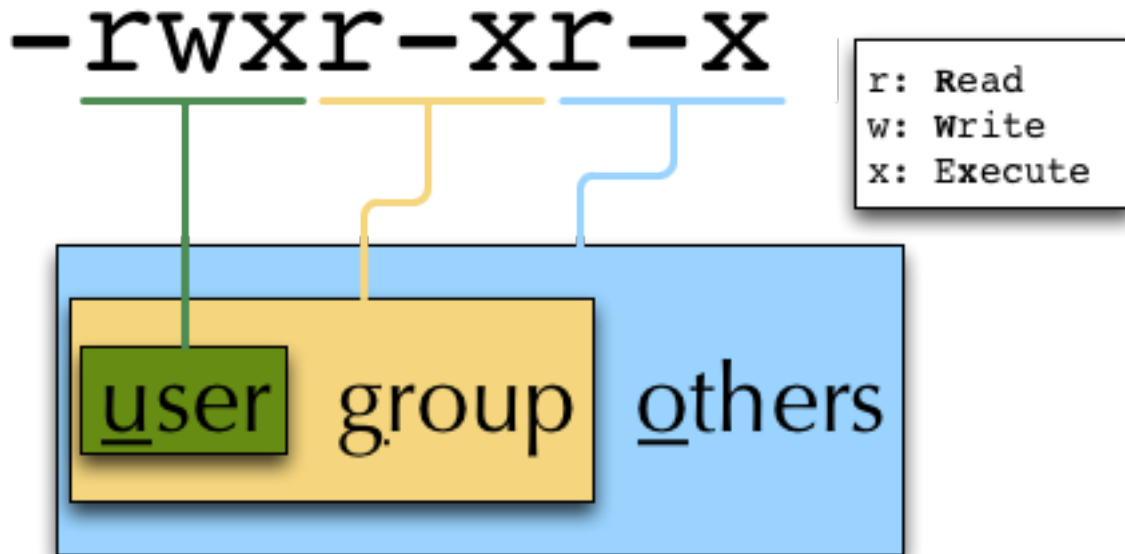
If this doesn't work, you can use `reset` to perform a re-initialization of the terminal:

Usage: reset [options]

```
> reset
```

using ls -l to view entries of current directory:

```
> ls -l
drwxr-xr-x 2 dinkel gibson 4096 Sep 17 10:46 adir
lrwxrwxrwx 1 dinkel gibson   15 Sep 17 10:45 H1.fasta -> H2.fasta
-rw-r--r-- 1 dinkel gibson  643 Sep 17 10:45 H2.fasta
```



Permissions are set using the `chmod` (change mode) command. **Usage:** `chmod [options] mode(s) files(s)`

```
> ls -l adir
drwxr-xr-x 2 dinkel gibson 4096 Sep 17 10:46 adir
> chmod u-w,o=w adir
> ls -l adir
dr-xr-x-w- 2 dinkel gibson 4096 Sep 17 10:46 adir
```

The mode is composed of

Who		What		Which permission	
u:	user/owner	+:	add this permission	r:	read
g:	group	-:	remove this permission	w:	write
o:	other	=:	set exactly this permission	x:	execute
a:	all xx		xx	xx	xx

Add executable permission to the group:

```
> chmod g+x file
```

Revoke this permission:

```
> chmod g-x file
```

Allow all to read a directory:

```
> chmod a+rx adir/
```

To execute commands at a remote machine/server, you need to log in to this machine. This is done using the `ssh` command (secure shell). In its simplest form, it takes just the machinename as parameter (assuming the username on the local machine and remote machine are identical):

```
> ssh remote_server
```

Note: Once logged in, use `hostname`, `whoami`, etc. to determine on which machine you are currently working!

To use a different username, you can use either:

```
> ssh username@remote_server
```

or

```
> ssh -l username remote_server
```

When connecting to a machine for the first time, it might display a warning:

```
> ssh sub-master
The authenticity of host 'sub-master (10.11.4.84)' can't be established.
RSA key fingerprint is 47:a4:0f:7b:c2:0f:ef:91:8e:65:fc:3c:f7:0c:53:8d.
Are you sure you want to continue connecting (yes/no)?
```

Type *yes* here. If this message appears a second time, you should contact your IT specialist...

To disconnect from the remote machine, type:

```
> exit
```

Copying files to and from remote computers can be done using `scp` (secure copy). The order of parameters is the same as in `cp`: first the name of the source, then the name of the destination. Either one can be the remote part.

```
> scp localfile server:/remotefile
> scp server:/remotefile localfile
```

An alternative username can be provided just as in `ssh`:

```
> scp username@server:/remotefile localfile
```

Redirect the output of one program into e.g. a file: (Caution: you can easily overwrite files by this!) Inserting the current date into a new file:

```
> date > file_containing_date
```

Filtering lines containing the term “src” from FASTA files and inserting them into the file `lines_with_src.txt`:

```
> cd /exercises/
> grep -i "src" *.fasta > lines_with_src.txt
```

Append something to a file (rather than overwriting it):

```
> date >> file_containing_date
```

Use the `|` pipe symbol (`|`) to feed the output of one program into the next program. Here: use `ls` to show the directory contents and then use `grep` to only show those that contain fasta in their name:

```
> cd /exercises
> ls | grep fasta
EPSINS.fasta
FYN_HUMAN.fasta
P12931.fasta
SRC_HUMAN.fasta
```

Environment variables are a set of dynamic named values that can affect the way running processes will behave on a computer.

Contains the location of the user's home directory. Although the current user's home directory can also be found out through the C functions `getpwuid` and `getuid`, `$HOME` is often used for convenience in various shell scripts (and other contexts).

Note: Do not change this variable unless you have a good reason and you know what you are doing!

`$PATH` contains a colon-separated list of directories that the shell searches for commands that do not contain a slash in their name (commands with slashes are interpreted as file names to execute, and the shell attempts to execute the files directly).

The `$PAGER` variable contains the path to the program used to list the contents of files through (such as `less` or `more`).

The `$PWD` variable points to the current directory. Equivalent to the output of the command `pwd` when called without arguments.

Use `echo` to display individual variables `set` or `env` to view all at once:

```
> echo $HOME
/localhome/teach01
> set
...
> env
...
```

Use `export` followed by the variable name and the value of the variable (separated by the equal sign) to set an environment variable:

```
> export PAGER=/usr/bin/less
```

Note: An environment variable is only valid for your current session. Once you logout of your current session, it is lost or reset.

EXERCISES

2.1 Misc. file tools

1. Which tool can be used to determine the type of a file?
2. Use it on the following files/directories and compare the results:
 - (a) `/usr/bin/tail`
 - (b) `~`
 - (c) `/exercises/SRC_HUMAN.fasta`

2.2 Searching

1. Which tool can be used to search for files or directories?
2. Use it to find all directories in the `/exercises` directory
3. Search for the file `date` in the `/bin` directory
4. List those entries in the directory `/bin` that are bigger than 400kBytes

2.3 Misc. terminal

1. Which two tools can be used to redraw/empty the screen?

2.4 Permissions

1. Create a directory called `testpermissions`
2. Change your working directory to `testpermissions`
3. Create a directory called `adir`.
4. Use the command `which date` to find out where the `date` program is located.
5. Copy this `date` program into the directory `adir`.
6. Check the permissions of the copied program `date`
7. Change the permissions on `date` to remove the executable permissions.

8. Check the permissions of the program `date`
9. Try running it as `./date` or `adir/date` (depending on your current working directory)
10. Change the permissions back so that the file is executable.
11. Try running it as `./date` or `adir/date` (depending on your current working directory)
12. Copy a textfile from a previous exercise into `adir`, then change the permissions, so you are not allowed to write to it.
13. Then change the permissions so you can't read/cat it either.
14. Change your working directory to `testpermissions`, and then try changing the permissions on `adir`.
15. What are the minimum permissions (on the directory) necessary for you to be able to execute `adir/date`?

2.5 Remote access

1. Login to machine "sub-master.embl.de" (using your own username)
2. Use `exit` to quit the remote shell (Beware to not exit your local shell)
3. Use `clear` to empty the screen after logout from the remote server
4. Use the following commands locally as well as on the remote machine to get a feeling for the different machines:
 1. `hostname`
 2. `whoami`
 3. `cat /etc/hostname`
 4. `ls -la ~/`
1. Copy the file `/etc/motd` from machine `sub-master.embl.de` into your local home directory
2. Determine the filetype and the permissions of the file that you just copied
3. Login to your neighbor's machine (ask them for the hostname) using the username `teach01` (password will be given by teacher)

2.6 IO and Redirections

1. Use `date` in conjunction with the redirection to insert the current date into the (new) file `current_date` (in your homedirectory).
2. Inspect the file to make sure it contains (only a single line with) the date.
3. Use `date` again to append the current date into the same file.
4. Again, check that this file now contains two lines with dates.
5. Use `grep` to filter out lines containing the term "TITLE" from all PDB files in the exercises directory and use redirection to insert them into a new file `pdb_titles.txt`.
6. (OPTIONAL) Upon inspection of the file `pdb_titles.txt`, you see that it also contains the names of the files in which the term was found. Use either the `grep` manpage or `grep --help` to find out how you can suppress this behaviour. Redo the previous exercise such that the output file `pdb_titles.txt` only contains lines starting with `TITLE`.

FILE FORMATS

File Types to research and present:

- Fasta/fastq (sanger and illumina)/fai
- GFF/GTF
- Sam/Bam/Bai
- Vcf/bcf/Pileup/interval/ROD
- Bed/bigBed
- Wig/bigWig

3.1 Presentation Task

Give a white-board/flip-chart presentation on one of the above file types. Your presentation should be 5 - 7 mins. Questions you could answer in your presentation:

- **What does each line in the file represent?**
- **What does each column represent?**
- **What character separates the columns? Can you write this character?**
- **Is there a file header?**
- **What online repositories are available to download this data and how can I download it?**
- What type of user/database/online tools uses this file format/who came up with it?
- Show an example of the file.
- What is the difference between the files you've been given?
- Can you convert between one file format and another?
- Any other information interesting information?

Note: Questions in bold should be asked about any file format you have been given.

Unsure about jargon? Ask for help!

3.1.1 Useful links about file formats

VCF:

<http://www.1000genomes.org/wiki/analysis/variant-call-format/vcf-variant-call-format-version-42>

GFF:

- <http://www.ensembl.org/info/website/upload/gff.html>
- <http://www.sanger.ac.uk/resources/software/gff/spec.html>

BED/BIGBED/WIG/BIGWIG:

- <http://genome.ucsc.edu/FAQ/FAQformat.html#format1>
- <http://www.ensembl.org/info/website/upload/bed.html>
- <http://www.ensembl.org/info/website/upload/wig.html>

SAM/BAM/BAI:

- <http://genome.sph.umich.edu/wiki/SAM>
- <http://samtools.sourceforge.net/SAMv1.pdf>
- <http://blog.nextgenetics.net/?e=18> = bitwise flags
- <http://picard.sourceforge.net/explain-flags.html>

FASTA/FASTQ:

- http://en.wikipedia.org/wiki/FASTQ_format#Illumina_sequence_identifiers
- http://bioinf.comav.upv.es/courses/sequence_analysis/sequence_file_formats.html
- <http://blog.nextgenetics.net/?e=33> = phred quality

All files!

<http://www.broadinstitute.org/igv/FileFormats> <http://genome.ucsc.edu/FAQ/FAQformat>

NEXT STEPS

Look at the databases mentioned in the presentations and go through them in a live demo. Typical will be Ensembl, UCSC, NCBI, GATK, 1000 genomes

SAM very important – re-enforce QUAL score, CIGAR string, FLAG Quality scores – make reference to illumina vs sanger (Sanger pretty standard now)

Raw data:

- data/smo.bam
- data/arm.bam
- data/fl1-1.bam
- data/fl1-2.bam
- data/smo.2Lsubset.fastq
- data/arm.Xsubset.fastq
- data/dnaSeq1.bam
- data/dnaSeq2.bam

Gene Reference:

- geneRef/dros_BD5.25.gtf
- geneRef/drosophilaMelanogaster.X.gtf
- geneRef/drosophilaMelanogaster.2L.gtf

Fasta Files:

- genome/dros_BD5.25.fa

Bowtie Index Files:

- bowtieIndex/drosophilaMelanogaster.2L
- bowtieIndex/drosophilaMelanogaster.X

Tophat output Files:

Cufflinks Output Files:

Other:

- header.sam
- intervalFile.bed

QUALITY CONTROL

5.1 In this section you will need to use the following three files:

- data/arm.fastq
- data/arm.bam
- data/FL1-1.sam

As a reminder, you can find out what each of these files is used for here.

One of the most important steps before spending lots of time on an analysis, is to check that the raw data from the sequencing run is of good enough quality. We will use a tool called `fastqc` to do some preliminary quality control of our raw data. The `fastqc` tool generates summary statistics of sequence and quality data and can be used to filter, reformat and trim next-generation sequence data.

Make a directory in your home directory called `fastqcTest`.

Recall, that when we don't know what a tool does, we can always get help (see section [Getting Help](#)):

We will use a tool called **fastqc** to do some preliminary quality control of our raw data. This is a tool that generates summary statistics of sequence and quality data and can be used to filter, reformat and trim next-generation sequence data.

Use your linux know how to make a directory in your home directory called `fastqcTest`.

Get help on how to use `fastqc`

```
> fastqc -h
```

Report graphs allow us to gain a visual overview of the sequence data. To generate report graphs type the following:

```
> fastqc -verbose -o fastqcTest -noextract -f fastq arm.fastq -
```

Review Questions

- What does the option `-o` do in the above command?
- Does the `-o` option have a value associated with it?
- What does the `-noextract` option do? What is the difference between the `-o` option and the `-noextract` option?

5.1.1 Task

Use `fastqc` to generate a report of `sno.fastq`

5.2 SAMstat

SAMStat is an efficient C program to quickly display statistics of large sequence files from next generation sequencing projects. When applied to SAM/BAM files all statistics are reported for unmapped, poorly and accurately mapped reads separately. This allows for identification of a variety of problems, such as remaining linker and adaptor sequences, causing poor mapping. Apart from this SAMStat can be used to verify individual processing steps in large analysis pipelines.

Use samstat to get statistics on the mapped data file in *arm.fastq* and *arm.bam*

Getting help

```
> samstat -h
```

Note the usage line you are given after typing this

```
Usage:    samstat <file.sam> <file.bam> <file.fa> <file.fq> ....
```

To get the statistics of the *arm.bam* file using samstat. This will create a html report in the current working directory

```
> samstat arm.bam
```

5.2.1 Task

Use samstat to create html report for *arm.fastq* and *FL1-1.sam*

ALIGNING RNASEQ READS

6.1 Example 1 - arm.Xsubset.fastq

This dataset contains reads from a mRNA sample after RNAi knockdown of the arm gene, in the drosophila Melanogaster cell line, S2. The arm gene is found on chromosome X of the drosophila Melanogaster genome. Here, to save time, we will restrict our analysis to this chromosome. You should for your own projects, you use all the genome information.

Your task is to align the reads to the X chromosome

To get information on tophat, open a terminal window and type

```
> tophat -h
```

The raw data is called arm.Xsubset.fastq in your home directory, in folder called course_data arm.Xsubset.fastq

The annotation Data is for chromosome X is drosophilaMelanogaster.X.gtf

The bowtie index is bowtieIndex/drosophilaMelanogaster.X

Note: remember to specify the output directory in the tophat command

```
> mkdir -p ~/course_data_output/arm
```

Note: Specify option `-no-coverage-search` in the tophat command. This will speed things up.

Review questions

- What does tophat do?
- What kind of aligner is tophat?
- What is the “bowtie-index”?
- How can specifying more mismatches with the `-n` option change the output?
- Why supply a gtf file to tophat?
- What are the output files from the tophat aligner?
- Why assemble transcripts with cufflinks?
- What is the gtf file and why do I need it?
- what data does the gtf file contain?
- what does the last column in the gtf file contain?

6.2 Example 2 - smo.2Lsubset.fastq

This dataset contains reads from a mRNA sample after RNAi knockdown of the smo gene, in the drosophila Melanogaster cell line, S2. The arm gene is found on chromosome 2L of the drosophila Melanogaster genome. Here, to save time, we will restrict our analysis to this chromosome, 2L. You should for your own projects, you use all the genome information.

Your task is to align the reads to the 2L chromosome

To get information on tophat, terminal window and type:

```
> tophat -h
```

The raw data is called smo.2Lsubset.fastq data/smo.2Lsubset.fastq

The annotation Data is for chromosome 2L is: geneRef/drosophilaMelanogaster.2L.gtf

The bowtie index is bowtieIndex/drosophilaMelanogaster.2L

Note: Specify option `--no-coverage-search` in the tophat command. This will speed things up.

Review questions

- What does the `-p` option in tophat do?
- How does `--no-coverage` search save time?
- What does the `-g` option do?

SAMTOOLS

Website:

- <http://samtools.sourceforge.net/>
- <http://samtools.sourceforge.net/samtools-c.shtml>
- <http://samtools.sourceforge.net/samtools.shtml>

samtools is a set of scripts (a toolbox so to say) that can be used to manipulate and view sam/bam files. In particular you can: sort, index, merge, view these files.

Getting help

To print a list of all the tools available in the samtools suite, simply type samtools on the command line

```
> samtools
```

To print a list of the parameters required and options available with each tool in the suite, simply type samtools followed by the name of the tool on the command line. For example, to get a list of the options available for the “view” tool in the samtools suite, simply type:

```
> samtools view
```

View bam files:

smo.bam is a bam file and is not human readable. To make it human readable you can convert it to a sam file.

```
> samtools view -h -o smo.sam data/smo.bam
```

- What does -h and -o do in the above example?
- Convert arm.bam into a sam file.

Viewing the header of a bam file

In some cases you might only want to see or generate the header of a bam file.

```
> samtools view -H smo.bam
```

Now try:

```
> samtools view -H smo.bam > smo.header.sam
```

- What does the “> smo.header.sam” of the above statement do?
- What information is stored in the header of the sam file?
- From the header of the file, can you tell which alignment program was used to generate the bam file
- Use the samtools view (and read the help) to view a specific region e.g. all reads mapping to chromosome X

Count the number of alignments in a bam file

```
> samtools view -c smo.bam
```

- Can you use samtools to count the number of alignments above a quality score of 20 in your file?
- What does the quality score of an alignment indicate?
- How many alignments are in the arm.bam above a quality score of 50
- What other flags/filtering options are there?

****Create a bam index ****

Use samtools index to create an index of smo.bam

Get help

```
> samtools index
```

Note the usage of the index command in the samtools toolbox suite.

```
> samtools index smo.bam sm0.bai
```

- What does creating an bam index mean?
- Why would one want to create a bam index?

Sorting the sam file

```
> samtools sort smo.sam
```

- What does -o in the above command do?
- Change the command to sort by read names rather than chromosomal locations.
- Change the above command so that the sorted reads are outputted to smo.sorted.bam

****Merging sam files ****

FL1-1.bam and FL1-2.bam are two technical replicates of the one control sample FL1. We would like to merge these two sam files.

Getting help

```
> samtools merge
```

Note the usage and run the command

```
> samtools merge -h header.sam FL1-merged.bam FL1-1.bam FL1-2.bam
```

- What does -h in the above command do?
- Does this command work? Why not? (TODO: The files are sorted wrongly). Change the command so that the files can be merged.

Get summary statistics

Use *samtools idxstats* to get summary statistics for the aligned file. Use the help.

****Create a fasta file index: ****

```
> samtools faidx dros_BD5.25.fa
```

- What is the benefit of creating an index of a fasta file?

****Create a pileup ****

samtools mpileup is a very useful utility for calling variants in alignment files. Read the help documentation carefully.


```
> samtools mpileup -g -l intervalFile.bed -I -D -q 20 -f dros_BD5.25.fa dnaSeq1.bam dnaSeq2.bam
```

Note: dros_BD5.25.fa needs to be indexed otherwise the above command will not work

7.1 Getting bored? Good to know:

Working with the stream You can take the output of one command from the “standard stream” and pipe it as input to an(other) samtools command.

```
> samtools view -u dnaSeq1.bam chr1 | samtools pileup -cf dros_BD5.25.fa -
```

Use samtools and awk to count the number of mapped reads in your file.

```
> samtools idxstats smo.bam | awk '{s+=$3} END {print s}'
```

- Is this the same number as with `samtools view -c smo.bam`
- What is the awk command doing in the above?

ESTIMATING EXPRESSION

8.1 Example - Expression in the arm gene knockdown sample

We have used reads sequences obtained from a knockdown of the *arm* and *smo* gene in the drosophila S2 cell line to estimate gene expression. We used tophat to align the reads. We will now use the output from the tophat step, to look at gene expression in these samples. We will use a program called **Cufflinks** to do this.

Let's first, take the arm gene knockdown sample and assemble expressed genes and transcripts in the ARM knockdown with CUFLINKS

To get help

```
cufflinks -h
```

Note the output of them help command

```
Usage: cufflinks [options]* <aligned_reads.(sam/bam)>
```

The type and position of *required* arguments to the cufflinks programs and in angular brackets "<>", while *optional* arguments are in square "[]" brackets.

The output of tophat in step 1 can be found in your home directory at ARM-1_tophatOutput/accepted_hits.bam

The annotation Data is geneRef/dros_BD5.25.gff

Run the command:

```
cufflinks -G geneRef/dros.gtf --upper-quartile-norm --compatible-hits-norm -p 2 -o ARM-1_Cufflin
```

8.2 Review Questions:

- Why assemble transcripts with cufflinks?
- What do the options in the cufflinks command do?
- What is the gtf file and why do I need it?
- what data does the gtf file contain?
- what does the last column in the gtf file contain?
- what is the "tss_id" tag in the gtf file and what is it used for?
- why is both transcript and gene information in the gtf file?
- What files does cufflinks output and what do the columns mean?

Next, we must estimate the expression in the smo gene knockdown:

8.3 Task

Assemble expressed genes and transcripts in the SMO knockdown with CUFFLINKS (similar to above)

The output of tophat in step 1 can be found in your home directory at
SMO-1_tophatOutput/accepted_hits.bam

ESTIMATING DIFFERENTIAL EXPRESSION

9.1 Example - Estimating significance of differential expression in the ARM gene knockdown compared to control.

We have estimated gene expression in SMO and ARM gene knockdowns. We would now like to see if this is expressed differently as one would find in a control. We will now use the output from the cufflinks step, to look at differential gene expression by comparing it to a control. We will use a program called Cuffdiff to do this.

(To save time we have aligned and pre-calculated gene expression in the control samples)

There is 1 replicate of the ARM sample: ARM-1 There are 2 replicates of the CTRL sample: FL1-1, FL2-1

Use the drosophila gtf file `geneRef/dros_BD5.25.gff`

Use the `accepted_hits.bam` output from the tophat alignment

For ctrl these are:

- `FL1-1_tophatOutput/accepted_hits.bam`
- `FL2-1_tophatOutput/accepted_hits.bam`

And for ARM these are: `ARM-1_tophatOutput/accepted_hits.bam`

```
cuffdiff -o ARM_vs_CTRL_diffOut -b genomeRef/dros_BD5.25.fa -p 8 -L FL1_ctrl,arm -u geneRef/dros
```

Review Questions:

- What normalization strategies are available with cuffdiff?
- What is the -L option for?
- Why are replicates important here?
- What are the output files from cuff_diff?

9.2 Use your linux know how

- How many significantly differentially expressed genes are there?
- Can you find the expression of gene “CG7224”?

9.3 Task

Use your know-how to estimate estimate differential expression in the smo knockdown compared to control.

The output can be found at: `SMO-1_tophatOutput/accepted_hits.bam`

The controls and other input files are as before.

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*