Introduction to the Linux Commandline for NGS Analyses

1.1

Grainne Kerr, Holger Dinkel, Jon Fuller, Matt Betts

September 18, 2013

CONTENTS

CHAPTER

ONE

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 tought 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 (especially 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 ("/").

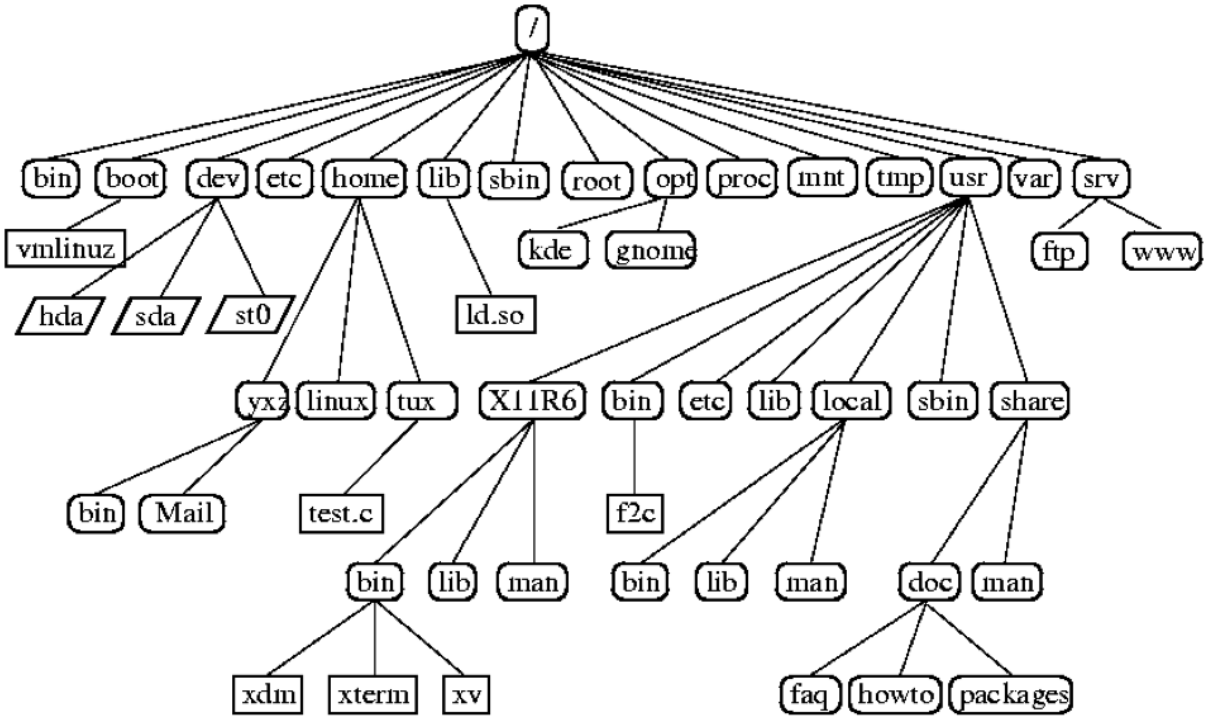


Figure 1.1: Depending on the Linux distribution you might or might not find all of above directories. Most important directories for you are `/bin` and `/usr/bin` (sometimes also `/usr/local/bin`) which contain the user software, `/home` which usually contains the users' homedirectories and `/tmp` which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

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

1.3 General Structure of Linux Commands

Many linux commands have options and accept arguments. Options are a set of switch-like parameters while arguments are usually free text input (such as a filename).

Commandline options (sometimes called comandline switches) commonly have one of the two following forms: The short form `-s` (just a single 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:

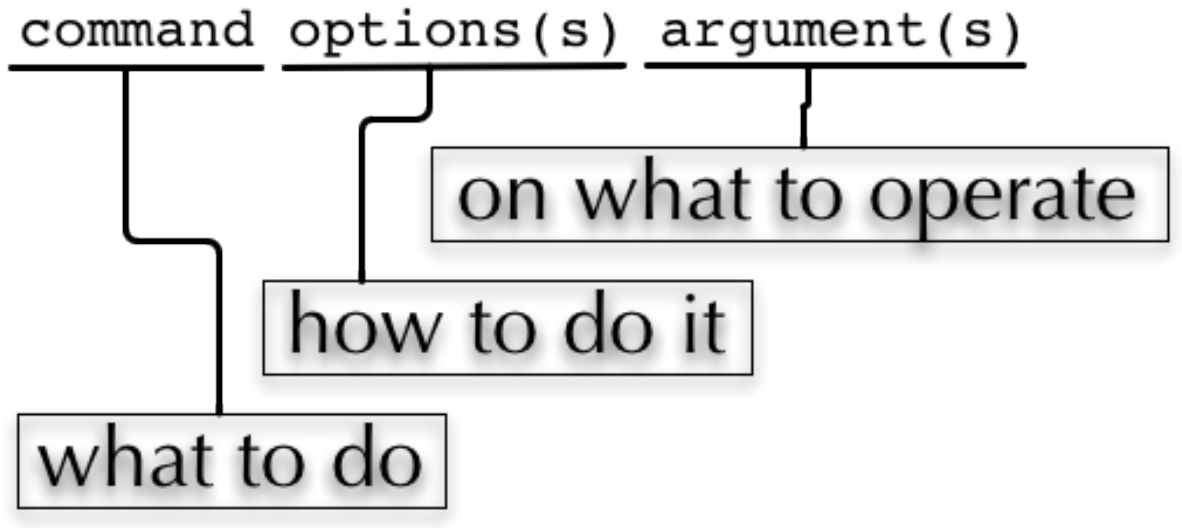


Figure 1.2: General structure of Linux commands.

```
> 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
```

1.4 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.

1.4.1 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 incorrect commandline options

man - show the manual page of a command

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

---

apropos – list manpages containing a keyword in their description

Usage: `apropos keyword`

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

Use apropos to find candidates for specific tasks

/usr/share/doc

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

1.4.2 Who am I, where am I

whoami – Print your username

Usage: `whoami`

```
> whoami
bg_36student
```

hostname – Print the name of the computer

Usage: `hostname`

```
> hostname
pc-teach01
```

pwd – Print the current working directory

Usage: `pwd`

```
> pwd
/home/bg_36student
```

date – Print current date and time

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.

1.4.3 Moving Around

cd – Change the working directory

Usage: `cd [new_directory]`

```
# pwd
/home/bg_36student
# 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 paths (starting with "/") and relative paths (starting with a directory name)

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

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

1.4.4 See What's Around

ls - List directory contents

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

```
> ls
/home/bg_36student
> ls -l /bin/date
-rwxr-xr-x 1 root root 54920 Dec 18  2012 /bin/date
```

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)

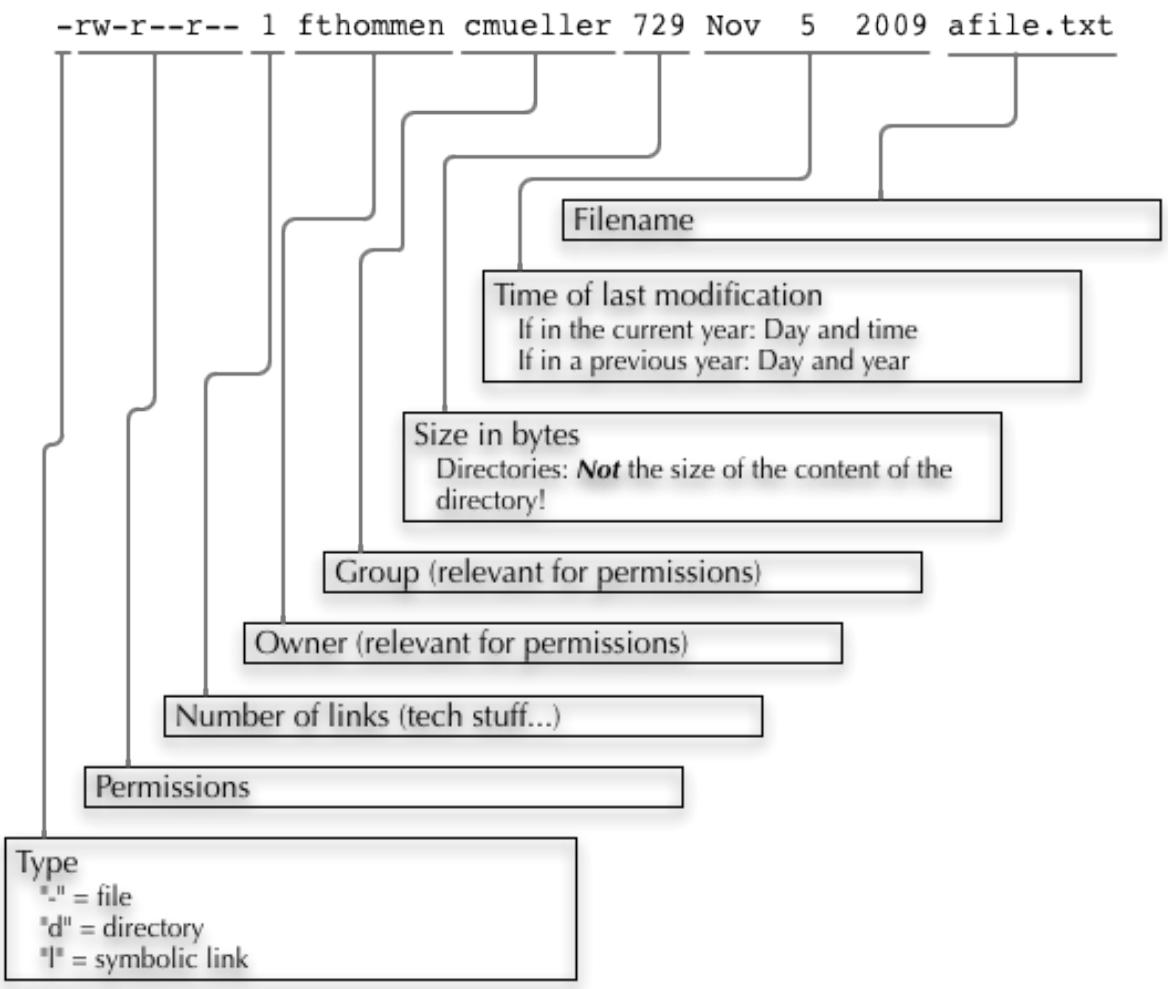
Digression: Shell globs

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"


-rw-r--r-- 1 fthommen cmueller 729 Nov 5 2009 afile.txt
Filename
Time of last modification
If in the current year: Day and time
If in a previous year: Day and year
Size in bytes
Directories: Not the size of the content of the directory!
Group (relevant for permissions)
Owner (relevant for permissions)
Number of links (tech stuff...)
Permissions
Type
"-" = file
"d" = directory
"l" = symbolic link

- `*.???`: All files having a three-character extension

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

1.4.5 Organize Files and Folders

touch – Create a file or change last modification date of an existing file

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

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

```
> ls -l ~/exercises/P12931.txt
-rw-r--r-- 1 dinkel gibson  53K Aug 11 14:21 P12931.txt
> touch ~/exercises/P12931.txt
> ls -l ~/exercises/P12931.txt
-rw-r--r-- 1 dinkel gibson  53K Sep 18 19:16 P12931.txt
```

rm – Remove files and directories

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

---

mv – Move and rename files and folders

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

---

mkdir – Create a new directory

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
```

rmdir – Remove an empty directory

Usage: `rmdir directory`

```
> rmdir adir/
```

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

---

cp – Copy files and folders

Usage: `cp [options] sourcefile destinationfile`

```
> 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

1.4.6 View Files

cat – Print files on terminal (concatenate)

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

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

less – View and navigate files

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

1.4.7 Extracting Informations from Files

grep – Find lines matching a pattern in textfiles

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 – Print first lines of a textfile

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)

tail – Print last lines of a textfile

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)

1.4.8 Useful Filetools

file – determine the filetype

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

---

which – find a (executable) command

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

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

find – search/find files in any given directory

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
```

1.4.9 Useful Terminal Tools

clear – Clear the "screen"

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:

reset – Reset your terminal

Usage: `reset [options]`

```
> reset
```

1.4.10 Permissions

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
```

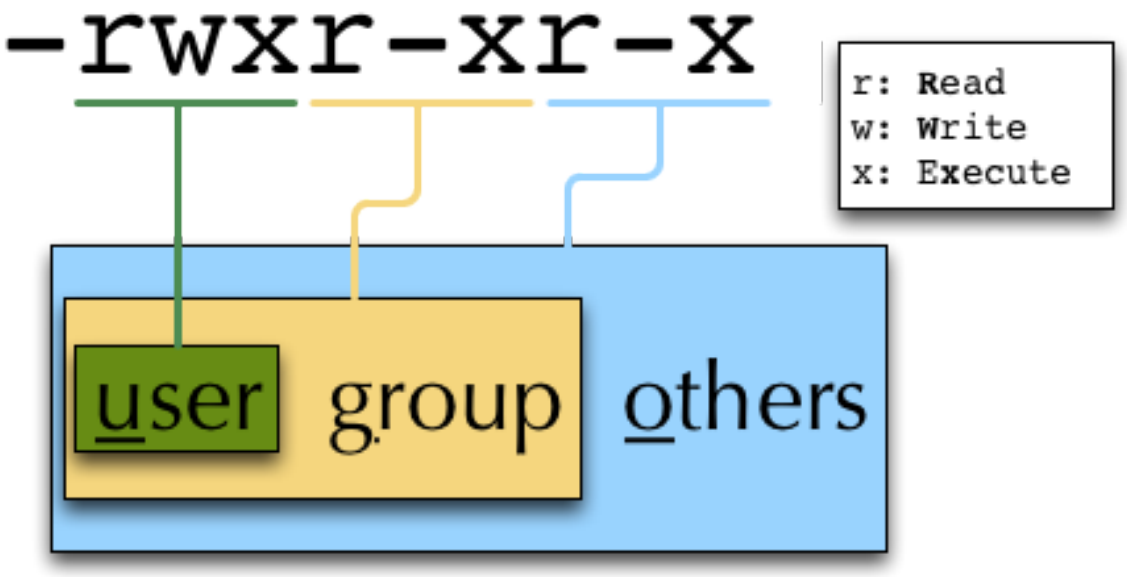
Changing Permissions

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				

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/
```

1.4.11 Remote access

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 and to get a feeling for your environment!

To use a different username, you can use either:

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

or

```
> ssh username@remote_server
```

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

```
> ssh cln038
The authenticity of host 'cln038 (129.296.243.53)' 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
```

If setup correctly, you can even use **graphical tools** from the remote server on the local machine. For this to work, you need to start the ssh session with the `-X` parameter:

```
> ssh -X remote_server
```

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
```

1.4.12 IO and Redirections

Redirect

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

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

```
> date >> file_containing_date
```

Pipe

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
```

1.4.13 Environment Variables

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

$HOME

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

$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).

$PAGER

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

$PWD

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

Displaying environment variables

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

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

Setting an environment variable

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.

CHAPTER

TWO

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 (in your homedirectory)
2. Change your working directory to this new directory
3. In there, create another 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 IO and Redirections

1. Use `date` in conjuction 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.

CHAPTER

THREE

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

CHAPTER

FOUR

FILES

Raw data:

- data/smo.bam
- data/arm.bam
- data/fl1-1.bam
- data/f11-2.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

CHAPTER

FIVE

BEFORE YOU START

You need to tell your computer where to find the programs we will use:

```
> export PATH=/net/netfile1/ds-russell/linuxForNGS/bin:$PATH
```

All the course data can be found in the following directory:

```
/net/netfile1/ds-russell/linuxForNGS/data
```

Note that you do not have permission to write to this directory.

CHAPTER

SIX

QUALITY CONTROL

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

- rawData/arm.fastq
- rawData/arm.bam
- rawData/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* (page 3)):

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 -o ~/fastqcTest -noextract -f fastq rawData/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?

6.1.1 Task

Use fastqc to generate a report of rawData/smo.fastq. Save the report in an output directory called fastqcSMO

6.2 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 try to create a html report file in the same directory as the input file.

```
> samstat rawData/arm.bam
```

The above command will not work as written. Can you figure out why and fix it?

6.2.1 Task

Use samstat to create html report for `rawData/arm.fastq` and `rawData/FL1-1.sam`

CHAPTER

SEVEN

ALIGNING RNASEQ READS

7.1 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 course folder called rawData `arm.Xsubset.fastq`

The annotation Data is for chromosome X is `geneRef/drosophilaMelanogaster.X.gtf`

The bowtie index is `bowtieIndexes/drosophilaMelanogaster.X`

Note: remember to specify the output directory in the tophat command. Create this output directory first with the following command.

```
> mkdir -p ~/arm_tohatOutput
```

Run the following commandÖ

```
> tophat --no-coverage-search -G geneRef/drosophila.2Lsubset.gtf -o ~/arm_tohatOutp
```

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?
- option –no-coverage-search in the tophat command will speed things up. Why?
- 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?

7.2 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 `rawData/smo.2Lsubset.fastq`

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

The bowtie index is `bowtieIndex/drosophilaMelanogaster.2L`

Review questions

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

CHAPTER

EIGHT

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 rawData/smo.bam
```

- What does -h and -o do in the above example?
- Convert arm.bam into a sam file.
- Use the samtools view (and read the help) to view a specific region e.g. all reads mapping to chromosome X

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 rawData/smo.bam
```

Now try:

```
> samtools view -H rawData/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

Count the number of alignments in a bam file

```
> samtools view -c rawData/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 rawData/smo.bam smo.bai
```

- Does the above command work as written? If not, what do you need to change?
- What does creating an bam index mean?
- Why would one want to create a bam index?

Sorting the sam file

```
> samtools sort rawData/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 rawData/FL1-1.bam rawData/FL1-2.bam
```

- What does -h in the above command do?
- Does this command work? Why not? 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 genome/dros_BDGP5.25.fa
```

- What is the benefit of creating an index of a fasta file?
- Does this command work? Why not?

**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 genome/dros_BDGP5.25.fa ra
```

Note: dros_BDGP5.25.fa needs to be indexed otherwise the above command will not work

8.1 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 rawData/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?

CHAPTER

NINE

ESTIMATING EXPRESSION

9.1 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 CUFFLINKS

To get help

```
cufflinks -h
```

Note the output of the help command

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

The output of tophat in step 1 can be found in the course dat folder at `ARM-1_tophatOutput/accepted_hits.q20.sam`

The annotation Data is `geneRef/dros_BDGP5.25.gtf`

Run the command:

```
cufflinks -G geneRef/dros_BDGP5.25.gtf --upper-quartile-norm --compatible-hits-norm
```

9.2 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:

9.3 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 the course data folder at `SMO-1_tophatOutput/accepted_hits.bam`

CHAPTER

TEN

ESTIMATING DIFFERENTIAL EXPRESSION

10.1 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_BDGP5.25.gtf`

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_tophatOuput/accepted_hits.bam`

```
cuffdiff -o ~/ARM_vs_CTRL_diffOut -b genome/dros_BDGP5.25.fa -p 8 -L FL1_ctrl,arm -
```

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?

10.2 Use your linux know how

- How many significantly differentially expressed genes are there?

- Can you find the expression of gene “CG7224”?

10.3 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_tophatOuput/accepted_hits.bam`

The controls and other input files are as before.

CHAPTER

ELEVEN

APPENDIX

11.1 Literature

11.1.1 Algorithms

- Bowtie: Langmead B, Trapnell C, Pop M, Salzberg SL. "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome", Genome Biology <http://genomebiology.com/2009/10/3/R25>.
- Tophat: Trapnell C, Pachter L, Salzberg SL, TopHat: discovering splice junctions with RNA-Seq, Bioinformatics, 25(9):1105-1111 <http://bioinformatics.oxfordjournals.org/cgi/content/abstract/btp120>.
- Cufflinks: Trapnell C, Williams BA, Pertea G, Mortazavi AM, Kwan G, van Baren MJ, Salzberg SL, Wold B, Pachter L.Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation <http://dx.doi.org/10.1038/nbt.1621> Nature Biotechnology
- BWA: Li et al " Fast and accurate short read alignment with Burrows-Wheeler transform" http://bioinformatics.oxfordjournals.org/content/25/14/1754.abstract

11.1.2 File Manipulation tools

- Samtools - http://samtools.sourceforge.net/
- Picard - http://picard.sourceforge.net/index.shtml
- FastQC - http://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- HTSeq - http://www-huber.embl.de/users/anders/HTSeq/doc/overview.html

11.1.3 Data Standards

- http://encodeproject.org/ENCODE/dataStandards.html
- Sam format - http://samtools.sourceforge.net/SAM1.pdf
- http://www.ebi.ac.uk/arrayexpress/
- http://www.ebi.ac.uk/ena/

11.1.4 Normalization

- Oshlack A. et al "From RNA-seq reads to differential expression results." Genome Biology
- Robinson and Oshlack "A scaling normalization method for differential expression analysis of RNA-seq data." Genome Biology
- http://www.biomedcentral.com/1471-2105/12/480/abstract - GC content normalization
- http://www.bepress.com/ucbbiostat/paper291/ - GC bias

11.1.5 Differential Expression

- Tarazona et al. Differential expression in RNA-seq: A matter of depth, 2011, Genome research 21 (12) p. 2213-23 <http://genome.cshlp.org/cgi/content/abstract/gr.124321.111v1>
- Anders and Huber, "Differential expression analysis for sequence count data", 2010, Genome Biology, 11 (10) p. R106 <http://genomebiology.com/2010/11/10/R106>
- Robinson et al, "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data", 2010, Bioinformatics (Oxford, England) 26 (1) p. 139-40 < http://bioinformatics.oxfordjournals.org/content/26/1/139.long>

11.2 Linux Resources

11.2.1 Websites

- A full 500 page book about the Linux commandline for free(!): LinuxCommand.org [1]
- Another nice introduction: "A beginner's guide to UNIX/Linux" [2]
- The "commandline starter" chapter of an O'Reilly book: Learning Debian GNU/Linux - Issuing Linux Commands [3]
- A nice introduction to Linux/UNIX file permissions: "chmod Tutorial" [4]
- Linux Cheatsheets [5]
- For the technically interested: Linux Filesystem Hierarchy Standard [6] and Linux Standard Base [7]
- Unix commands applied to bioinformatics [8]
- BioPieces [9]

[1] http://linuxcommand.org/
[2] http://www.mn.uio.no/astro/english/services/it/help/basic-services/linux/guide.html
[3] http://oreilly.com/openbook/debian/book/ch04_01.html
[4] http://catcode.com/teachmod/
[5] http://www.cheat-sheets.org/#Linux
[6] http://www.pathname.com/fhs/
[7] http://www.linuxfoundation.org/collaborate/workgroups/lsb
[8] http://rous.mit.edu/index.php/Unix_commands_applied_to_bioinformatic
[9] http://code.google.com/p/biopieces

11.2.2 Real printed paper books:

- Dietz, M., "Praxiskurs Unix-Shell", O'Reilly (highly recommended!)
- Herold, H., "awk & sed", Addison-Wesley
- Robbins, A., "sed & awk Pocket Reference", O'Reilly
- Robbins, A. and Beebe, N., "Classic Shell Scripting", O'Reilly
- Siever, E. et al., "Linux in a Nutshell", O'Reilly

11.2.3 Live - CDs

A Live-CD is a complete bootable computer operating system which runs in the computer's memory, rather than loading from the hard disk drive. It allows users to experience and evaluate an operating system without installing it or making any changes to the existing operating system on the computer.

Just download an ISO-Image, burn it onto a CD/DVD and insert it into your DVD-Drive to boot your computer with Linux!

Fedora Live CD

This Live CD contains everything the Fedora [10] Linux operating system has to offer and it's everything you need to try out Fedora — you don't have to erase anything on your current system to try it out, and it won't put your files at risk. Take Fedora for a test drive, and if you like it, you can install Fedora directly to your hard drive straight from the Live Media desktop.

Knoppix

Knoppix [11] is an operating system based on Debian designed to be run directly from a CD / DVD or a USB flash drive, one of the first of its kind for any operating system. When starting a program, it is loaded from the removable medium and decompressed into a RAM drive. The decompression is transparent and on-the-fly. More than 1000 software packages are included on the CD edition and more than 2600 are included on the DVD edition. Up to 9 gigabytes can be stored on the DVD in compressed form.

BioKnoppix

Bioknoppix [12] is a customized distribution of Knoppix Linux Live CD. With this distribution you just boot from the CD and you have a fully functional Linux OS with open source applications targeted for the molecular biologist. Beside using RAM, Bioknoppix doesn't touch the host computer, being ideal for demonstrations, molecular biology students, workshops, etc.

[10] http://fedoraproject.org/wiki/FedoraLiveCD
[11] http://knopper.net/knoppix
[12] http://bioknoppix.hpcf.upr.edu

Vigyaan

Vigyaan [13] is an electronic workbench for bioinformatics, computational biology and computational chemistry. It has been designed to meet the needs of both beginners and experts.

BioSlax

BioSLAX [14] is a live CD/DVD suite of bioinformatics tools that has been released by the resource team of the BioInformatics Center (BIC), National University of Singapore (NUS).

Weblinks:

[13] http://www.vigyaancd.org
[14] http://www.bioslax.com

CHAPTER

TWELVE

ACKNOWLEDGEMENTS

Graphic of the *Linux Filesystem* (page 2) taken from the SuSE 9.2 manual © Novell Inc.

All other graphics © Frank Thommen, EMBL Heidelberg, 2012

Course material was compiled by:

- Linux part: Frank Thommen and Holger Dinkel
- NGS part: Grainne Kerr

INDEX