CGAT course 27th April

Navigating the Linux file system – Lucy G

Ctrl-c #aborts current command

Ctrl-z #pauses

Ctrl-l #clears screen

Clrl a #start of line

Ctrl e #at end

Ctrl-u #cuts from start of line

Ctrl k #cut to end of line

Ctrl-r #searchs history

Ctrl-d #logout or exit

Ssh #stands for ssh

X11 #forwarding (graphical interface)

Cat #prints the file contents to standard out

Head -n 10 <file> #prints first 10 lines

Tail #for end (can track end of file

Tail -f <file> #prints only changes to file

Cp file1 file2 #copies the files to a new file

Cp file1 dir/ #copies to directory

Mv file1 file2 #renames the file and can use to move to a direction

Rm \*.perl removed any file ending in prl

Useful recursives

-a #archieve

-I #interactive

-r #recursive

-f #force

-v #verbose

Symbolic link #basically points to a file

Lin -s file1 link1 #creates a symbolic link, ie copies them them to maybe your pwd (literally just says this file is here)

A hard link can’t span hardware drives etc

**Connecting to cgath1 via ssh**

* Sign in with ssh and username@cgat etc
* Type yes to authenticate host
* Password
* Ssh cgath1 or ssh cgath2 to perform work on two hosts
* Have to put yes again to authenticate host
* Password

Setting up so we don’t have to set up pathway every time

In home directory on terminal:

cd ~

cd .ssh #changes directory to this file

#now going to create a key pair

Ssh-keygen -t rsa

Gives you a private and a key pair (id\_rsa and id\_rsa.pub). given same password as comp login

Add key pair to ssh agent

Eval $(ssh -agent -s) #starts ssh agent in background

Config file in nano saying:

Host \*

AddKeysToAgent yes

UseKeychain yes

IdentityFile ~/.ssh/id\_rsa

Ssh-add -K ~.ssh/id\_rsa #add ssh private key to the ssh agent and store passphrase in the keychain

Slide 22.

cp -r /ifs/obds-training/apr20/shared/week1/ /ifs/obds-training/apr20/rose/obds

#error

cp: cannot open `/ifs/obds-training/apr20/shared/week1/bedtools/id\_rsa' for reading: Permission denied

Making a link thing

ln file1 link1 #creates a hard link

ln -s file1 link1 #creates symbolic link

a symbolic link is just a pointer to a file, it isn’t frozen and updates with

touch creates a new file

if you want to copy something to the home directory

cp location/filename .

Shell configuration

.bashrc – runs automatically everytime we log in. located in home directory but hidden

Can set up environment pathwaysa dna aliases and conda environments and pathways etc, shortcuts etc

Variables – access using $ notation

Can do echo $PATH to give list of locations in file tree

Alias rm=”rm -I” #prompts before every removal

Can add things like this into .bashrc, basically personalises your approach, improves effiency

Alias obds=”cd /ifs/obds-training && pwd ls -lh” #Change to obds directory, print pwd and list contents. So when you print obds it does that

.inputrc #changes what keys do on your keyboard

Changes things so you can move like one word forward or one word back rather than single characters

Alt right or left works to move a word at a time right or left

History #shows all the recent commands of the session

I saved all of the text in this current session to 27April20\_Linux1.txt

We got to the end of exercises 2 (shell configuration)

28th April – linux continues

man <command> #for command manual

Wc #word count

Wc -m <file> #counts characters

-l #lines

Find ~ -type f -name “\*.JPG” #searches home directory for files (specified with f) with .JPG

Find . -name “\*.tsv” -exec wc -l {} \; #Find files in current directory, then execute wc and the {} specifies where to put the current file name in the command. \; finishes the command

Diff file1.txt file2.txt #finds differences between files, ie look for code differences, will show which lines have been modified, outputs altered with -c and -u

#File compression – good to compress non-trivial text files, save harddrive space

#Different options on slide 32

Zless <file1> #Can look inside a compressed file

Tar cf file.tar files #archive files using this command, file.tar is name, files is the files to include

#File transfers, ssh, scp and SFTP through FileZilla etc

Wget curl #get files from internet or server (use separately)

Checksums – check that we’ve properly transferred files across systems, hashes across files.

Md5sum file1.txt file2.txt > hashes.txt #calculates and stores hash values for files and then can use this to check file transfer ok

Hashes.txt and files need to be in same directory for this to work

Md5sum –- check hashes.txt #check files post transfer

I had to install md5sha1sum using home brew

Brew install md5sha1sum #install md5 so can use the md5 check thing

File properties and permissions

Ls -l #extra properties on the file, the black is the

Octal notation numbers are the permission values, add up the number of permission

Gives you the user permissions of the user, group and others (this is because the read, write and execute permissions can never add up different values

Chmod #change file permissions, dash means deny permission

Chmod go-rwx <file> #restricts access to group and others to – (which means no access) for read write or x change

Chmod 751 file1.txt #will give user all permissions, will give group read and change and just change for the other users

Exercise 4.

Ls -a -l #To list contents of directory

chmod g+rwx .bashrc #adds rwx permissions to a file

chmod 770 .inputrc #changes permissions to rwx for group and user but leaves others blank

Linux process, either in the background or foreground

Ps #gives a status of progress

Kill <PID> #kill a process by the Id number of the process

Nohup command #keeps commands going regardless of a remote session dropping

Managing system resources

Top #gives info about system you’re on, ie any tasks atm (tells about how much memory etc in top 4 lines – stats of resource lines)

Can change priority of jobs – don’t do this on shared servers.

Htop #user interface, colours give more info

56 cores on the system, bars show how busy each core is on the system

Df #check the hd space that’s free

Du #shows how much usage your directory has

Free #how much RAM on system

Sort by %mem = m when top is open (capital)

Load average at the top of top gives you an idea of how long your job is likely to be waiting

Can set memory requirements/cores so it runs quicker

Manipulating files on the linux command line: next lecture slides

**Linux streams**

Pipes | #connect stdin to stdout

Using streams to move data from one place to another

C < f #directs f to c

C > f #directs c to f

C >> f #appends stdout of command 1 to file

Command1 && command2 #runs command 2 if command 1 works

**Loops**

Slide 6

For the values of 1-5 do this command and then echo the number our

Bottom example, for I between 1-10 (in 2 increments) do this command, then close loop with “done”

So use to complete with all files with \*.txt

Useful if you want to say make a symbolic link for all files ending in \*.txt at the same time

**Regular expression**

[www.regexe.com](http://www.regexe.com) – good test for regular expression

[bracket] #gets anything that contains letters within bracket

adding a ^ matches anything that isn’t that

[abc]{1} #I want to match exactly 1 a b or c

t$ #if you want everything that ends in a t

grep searches for text that matches a pattern

grep -I “exception” file #prints lines containing exception in the file, ignoring case

-v “#” file #invert everything – returns lines that DON’T contain a # (ie just code)

| #means or

\b #before the end of the word

Slide 12

- Example 2 – whats in “” will be what we’re looking for, selects things that start with a-g, then there is anything with a-z and any number of times (\*) and the ends in t, h or e (| means or) and $ means ends in, in the file1.txt

- example 3, uses the extended reg expression to search for the ENSMUSG000000 specifically then ending with a number between 0-9 then 5 numbers (as defined with {5} at the end \b) in file1.txt

Exercise 1.

1. Wc -l \*.bed
2. for files in \*.bed; do $files wc -l; done
3. grep -c "chr5" cpg.bed > chr5.txt
4. grep -v "chr5" cpg.bed > notchr5.txt
   1. grep "chr5" notchr5.txt #check no chr5
5. grep --extended-regexp "chr[1, 6]{1}\b" cpg.bed > chr16.txt

**Pasting join n cut**

Join #(ie like rbind – give it a key), probably more flexibility in r, good for quick n dirty join but not ideal

Sort #sorts files together, give it a key,

Uniq #can remove duplicate lines but only if the file is sorted

Tr #change from upper to lower case

G #flag, like find and replace

Sed #useful for cleaning up files

At end of example ending in “trim leading whitespaces” need to remove “ ‘ “, true in most cases

P #print line

Last example slide 21, useful for FASTQ sequence

Exercise 2

1. paste sample1.counts sample2.counts > allsample.counts
2. sort -k1,1 sample1.counts > sorted1.counts
   1. sort -k1,1 sample2.counts > sorted2.counts
   2. join sorted1.counts sorted2.counts > sorted.counts
3. cat sorted.counts | tr -s ' ' ',' > sorted.counts.csv
   1. could have used sed here
4. sed '$d' sample2.counts |sed '$d' > sample2new.counts
   1. tail sample2.counts #check last lines of each one
   2. tail sample2new.counts
5. cut -f1 coding\_gene\_region.bed | uniq -c
   1. first bit selects the 1st column with f1 and then uniq c counts the number of occurences
6. sed -i 1d allsample.counts
   1. sed -i '1 i\ Geneid,reads,Geneid,reads' allsample.counts
   2. sed -i '1 i\ Geneid,sample1,sample2' sorted.counts.csv

**AWK**

Pattern #test to perform

{ action } if the test is true, what you want to do. Then name of file

If pattern isn’t there, does something for every line of the file

Patterns

Slide 27

1st example

Pattern is $1 # means column one that is chr1.

If it is chr 1, then print field 2   
  
2nd example

// allow a regular expression

Slide 28

1. Print interval between column 2 and 3
2. Action is print first field, then colon, then print second field, then 3rd field #converts to a IGV file format, chr:start-stop
3. “:” this is a string concatenation

Slide 29.

1. Green Action, for every line, add on value of field 3 minus field 2
2. Yellow, pattern is end – print this sum variable
3. Would tlel you the total value of the intervals between field 2 and 3 in the file

Slide 30

1. FS are tabs or commas
2. \n means new line or RS

Print out lines with less than 4 coloumns

Number of records is greater for or equal to 100 and less than or equal to 150

<https://github.com/stephenturner/oneliners#basic-awk--sed>

Exercise 3

1. awk '{ SUM += $3-$2 } END { print SUM }' coding\_gene\_region.bed
2. awk 'BEGIN { FS= "\t"; OFS ="\t" } { print $1,$2-100,$3+100} ' coding\_gene\_region.bed
   1. Here we want to add 100 onto both column 2 and column 3, but also print column 1
3. awk 'BEGIN { SUM = 0 } { SUM += $3-$2 } END { print SUM }' coding\_gene\_region.bed
   1. Calculate the total number of bases covered by annonations
4. awk 'BEGIN {FS= "\t"; OFS="\t"} { print $1,"Ensembl","Gene",$2,$3,$5,"." $6,".", ".", $4} ' coding\_gene\_region.bed
   1. Convert to GFF formatted file, chr, start, stop, name, need to insert columns between col 1 n 2
   2. File format here: <https://www.ensembl.org/info/website/upload/gff.html>

**Day 3: 29th April HPC with Ewan Mac Mahon**

**Intro to batch systems/SGE**

**Background**

Assuming HPC means high performance computing with sun grid engine

Sun grid engine is a batch queuing system, a way of sharing cpu between users/jobs

* Can give all these things a go on the ccb cluster too, somethings between clusters differ. We are set up on the CGAT training cluster during the course.

HPC – When you need more resources than you have available locally. Things will freeze/take ages etc to run, that’s how you’d know you’d need HPC

Downsides, less convenient, less interactive than local

How does the cluster work/connected?

Separate storage nodes, master node (coordinator). 2 networks. Split between designs of cluster, depending on how computational network is set up.

CGAT cluster designed Data off the storage nodes into the compute nodes rapidly

Other clusters act to move data quickly between nodes, depends how things are set up

**Slide 8 – definitions**

Traditional machine, single CPU, all jobs happen in sequence (one thing at a time), more modern computers can run things in parallel. Divide into chunks to run at same time

Amdahl’s law – how much can be parallelised at all.

How we kind of work out how much work can be paralllelised. Ie how much time you save, efficiency

As you increase processing units, the efficiency increase slows – the increase in speedup is not linear to no.processors (depending on how much can be parallelised)

Useful to give jobs a couple more cores but any increase on that doesn’t really help speed up.

24 CPU cores per node on our cluster, rare that a job would need more cpu nodes than that.

**Slide 12 – when to parallelise?**

If you have 20 input files, easy to parallelise 20 jobs, but harder if you need to do certain parts of job sequentially, ie dependency of one job on another

SMMP – my comp will have multiple cpu, sharing same main memory but individual cache per cpu. Good communications and efficient. Worry about synchronisation – openMP way of coordinating jobs. Will only encounter this really if it’s built into a certain tool.

**Slide 14 – Message passing**

Each CPU has its own memory/RAM (multiple nodes in a compute cluster)

Each CPU/node works on its own lil bit of data, passes a message to other nodes to say yes have done the work, here’s the reply/result (communication across nodeS)

Comes down to Amdahl’s law – message passing isn’t a parallel step as have to do it between steps, if you can speed this up, reduce the %job that isn’t parallelisable

This is all very complex, not sure I understand. Don’t know if I need to

**Slide 15**

Dividing job up into different chunks, what the CGAT pipeline does.

Multiple steps, pipeline code will get the jobs where they need to go to help parallel

Staggered starts – computer will submit multiple batch jobs for you when you submit one run

**Slide 16 – batch systems**

Take load off and execute work ina batch system (ie master host)

Give it a job, leave it to the master to give it to an execution host. Takes off aspects of figuring out which jobs/nodes to run/use

Ie. Could submit 500 jobs and master would stop you monopolising the system

**Slide 18 SGE**

Deva and klyn

Q- why is it called Sun Grid Engine

1. Idea of it being like plugging your job into the mains elec source. Doesn’t work quite as big as this, not quite grid-like – still quite local.
2. Cern etc have huge big grid networks, users may have no idea where data is stored etc
3. Sun because Sun wrote it.
4. Moving to a diff batch system from SGE as Sun now doesn’t exist so SGE hasn’t been updated.

**Slide 19**

Scheduling step, goes through queues (lucky dip pool of jobs, batch system selects jobs) based on priority, how busy system is, what job needs and a scoring system (job priority – who’s job, how long it’s been waiting. CGAT does look at how long a job has been waiting and also uses a fair share algorithm, more you do, less priority you get – ie so you don’t block/monopolise the system. Drops your priority, but if no one else usingsystem then can go for it)

Less the job asks of the system, generally quicker it will run.

**Slide 20**

Good to know your jobs, ie could run interactively first, see how much memory you need then sbmit batch?

I.e run a mini interactive version, downsampling – basically check your code and the job details

**Slide 21**

Run on batch system, monitor cpu usage and monitor within code yourself

**Exercise 1**

Log into cluster, type following commands

Qhost #Overview of hardware on the system. Gives you info on different systems, numbered nodes like cgat005 arebatch system worker nodes. Dashed outs on like load and memory use – currently offline nodes (switched off) but batch knows they exist. Reason is as they’re old nodes, low mem. Cgat016 on the other han has more memtot – load column gives you info on how busy machine is on (0.05 so empty) 12g memuse

Qconf -shgrpl #Shows groups on system. Allhosts means everything. On cgat cluster, don’t get anything as the nodes aren’t grouped

Qconf -sql #show me the queues as a list.

Qconf -sgl

On CBRG, qhost gives you some nodes with 252g and big mem machines have 1000g each.

I just logged into cbrg klyn

On CBRG – can give jobs with bigger memories requirement to the bigmemq

**Slide 27 – Cluster**

Normally just want dedicated, all of the cores will be on a single node. Generally only use the dedicated

Make n mpi (mpi will try and give 4 sltos on different nodes, make will give on one node unless spillover. Gives you a list of nodes)

**Slide 28**

Is the cluster busy?

Qhost -j #lists jobs on machine

Qstat -f -u “\*” #Lists jpobs that are running n qhere they are (-f means full output) (-u means “user”, using “\*” will give all users)

Qstat #on cbrg can just use qstat as they’ve set this up as an alias

Qstat | less #this will pipe the output of qstat into less so it’s more manageable to read

**Slide 30**

Qsub pipeline.py #pipeline will submit batch jobs for you from this pipeline

Interactive sessions

Qrsh #remote shell, login from 1 machine to another but qrsh will schedule login like a batch job. Ie it will find a good node for you to do an interactive heavy R session (eg). This will be useful

Slide 31

Adaptations of qsub to define where to do the job, redirect output etc. Tbf you can add this into your bash script

Better to put this info into the bash script, but its quick n dirty to include in command (pretty sure the command will override scrpt tho)

**Slide 34**

Pe dedicated #will give a certain number of slots if you know your job needs more memory

Skipped 35 on array jobs, he’ll come back to this

**Slide 36**

Submitting this info into the script rather than the command

Lines starting in # are comments that bash will ignore (so will SGE)

But #$ are comments for bash but SGE will see them as commands!

**Exercise2 Slide 38**

Run the commands on the slide (can do both cgat traing

If you ls-l gives you the following:

Gives you serial.error, serial.output

Cat serial.output #gives you which node it used, queue it used and job name, and how many machine it used.

Qsub -l h=cgat016 sge-serial.sh #Will force it to run on that node

Qsub -l mem\_free=2G sge.serial.sh #Says it needs this much memory

Default is that the operating system will choose an output file. But make sure if you specify an output file, you change it each time so you don’t write over the output file from previous job

If you submit a job with more memory than available, it will appear as qw on qstat

Parallel script just runs 4 threads

Qrsh #interactive batch session

Will change which node you’re on (can see with username at start) – each user will be on a diff node

Exit #to logout of qlogin

Can put job requirements on this too

Qrsh -pe dedicated 8 #opens the login session on a node that memory is available (basically reserves the nodes for you and you can just caryr on working normally or submit batch jobs from here)

Can qrsh a particular command ie. Try this and look at user name

Qrsh “echo \$HOSTNAME”

Qdel #delete jobs (you can’t delete other users’s jobs)

Vim #text editor like nano, can quit with :q!

: #command

Q #quit

! # JUST DO IT

**Slide 44-45: DRMA API**

Data structure, that has the info for a qsub command and then make a function for the job submission. Python script available but currently we don’t have the conda environment set up

Portable api across batch systems, but drma gives a site independent way of doing it (I think) DRMA is a good think of communicating between batch systems but can’t gloss over all the site specific batch system things (argh!)

Slides will be available on files tab

**Pm Maanaging your software environment with Conda**

Dependency – packages are dependent on different versions of certain packages, one package may be dependent on v1 dplyr where one package is dependent on v2 dpylr

Package managers – allow automatic download of all the dependcies,

Conda allows use of packages from different programming languages across operating systems

For conda, don’t need administrator privileges so we can use on CBRG.CGAT clusters

Cond is a package/environment manager

Pinned versions – can specify which version of a package you want so it doesn’t update even if you install new packages (conda meta folder)

Condalist - -revisions #what packages were added etc

In Brackets is where the package comes from

Anaconda takes up a lot more diskspace, lots more packages and more packages for basic use

Miniconda is just conda plus dependcies, more control as you can just install specific pathways

**Conda tutorial**

<https://github.com/OBDS-Training/Conda_Workshops/blob/master/1_Conda_intro.md>

qrsh #use a qlogin

cd /ifs/obds-training/apr20/rose/obds/week1/conda #Here is where we’re going to install

curl -o Miniconda.sh <https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh> #conda install script retrieving

bash Miniconda.sh -b -p obds\_conda #run install script

#Then need to add location for conda software to our $PATH variable so we can use it. Where comp looks for executables (tell comp where to find it). Add path for conda to path variable

source /ifs/obds-training/apr20/rose/obds/week1/conda/obds\_conda/etc/profile.d/conda.sh #within this, will have to locate the profile.d

Eduardoo had a problem with the path (, so he had to do this into his .bashrc file

**Echo $PATH #try this to see if it’s been added**

**export PATH=$PATH:/place/with/the/file**

conda activate base #Activates base environment and moves into default conda software environment

which conda #Check the source command worked.

Conda list #packages in base environment

#Use conda to search for software packages to install, but to find pakcages, conda need address of certain internet sites, ie channels. We need to add appropritate conda channels to get software we need/trust. ORDER is important of channels here

Conda config –add channels defaults #this was already in channels list, moved it to the top

Conda config –add channels conda-forge

Conda config –add channels bioconda

#Defaults contains major software packaged for conda by conda

#conda-forge contains general programming packaes packaged for conda by people in programming/computational comm

Bioconda – biology specific programmes packaged for conda by people in computation biology community

Conda info #check channels have been added to installisation

Conda list #check currently installed packages

Conda update –all will update all packages

Conda search fastqc #search for package fastqc (command line) – searches channels, finds bioconda has this and shows all the different versions

Find fastqc #check if you have it installed

Conda list fastqc #check if installed

OR

Which fastqc #search your PATH variables

Conda install fastqc #insstall. Can specify specific package version with fastqc=0.11.7

Conda list fastqc #Check it’s installed

Which fastqc

Fastqc –help

Conda remove fastqc #removing it

Conda install pysam –dry-run #install a package without actually installing it

#Section 3: conda environments

#Aiming ot set up a conda environment with python 3 and an R environment for the OBDs training programme, that contains the software that we will use in the lecture sover next week

**) Setting up your Python 3 environment for the course**

**Python & associated libraries**

* python
* numpy (a python package for doing fast mathematical calculations and manipulations)
* pandas (a python package for making/using dataframes)
* scipy (a collection of python packages for data analysis - includes ipython, pandas etc.)
* matplotlib (a python package for plotting)
* seaborn (a much prettier python package for plotting)
* ggplot (python version of ggplot)
* plotly (interactive and browser-based graphing library for Python)
* scikit-learn (a python package for machine learning)
* pysam (a python package for working with BAM/SAM alignment files)
* pybedtools (a python wrapper for bedtools meaning that you can use bedtools functionality in python scripts)
* ruffus (a python pipelining program that we will use to write pipelines)
* drmaa (for the management of submitting jobs to the cluster)
* cgatcore (a library from cgat to make pipelines usable with a computer cluster)
* spyder (an interactive development environment (IDE) for python similar to RStudio)
* jupyter (interactive notebooks for python)

**Bioinformatics software**

* fastqc (QC of FASTQ raw sequence files)
* multiqc (collects summary statistics from other bioinformatic programs)
* trimmomatic (read trimming tool)
* hisat2 (quick read aligner (mapper) for spliced sequencing reads)
* bowtie2 (slower read aligner for unspliced sequencing reads)
* kallisto (alignment-free RNA quantification tool)
* samtools (manipulate BAM/SAM alignment files)
* bedtools (comparison, manipulation and annotation of genomic features)
* picard (QC of alignment files)
* subread (counting of reads in features)

conda env -h #get help about conda environments

conda env list #list of existing environments

Using files obds\_py3.yml and obds\_r.yml to create 2 new conda environmets, within the obds py file, can see the cnhannels, dependencies hat we need to make the environment. Don’t need all the version numbers of the packages

Conda env create -f obds\_py3.yml #uses this file to create a new conda environment, collects package metadata. This job takes quite a while

Conda env create -n python\_env -f obds\_py3.yml #name the environment

Conda activate obds\_py3 #activate new conda environment

Python –version #check version of python in the environment (3.6.10)

Useful to be able to export environemtns

Conda env export -n obds-py3

Conda env export -n obds-py3 < my\_environmet.yml #directs output to a file

Conda env create -f obds\_r.yml #create r environment using the yml package

Conda activate obds-r

Conda list

#Had to source conda to enable it’s use and then made the following alias to enable me to just literally put obds-py3/obds-r to activate the conda environment

source /ifs/obds-training/apr20/rose/obds/week1/conda/obds\_conda/etc/profile.d/$

alias obds-py3='conda activate base && conda activate obds-py3'

alias obds-r='conda activate base && conda activate obds-r'

Section4. Installing conda/creating a new environment on local machine

<https://docs.anaconda.com/anaconda/install/uninstall/>

rm -rf ~/anaconda3 #Remvoing anaconda using above from home directory – this took ages too

nano .bash\_profile #In ~ checking bash for Paths for anaconda – I did have this. I had to remove all the init conda stuff using ctrl k.

source .bash\_profile #to reset the source

curl -o Miniconda.sh https://repo.anaconda.com/miniconda/Miniconda3-latest-MacOSX-x86\_64.sh

bash Miniconda.sh -b -p obds\_conda

configured the channels as before

scp rose@cgatui.imm.ox.ac.uk:/ifs/obds-training/apr20/rose/obds/week1/conda/obds\_py\_mac.yml .

conda env create -n obds-py3 -f obds\_py\_mac.yml

source /Users/rhodgson/conda/obds\_conda/etc/profile.d/conda.sh #Tells my comp where conda is

alias obds-py3='conda activate base && conda activate obds-py3'#

**30April20**

**AM: Looking at different genomic file formats like BAM or fastq**

**Fastq files**

Raw sequencing files, 4 line blocks containing sequence id(unique id for each sequence), nucleotide sequence, padding row and then ascii characters are quality score for each base.

Illumina/Phred are quality scores, using a diff character for each score and base.

Each base has its own quality – confidence of the sequencer that the base is correctly guessed. One image on base added, software looks at the image for each base on each fragment, gives you an idea of how confident the software is at guessing the base

In bam file, the forward and reverse are together but with fastq will give you a forward and reverse sequence in different files

**FASTQC – read quality control**

Per base quality score (0-40). Can look at the error probability, green zone – all above 30, red is bad QC score – can trim or remove these sequences

Typical for illumina sequencing to drop off slightly at 3’ end.

On right need to trim, may meet Minimum length threshold may be met so may need to just get rid of this read (may indiciate poor quality RNA or sequencing). If you go below 20, mapping is difficult

Minimum reads – try and get 25 million paired end reads per sample but if you need to look at rare genes, more genes expressed more runs you need

Need samples to have even coverage but if one sample has vastly difference depth on read count or quality – need to consider downsampling if higher reads etc

Asked about the submission of sequencing units – can just kind of see what the depth is and then add more if you haven’t reached that

Average sequence quality – on right can see hump of the sequences with low quality. Either trim based on QC score, and discard below min length – trim Galore etc

Adaptor contamination – if poor quality or not enough sequence, will read into the adaptor – get adaptor contamination

GC content – should see a normal distribution, don’t read too much into this

Don’t worry if a couple of sections are red

**SAM**

Contains alignment/mappin info to genome

@ means header

@HD tells you sam format used

@SQ line for each chromosome in genome – seq name and length

1 alignment of a read – name for alignment,

Some aligners don’t give a mapping quality score

Cigar tells you about matches and deletions in genome – match and mismatch both give an M

7 Name of mate/pair of sequence, = means matched to same chromosome

89 posiiton and length (info not available in this example)

Aligner may not output some if this info

SEQUENCE then quality score based on phred

SAM flags (col 2)

Look at the broad insititute website to understand what flags mean what – first in a pair means it’s the forward mapping read. \*\*

BAM is compressed SAM file (SAM files are big) Sorted by index

CRAM also compression of SAM, but not used as much

Samtools – interaction with these SAM files, worked on linux

**Exercise 1**

<http://jkbonfield.github.io/www.htslib.org/doc/1.2/samtools.html>

<http://quinlanlab.org/tutorials/samtools/samtools.html>

<http://www.htslib.org/doc/samtools-view.html>

Load samtools – we’re in conda environment

samtools view example.sam | head

How to specify output file name

samtools view -S -b example.sam > example.bam

Lucy says can tell if it’s a bam or a sam because less works for a sam but not bam

samtools sort example.bam -o example.sorted.bam

samtools index example.sorted.bam #Creates the index file

Keep bam and index in same folder

The bai is the index for the bam file, it helps accessing the bam file for different software (just like a contents page)

View header

samtools view -H example.bam | head -n 100

Q. View alignments from example.bam on chr7

samtools view example.sorted.bam chr7 | head -n 100 #Region you want to search for at the end of command

Bed file formats

GFF specifies where the genes are in the genome, gives you feature and then where the feature starts and stops in the genome. GTF files is the annontation to tell you what it is each position in genome

GFF and GTF are similar

Bedtools for bed files comparing gene regions between files, want to look at chip seq peaks between two sets of cell types. Merge regions too –

Wiggle – website at bottom is useful

Big wig – compressed version. UCSC tools- conversions

Bedtools online tutorials in exercise 2 (have a go ourselves)

**RNAseq analysis**

Pseudo alignment very fast – is it as accurate?

Workflow – today going to the count tables,

Question – how consistent is the processing between pre-processing to counts table. Lots of different ways to do the same thing, different programs or formats.

How does this alter the results? Comparison reviews, ie between STAR etc other mappers

High correlation between packages

**RNAseq group exercise – workflow and tools**

We’ll do 1 sample each so and will merge count tables at end

BAMs or FastQ files provided in supp info normally

**Exercise – RNAseq worklow**

First need to download data & verify download:

Ø European Nucleotide Archive PRJEB18572

<https://www.ebi.ac.uk/ena/data/view/PRJEB18572>

Each line, get the copy link address

Would download to /ifs/obds-training/apr20/rose/obds/linux-rnaseq  
To download: can use wget or curl

On the website take FASTQ files (FTP) or click the TEXT file on page, then on excel spreadsheet and create a bash script

Wget <ftp link> #ftp link address from copy link address from each file for each one

I did from the shared exercises folder

cp ERR1755087\* /ifs/obds-training/apr20/rose/obds/linux-rnaseq

To verify the download: use the md5 check sums as well – can download this from website by looking at selecting clumn on the left hand side so it’s included as an extra column – this shows you what the file should be like – now verify

From where you copied the files to

md5sum --check md5sum.txt #check the hashes text file

First do fastqc on the fastq files to confirm good read quality etc. Next will need to trim based on what th

fastqc ERR1755087\_1.fastq.gz ERR1755087\_2.fastq.gz -f fastq #run fastqc on the fastq files

Could have sped this up using -t which is threads parallelising

Nohup & #keeps the command running in the background even terminal disconnects etc

Move the files locally, I used scp to transfer to OBDS training course and new directory FASTQC

Check out the fastqc html files – do they need trimming?

If you have plenty of reads, go for it. But if not, leave if quality is ok. Can adapt the trim settings – default settings based on the quality scores, but potentially if poor data then maybe don’t trim as much off n just have a go at mapping

multiQC

multiqc .#makes a nice html file that gives you a summary of the fastqc results from both files. This command searches the directory that you’re in

**Trimming**

Use trimmomatic

Don’t need java thing as we’re installed in conda environ, so they know where to find it already

<http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf>

Useful to do commands like this on a downsampled amount of data to check the command does what you want it to

zless ERR1755087\_1.fastq.gz | head -n 1000 > ERR1755087\_1DS.fastq #pipe it to a new fastq file, downsamples

zless ERR1755087\_2.fastq.gz | head -n 1000 > ERR1755087\_2DS.fastq #do number 2

wc -l ERR1755087\_2.fastq.gz #check line count of the new file to make sure you’ve got it right

Test on downsampled data

<http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/TrimmomaticManual_V0.32.pdf>

trimmomatic PE -phred33 ERR1755087\_1DS.fastq ERR1755087\_2DS.fastq -baseout ERR1755087\_DS.fastq ILLUMINACLIP:TruSeq3-PE-2.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36 #the trimmy bit

Should then have to do this for the bigger fastqc files.

Check the length of the reads (typical is 50/75bp paired end) look at fastqc files to check this, or alternatively just do zless on the original fastq file and just manually count the sequence length

trimmomatic PE -threads 4 -phred33 -trimlog trimlogreal.txt -summary statssummarytrimreal.txt ERR1755087\_1.fastq.gz ERR1755087\_2.fastq.gz -baseout ERR1755087\_.fastq ILLUMINACLIP:TruSeq3-PE-2.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36 #added the trimlog and a summary results section which give nice info on how many reads are trimmed etc. Make sure your 1 and 2 are in correct format, so forward first and reverse 2nd.

Checking the proper output

After trimming, should do fastqc again to confirm that your adapter sequence/contam has been removed and have you removed enough poor quality sequences

**Mapping to genome**

HISAT2 <http://daehwankimlab.github.io/hisat2/manual/>

<https://github.com/griffithlab/rnaseq_tutorial/blob/master/manuscript/supplementary_tables/supplementary_table_5.md>

hisat2 --rna-strandness RF --threads 12 -x /ifs/mirror/genomes/hisat2/mm10 -1 ERR1755087\_DS\_1P.fastq -2 ERR1755087\_DS\_2P.fastq -S ERR1755087DS.sam --summary-file hisat-sum.txt

#This is the command to run it on the downsampled bit, the threads has to be the right one. Can then add this to the multiseq info

Now need to index/sort the sam filed and convert to bam

samtools view -S -b ERR1755087DS.sam > ERR1755087DS.bam #converts to bam

samtools sort ERR1755087DS.bam -o ERR1755087DS.sorted.bam #sorts it

samtools index ERR1755087DS.sorted.bam #indexes

samtools idxstats ERR1755087DS.sorted.bam > idxstats.txt #reports alignment summary stats to a text file

samtools flagstat ERR1755087DS.bam > flagstat.txt #counts the number of alignments for each flag type

For real data:

hisat2 --rna-strandness RF --threads 12 -x /ifs/mirror/genomes/hisat2/mm10 -1 ERR1755087\_\_1P.fastq -2 ERR1755087\_\_2P.fastq -S ERR1755087.sam --summary-file hisat-sum.txt #this command runs the full thing

<https://github.com/sansomlab/scseq/blob/master/pipelines/pipeline_scrnaseq.py#L282>

translates what each tool uses for stranded

samtools view -S -b ERR1755087.sam > ERR1755087.bam

samtools sort ERR1755087.bam -o ERR1755087.sorted.bam

samtools index ERR1755087.sorted.bam

samtools idxstats ERR1755087.sorted.bam > idxstats.txt #reports alignment summary stats to a text file

samtools flagstat ERR1755087.bam > flagstat.txt #counts the number of alignments for each flag type

Never run anything on the index file

<http://www.htslib.org/doc/samtools-idxstats.html>

Subread <http://bioinf.wehi.edu.au/subread-package/SubreadUsersGuide.pdf>

cp /ifs/obds-training/apr20/rose/obds/week1/rnaseq/genes.gtf.gz .#copy over the gtf file

featureCounts -p -a genes.gtf.gz -s 2 -o Counts.txt ERR1755087.sorted.bam #This generates our counts table (OMG!)

-p #means paired end

-s 2 #stranded ness – this is which strand the original strand of sequencing came off, so we need to put in 2 as this means reversely stranded . This command depends on the seq kit

Friday 1st May. Version control – GitHub

Hard to keep tabs – track changes with notes, explanations etc

Atrribute ownership to individual changes

Version control, record changes to set of files at a particular time, allows you to revert back to original code

Make a repository folder (ie on server – local reps)

Remote repository is on github

-p #means you can make more than one folder (within another folder)

mkdir -p devel/obds-training #make more than one folder

/ifs/obds-training/apr20/rose/devel/obds-training

Which git #check which one is available

Git init #initialises repository production – now everything in this folder will be in the repos . Only stuff from this folder will be in this git thing

Ls -al #now see a git folder, you have initialised your git repos

Git status #shows us branches.

Branch master is where the main changes you make are stored ie history. Master is main branch

Could make another folder in devel and this can be a new repository – for different research projects

Devel is a standard folder name where people store code.

**Configuring git**

git config --global user.name "rosehodgson"

git config --global user.email "rose.hodgson@sjc.ox.ac.uk"

git config --global core.editor nano

git config --global –list #gives the following info

user.name=rosehodgson

user.email=rose.hodgson@sjc.ox.ac.uk

core.editor=nano

nano 01May20.txt #make file

git status #checks whether it’s tracking that file and if you’ve changed anything

git add 01May20.txt #adds the file to tracking

git commit -m "I added second line" #will commit this – saving the state of the file at that particular moment

git log #info on the commit stuff – got a 40 number

git log --oneline #gives you a shorter version

Commit history – git log sha-1 40 character code, like a hash – unique number specifying the file again. Short version gives short version

Git diff #gives you differences between difference commits

Tag – we saying it’s an important version , ie version 1 or 2 of a package. Tagging can be useful for this

Changing files

Have to re-add each document every time – when you commit they wont be added to git if you don’t add it each time

Git head – the head points to the master brance

Git branch #shows you just have one branch and it is the master – head is the pointer to the current branch/current commit

Branch is a way to work on code without altering master branch – ie make changes, test out code then can merge branches back to master

Ie good for multiperson code – each person makes their own branch, alters it until theyre happy then merges it into the master

Git branch fix-1 #makes a new branch called fix1

Git checkout fix-1#switches to this one so files changed will only be ont hat file

Add n commit

Switch back to master – changes will not be there as on other branch

Git merge fix-1 #merges the branches

Git branch -d fix-1 #deletes the branch

Conflicts – if you both try and merge things – Git doesn’t know which line should be first, then have to manually edit it

Usually people on branches will be working on different parts of the file

Now gonna do github – online repository – where code can be shared between people, each person works locally and then when they want to merge your changes – commited to main packages, then do git push

Git push #push changes to central repository (github)

Git pull #get changes from central to local repositiory, say if someone mkes changes that you need!

Set ssh key so we don’t have to enter password when you want to push to github

Check for exisitin ssh keys

Ls -al ~/.ssh

less ~/.ssh/authorized\_keys

go to id\_rsa.pub instead as in authorized keys is the one from computer

so need to generate a new key pair from cluster to thing (as we don’t have an id\_rsa.pub on cluster)

New key:

ssh-keygen -t rsa -b 4096 -C [rose.hodgson@sjc.ox.ac.uk](mailto:rose.hodgson@sjc.ox.ac.uk) #the number allows you to set your own

eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id\_rsa

<https://help.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent#adding-your-ssh-key-to-the-ssh-agent>

<https://serverfault.com/questions/253313/ssh-returns-bad-owner-or-permissions-on-ssh-config>

Make sure you use the linux instructions – the config file doesn’t apply to linux only to mac

ssh -T [git@github.com](mailto:git@github.com) #connect to github

The authenticity of host 'github.com (140.82.118.3)' can't be established.

RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'github.com,140.82.118.3' (RSA) to the list of known hosts.

Hi rodgson543! You've successfully authenticated, but GitHub does not provide shell access.

Adding the repository to github

Git remote -v

git remote add origin [git@github.com:rodgson543/obds-training.git](mailto:git@github.com:rodgson543/obds-training.git)

git push -u origin master #pushes from local to github

Have to create a new branch

Pull request

* have to push a branch, but to stop anyone pushing shit code – have to create a pull request on gihub so everyone can check it
* Do my work on local branch, ie name it rose – then psuh branch to repository shared wit everyone – go to github and create a pull response

Now going to make a branch

Git pull origin master #makes sure branch is up to date – should be in repository folder to do this

Git checkout -b fix-2 #create/switch to new branch

Git add 01May20.txt

Git commit -m “Moved to branch fix-2”

Git push origin fix-2 #pushes branch to remote

Have to click on compare and pull requests – this creates your pull request but then people would check

Merge and can delete branch

Locally want to up to date:

Git checkout master #switch to master branch.

Git pull origin master #update local

Git branch -a #gives branches of all fo them – wants to get rid of one

Git fetch - -prune #remove any branches on local that have been removed on github

Git branch -d fix-2 #delete local fix-2 branch

<https://www.educative.io/edpresso/how-to-delete-remote-branches-in-git>

This website shows how to delete remote branches from local

Undoing changes

How to undo changes – if you want to go back a change – can use git checkout and the hash based on which commit message you want to go back to – important to comment when you commit changes

Git revert -undo a commit but keeps history of the gits kind of ahead of it – revert back to previous commit but

Git reset – removes the history

Git checkout c1333cf #”detached head state” now looking at another version. If we check the file out can see we’re removed lines

Git checkout master #brings back to current master

Can use this to just look what previous versions were

Git revert HEAD #shows the latest commit

Git revert #multiple commits – comes up with a problem, have to fix it manually by editing the text file, removing the HEAD etc

Might be easier to just git checkout to the version you want

git reset 31bb5d1

git checkout 01May20.txt #moves the head to the commit

git status – branch is behind by 4 commits so your remote is ahead of local so want to push to correct the remote. If you’re sad and want it back can pull from remote to locate

git push -f origin master #force it to sort itself out

what files should you have on git – don’t need to track html files generated from code but should track code – or just normal text files

Write what error you fixed in commit messages – so you know what you’ve done wrong and don’t repeat it etc

A readme.md file is what the repository is – project etc

For each repository want a separate folder – own research but don’t put inside others

Clone repository to computer

Git innit – don’t ever initate a repository within a repository

Don’t need git innit if you’re cloning a repos from elsewhere

git clone [git@github.com:OBDS-Training/OBDS\_Training\_Apr\_2020.git](mailto:git@github.com:OBDS-Training/OBDS_Training_Apr_2020.git) #clone a repository thign

cd to folder

git remote -v #connects it

git push origin master

git pull origin master

can now see a test.file in the txt and gonna edit ti

If someone edits the file and trys to push it to github while you’re trying then you’ll have to manually resolve conflicts