# Advanced Bioinformatics (7BBG2016): Practical Bioinformatics Data Skills

**Student ID: 1890093**

## 1. Basic Linux and the command Line (20pts – 15% of final mark, each question provides 1 point)

1.1 What does ./../.. stand for ?  
A. Current directory  
B. Up one directory  
**C. Up two directories x**  
D. None of Above

1.2 What does cd / mean in UNIX? Please explain what the cd command does.

Changes your current directory to the root.

cd allows the user to move between directories on the command line, by specifying a path or using special characters (e.g. ~ to return home).

1.3 What command would you use to get help about the command cp? (please provide an example command)

cp –help

(alternatively, man cp)

1.4 What does the command pwd do?

Prints the path of the current working directory.

1.5 How do you display a listing of file details such as date, size, and access permissions in a given directory? (please provide an example command)

ls -lh

1.6 How do you print on the terminal the first 15 lines of all files ending by .txt? (please provide an example command)

head -n 15 \*.txt

1.7 How do you rename a file from new to old? (please provide an example command)

mv old\_name.txt new\_name.txt

1.8 How do you display the contents of a file myfile.txt? (please provide an example command)

less -S myfile.txt #To display on command line and navigate through it. (-S is optional but can aid interpretability by adding structure.)

nano myfile.txt #To open it in a command line editor (e.g. nano, but can also use vim/emacs etc.)

1.9 How do you create a new directory called flower? (please provide an example command)

mkdir flower

1.10 How do you change the current directory to /usr/local/bin? (please provide an example command)

cd /usr/local/bin

1.11 How can you display a list of all files in the current directory, including the hidden files? (please provide an example command)

ls -a

1.12 What command do you have to use to go to the parent directory? (please provide an example command)

cd ..

1.13 Which command would you use to create a sub-directory in your home directory? (please provide an example)

mkdir ~/subdirectory\_name

1.14 Which command would you use to list the first lines in a text file? (please provide an example)

head filename.txt

1.15 Which command will display the last lines of the text file file1? (please provide an example)

tail file1.txt

1.16 Which command is used to extract a column from a text file? (please provide an example)

cut -c 1

1.17 How do you copy an entire directory structure? E.g. from Project to Project.backup (please provide an example)

cp -r project/ Project.backup/

1.18 How would you search for the string Hypertension at the end of the line in a file called diseases.txt? (please provide an example)

grep ‘Hypertension$’ diseases.txt

1.19 How do you see hidden files in your home directory? (please provide an example)

ls -a ~/

1.20 How do you run a job that will continue running even if you are logged out? (please provide an example)

nohup bash bashscript.sh

*or if applicable, submit job to a Slurm scheduler, e.g.:*

sbatch -p shared bashscript.sh

*or:*

bash bashscript.sh

bg

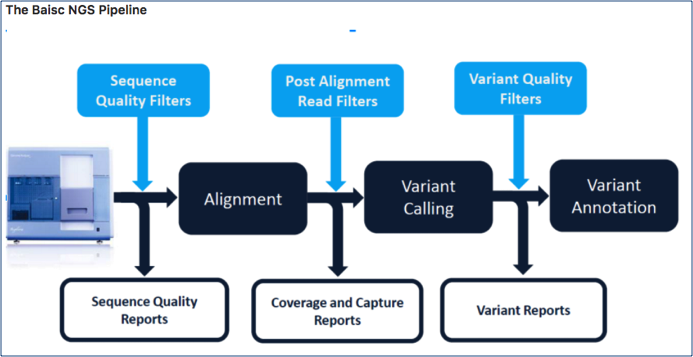
disown

## 2. The NGS Pipeline (65pts – 45% of final mark)

## 2.0 From raw data to alignment and variant calls (20pts)

The assessment is designed to:

* Test your ability to run standard NGS pipeline using the command line on a Linux system.
* Test your ability to create a Bash script that executes your NGS pipeline
* Test your basic knowledge of a standard NGS pipeline.



You have been provided with paired end fastq data and an annotation bed file from an Illumina HiSeq 2500 run. Using the assigned Openstack instance (please contact the module leaders if you have any problems with your Openstack instance), install the necessary tools and execute a standard Bioinformatics NGS pipeline to perform read alignment, variant discovery and annotation as described in the following NGS Pipeline section. **You are required to share a bash script that runs the workflow and takes the provided sequencing data as input (links provided below) with the examiner by uploading it with this report.** Please make sure the bash script lines are adequately commented to provide a clear description of what it is doing. **The script will be evaluated by the examiner and up to 20pts will be given for a fully running and easy to read script.** Based on your pipeline, provide the following information and answer each question.

**Fastq Read 1 (~750MB):** <https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R1.fastq.qz>

**Fastq Read 2 (~750MB):**<https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R2.fastq.qz>

**Annotation File (10M):** <https://s3-eu-west-1.amazonaws.com/workshopdata2017/annotation.bed>

**In the following questions you will be asked to provide the command lines used to perform the steps of the pipeline and to comment and explain the choice of tools and all options. Please do not forget the latter as copying and pasting the command lines from the bash pipeline will not be sufficient to pass. You will need to demonstrate a clear understanding of your choices. Feel free to provide examples (even graphical/screenshots) if helpful.**

## 2.1 Install the tools and dependencies of your pipeline (using Miniconda when possible) and Download the input files (10 pts)

1. List the command lines to install all dependencies necessary to run the pipeline (3 pts)

Use Miniconda to install each program we will use (e.g. trimmomatic, fastqc):

conda install trimmomatic

conda install fastqc

conda install bwa

conda install samtools

conda install freebayes

conda install picard

conda install bedtools

conda install vcflib

Databases for Annovar (Annovar is assumed to be installed already):

cd ./annovar

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar knownGene humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar refGene humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar ensGene humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar clinvar\_20180603 humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar exac03 humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar dbnsfp31a\_interpro humandb/

./annotate\_variation.pl -buildver hg19 -downdb -webfrom annovar avsnp150 humandb/ #This one is particularly important, as a later step will be to filter by dbSNP

Download Human reference genome hg19 and place in correct location:

cd ./assessment

wget http://hgdownload.cse.ucsc.edu/goldenPath/hg19/bigZips/hg19.fa.gz

mv hg19.fa.gz ~/assessment/data/reference

Download and unzip snpEff:

cd ~

wget https://snpeff.blob.core.windows.net/versions/snpEff\_latest\_core.zip

unzip snpEff\_latest\_core.zip

1. List all command lines necessary to download the input files (e.g. fastqs, reference genomes, etc) (2 pts)

Wget <https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R1.fastq.qz>

Wget<https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R2.fastq.qz>

Wget <https://s3-eu-west-1.amazonaws.com/workshopdata2017/annotation.bed>

Wget downloads the link it is supplied.

Change name to .gz:

mv NGS0001.R1.fastq.qz NGS0001.R1.fastq.gz

mv NGS0001.R2.fastq.qz NGS0001.R2.fastq.gz

***Implement and run the following NGS Pipeline (please provide the command lines to run the following steps of your pipeline and comment/explain the choice of options):***

### 2.2. Pre-Alignment QC (4 pts)

1. Perform quality assessment and trimming (2pt)
2. Perform basic quality assessment of paired trimmed sequencing data (2pt)

fastqc -t 4 ~/assessment/data/NGS0001.R1.fastq.gz \

~/assessment/data/NGS0001.R2.fastq.gz

fastqc -t 4 $1 $2

Quality assessment with fastQC. -t specifies number of threads to use (i.e. no. of files which can be processed simultaneously). Arguments are Read1 and Read2 file paths. The pipeline uses variables $1 and $2 (user-defined on submission, order-specific)

mv ~/assessment/data/\*fastqc\* ~/assessment/results/fastqc\_untrimmed\_reads

Move fastQC output to correct folder.

Chart

Description automatically generated

Image shows per base sequence quality before trimming – note the poor quality towards the high end of reads.

trimmomatic PE \

-threads 4 \

-phred33 \

$1 $2 \

-baseout ~/assessment/data/trimmed\_fastq/trimmed\_data \

ILLUMINACLIP:/home/ubuntu/anaconda3/pkgs/trimmomatic-0.39-1/share/trimmomatic-0.39-1/adapters/NexteraPE-PE.fa:2:30:10 \

TRAILING:25 MINLEN:5

PE for paired end data

-threads tells trimmomatic how many threads to use (more -> faster)

-phred33 specifies the type of phred score that is present in the data (33 or 64 dependent on the kind of Illumina pipeline used).

Baseout tells trimmomatic where to save the trimmed data and what to call the outputs (‘trimmed\_data’ is the base of the name, to which suffixes will be added for each of the 4 outputs).

Illuminaclip removes sequences that are specific to the illumina sequencing process. In this case, it is removing Nextera adapters (although FastQC indicates that none are present in the dataset, so this is not strictly required).

Chart

Description automatically generated

Trailing cuts bases off at the ends of reads if they are below the specified quality (25)

Minlen trims reads below the specified minimum length (5).

fastqc -t 4 ~/assessment/data/trimmed\_fastq/trimmed\_data\_1P \

~/assessment/data/trimmed\_fastq/trimmed\_data\_2P

QC step again, but with trimmed data. Per base sequence quality looks much better after trimming.

Chart

Description automatically generated

### 2.3. Alignment (17pts)

* Align the paired trimmed fastq files using bwa mem and reference genome hg19 (edit your bwa mem step to include read group information in your BAM file) (9pts)

First create a Genome index for bwa mem:

bwa index ~/assessment/data/reference/hg19.fa.gz

Then run bwa mem:

bwa mem -t 4 -v 1 -R '@RG\tID:11V6WR1.111.D1375ACXX.1.NGS0001\tSM:NGS0001\tPL:ILLUMINA\tLB:nextera-NGS0001\tDT:2017-02-23\tPU:D1375ACXX.1' -I 212,31 \

~/assessment/data/reference/hg19.fa.gz ~/assessment/data/trimmed\_fastq/trimmed\_data\_1P \

~/assessment/data/trimmed\_fastq/trimmed\_data\_2P > \

~/assessment/data/aligned\_data/NGS0001.sam

-t: thread count

-v: Verbose (shows a longer output as it runs)

-R: add read group information (followed by a tab separated header). Header consists of:

* ID info (Instrument name, run ID, flowcell ID, lane, sample name)
* SM: Sample name
* PL: Platform used to generate reads
* LB: DNA prep library ID
* DT: Date run produced
* PU: Platform Unit

-I: Insert length mean and standard deviation

Input of human reference genome (hg19), trimmed data (R1 & R2), output file directory/name.

Convert sam to bam format with samtools view (note, in the pipeline this was done directly with a pipe | because the sam file was too large to create)

samtools view -h -b > ~/assessment/data/aligned\_data/NGS0001.bam

-h: include header

-b: output in bam format

change directory to aligned data folder, then Sort and index bam file for later steps.

cd ~/assessment/data/aligned\_data

samtools sort NGS0001.bam > NGS0001\_sorted.bam

samtools index NGS0001\_sorted.bam

* Perform duplicate marking (2pts)

picard MarkDuplicates I=NGS0001\_sorted.bam O=NGS0001\_sorted\_marked.bam M=marked\_dup\_metrics.txt

Identifies duplicate reads.

I: input sorted bam file

O: output bam file with marked duplicate reads

M: name of duplicate metric (summary) file to output

* Quality Filter the duplicate marked BAM file (2pts)

First create a new index for the (sorted and marked) bam file:

samtools index NGS0001\_sorted\_marked.bam

Then use samtools view to quality filter bam file:

samtools view \

-F 1796 \

-q 20 \

-o NGS0001\_sorted\_filtered.bam NGS0001\_sorted\_marked.bam

-F: Does not output alignments with the following integers (1796) present in the *flag* field. 1796 represents four unwanted flags: unmapped reads, non-primary reads, read failed platform quality checks, and read is a PCR/optical duplicate.

-q: ignores (i.e. removes) alignments with MAPQ (quality score) smaller than the specified value (20)

-o: set output file name

The final file listed is the input bam.

Create another index for the sorted and filtered bam:

samtools index NGS0001\_sorted\_filtered.bam

* Generate standard alignment statistics (i.e. flagstats, idxstats, depth of coverage, insert size) (4pts)

Flagstat (provides a summary/counts of reads for each type of flag):

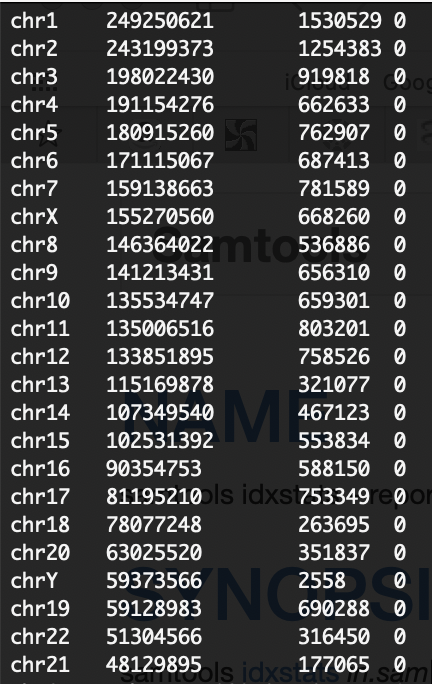
samtools flagstat NGS0001\_sorted\_filtered.bam > flagstats\_out.txt

Text

Description automatically generated

Idxstats (summary of alignments per chromosome):

samtools idxstats NGS0001\_sorted\_filtered.bam > alignstats\_per\_chr.txt



Insert Size Metrics:

picard CollectInsertSizeMetrics I= NGS0001\_sorted\_filtered.bam O= insert\_size\_metrics.txt H= insert\_size\_histogram.pdf M= 0.05

Text

Description automatically generated

Chart, histogram

Description automatically generated

This time, we can create a histogram with ‘H=hist\_name.pdf’

‘M=’ allows you to specify the minimum percentage reads any category (pair orientation) should have in order to be shown on the histogram (in this case, only FR).

Coverage:

bedtools coverage \

-a NGS0001\_sorted\_filtered.bam \

-b ~/assessment/data/annotation.bed > coverage.txt

-a: File a

-b: File b

Bedtools coverage searches for overlaps in features between files a (our bam file) and b (annotation bed file) to calculate coverage depth and breadth.

### 2.4. Variant Calling (4pts)

* Call Variants using Freebayes restricting the analysis to the regions in the bed file provided (2pt)

Decompress reference genome:

zcat ~/assessment/data/reference/hg19.fa.gz > ~/assessment/data/reference/hg19.fa

Index the reference genome with faidx:

samtools faidx ~/assessment/data/reference/hg19.fa

Use Freebayes for variant calling:

freebayes \

--bam ~/assessment/data/aligned\_data/NGS0001\_sorted\_filtered.bam \

--fasta-reference ~/assessment/data/reference/hg19.fa \

--vcf ~/assessment/results/NGS0001.vcf

--bam: input bam file

--fasta-reference: input human reference genome

--vcf: output variant call format file

--targets [bam] could be used here to limit regions to those in the bam file. Instead, *bedtools intersect* is used later in the pipeline to limit to regions in the annotation bed file.

Compress the VCF:

bgzip ~/assessment/results/NGS0001.vcf

Index with tabix:

tabix -p vcf ~/assessment/results/NGS0001.vcf.gz

-p: specifies the type of input file to be indexed (vcf).

* Quality Filter Variants using your choice of filters (2pt)

vcffilter \

-f "QUAL > 1 & QUAL / AO > 10 & SAF > 0 & SAR > 0 & RPR > 1 & RPL > 1" \

~/assessment/results/NGS0001.vcf.gz > ~/assessment/results/NGS0001\_filtered.vcf

-f: filter alleles on info column using the terms in quotation marks.

QUAL: Quality of the site (note that this does not appear in the VCF info field, but can be successfully called)

A0: Alternate allele observations

SAF: Count of alternate observations on the forward strand

SAR: Count of alternate observations on the reverse strand

RPR: ‘Reads placed right’

RPL: ‘Reads placed left’

Restrict to only regions in bed file with bedtools intersect:

bedtools intersect \

-header -wa -a ~/assessment/results/NGS0001\_filtered.vcf \

-b ../annotation.bed > ~/assessment/results/NGS0001\_filtered\_anno.vcf

Compress and index new VCF, as before:

bgzip ~/assessment/results/NGS0001\_filtered\_anno.vcf

tabix -p vcf ~/assessment/results/NGS0001\_filtered\_anno.vcf.gz

### 2.5. Variant Annotation and Prioritization (10pts)

* Annotate variants using **ANNOVAR** (4pt) and **snpEFF** (4pt)

Run convert2annovar.pl perl script to create a file for annovar input:

~/assessment/annovar/convert2annovar.pl \

-format vcf4 ~/assessment/results/NGS0001\_filtered\_anno.vcf.gz > \

~/assessment/results/NGS0001\_filtered\_anno.avinput

-format vcf4: tells the perl script to convert from a VCF file input

Run the annovar table perl script to output a csv:

~/assessment/annovar/table\_annovar.pl \

~/assessment/results/NGS0001\_filtered\_anno.avinput ~/assessment/annovar/humandb/ \

-buildver hg19 \

-out ~/assessment/results/NGS0001\_filtered\_anno \

-remove \

-protocol refGene,ensGene,clinvar\_20180603,exac03,dbnsfp31a\_interpro,avsnp150 \

-operation g,g,f,f,f,f \

-otherinfo \

-nastring . \

-csvout

Humandb is a directory containing human reference databases. It should be within the annovar directory.

-buildver: human reference genome build used

-out: output file path/name

-remove: remove all temporary files

-protocol: specifies protocols to use (each one will be a column in your csv).

-operation: tells annovar which operations to run for each protocol in order. G is gene-based and f is filter-based. Number of inputs here must equal number of protocols to include.

-otherinfo: include otherinfo columns in output table

-nastring: N/A is written as the input string ‘.’

-csvout: return a csv as output

SNPeff:

java -Xmx8g -jar snpEff.jar -v -stats snpeff\_ann\_stats.html hg19 ~/assessment/results/NGS0001\_filtered\_anno.vcf.gz > ~/assessment/results/NGS0001\_snpeff.ann.vcf

-Xmx8g: Java memory allocation flag

-jar: use a .jar file

-v: verbose (show working in terminal).

-stats: output a html summary file

Output is an annotated VCF, summary HTML, and a tab-separated summary table.

Move outputs to results:

mv ./snpeff\_ann\_stats\* ~/assessment/results/

* Perform basic variant prioritization: filter to exonic variants not seen in dbSNP (2pts)

This step can be performed manually in Wannovar (online tool) or in excel.

Here, variants were filtered to include only exonic regions (col F and col I), and not seen in dbSNP (col AD = ‘.’)

Graphical user interface, table

Description automatically generated\

Graphical user interface, application, table, Excel

Description automatically generated

## 3. R/RStudio assessment (40pts – 40% of final mark)

In this assessment you will be asked to perform a number of tasks in R/RStudio and report them in your own markdown document.

Initial task: Create a new markdown document in *RStudio*, set the title to "Advanced Bioinformatics 2019 assessment", and insert an "author:" tag below the title, followed by your student id. **Share your markdown document and html via your github account.**

In the following, for each task, create a new heading called "Task X" for task X, and insert a new R code chunk that holds any code required. Make sure to evaluate the expression before saving to include the output in the html file. If you have multiple lines that produce outputs, you can split them into separate code chunks for increase clarity (but it is not necessary to pass the assessment). Please also explain your steps.

3.1. Using the *sum*() function and : operator, write an expression in the code snippet to evaluate the sum of all integers between 5 and 55. (5pt)

3.2. Write a function called *sumfun* with one input parameter, called *n*, that calculates the sum of all integers between 5 and *n*. Use the function to do the calculation for *n* = 10, *n* = 20, and *n* = 100 and present the results. (5pt)

3.3. The famous Fibonacci series is calculated as the sum of the two preceding members of the sequence, where the first two steps in the sequence are 1, 1. Write an R script using a for loop to calculate and print out the first 12 entries of the Fibonacci series. (5pt)

3.4. With the *mtcars* dataset bundled with R, use *ggplot* to generate a box of miles per gallon (in the variable *mpg*) as a function of the number of gears (in the variable *gear*). Use the fill aesthetic to colour bars by number of gears. (5pt)

3.5. Using the *cars* dataset and the function *lm*, fit a linear relationship between *speed* and breaking distance in the variable *distance*. What are the fitted slope and intercept of the line, and their standard errors? What are the units used for the variables in the dataset? (5pt)

3.6. Use *ggplot* to plot the data points from Task 6 and the linear fit. (5pt)

3.7. Again using the cars dataset, now use linear regression (*lm*) to estimate the average reaction time for the driver to start breaking (in seconds). To simplify matters you may assume that once breaking commences, breaking distance is proportional to the square of the speed. Explain the steps in your analysis. Do you get reasonable results? Finally, use *ggplot* to plot the data points and the fitted relationship. (10pt)