Advanced Bioinformatics (7BBG2016): Practical Bioinformatics Data Skills

Student ID: 18040

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
D. None of Above

Up 2 directories from the current directory (C).

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

"cd /" takes you to the root directory, i.e. the "highest" directory in the hierarchy.

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

```
man cp
```

This brings up the help documentation for the command.

1.4 What does the command pwd do?

Prints the complete path of current working directory, starting from the root (/).

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 -l
```

This displays a listing of file details such as date, size, and access permissions in a given directory. If no directory is specified, the listing will be for the current working directory.

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 newfile oldfile
```

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

```
less myfile.txt
```

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 ~/newdir
```

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

```
head file1.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 -f2 file1.txt
```

This extracts column "2" from file1.

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

```
mkdir Project_backup; 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)

```
sleep 1337
```

Then press Ctrl+Z on keyboard.

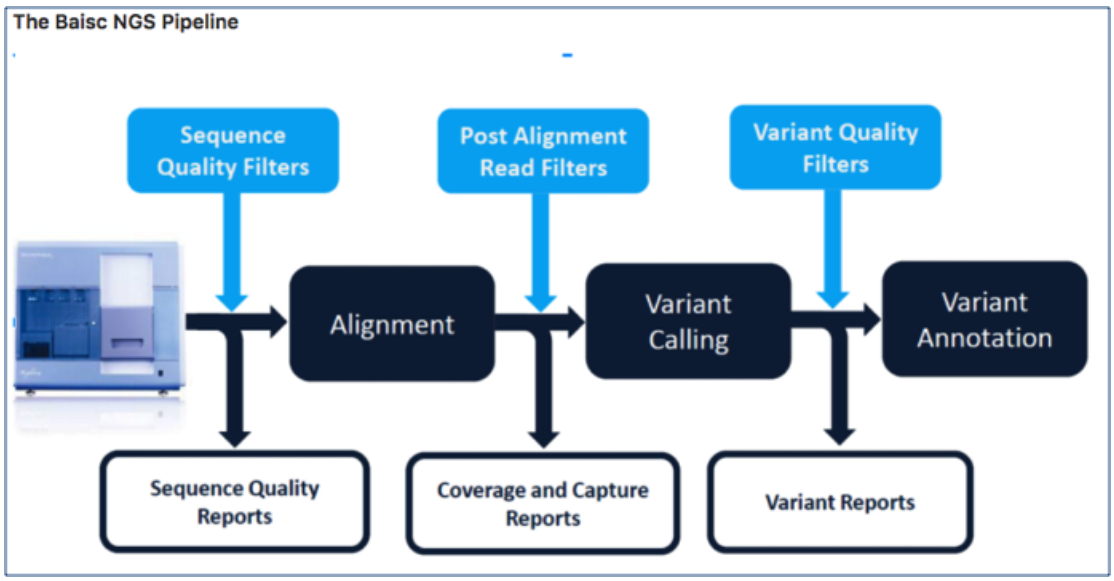
```
bg
disown
logout
```

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)

```
# Install Anaconda
wget https://repo.anaconda.com/archive/Anaconda3-2020.02-Linux-
x86_64.sh
chmod +x ./Anaconda3-2020.02-Linux-x86_64.sh
bash ./Anaconda3-2020.02-Linux-x86_64.sh
source ~/.bashrc

# Install required packages with Anaconda
conda install samtools bwa freebayes picard bedtools trimmomatic
fastqc vcflib
```

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

```
cd data
mkdir untrimmed_fastq trimmed_fastq

# download FASTQ raw read data
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

mv *fastq.qz untrimmed_fastq

# download BED file
wget https://s3-eu-west-
1.amazonaws.com/workshopdata2017/annotation.bed

# Convert .qz FASTQ files to more usable .gz files
cd untrimmed_fastq
mv NGS0001.R1.fastq.qz NGS0001.R1.fastq.gz
mv NGS0001.R2.fastq.qz NGS0001.R2.fastq.gz

# Download reference genome FASTA file
mkdir ../reference
cd ../reference
wget
http://hgdownload.cse.ucsc.edu/goldenPath/hg19/bigZips/hg19.fa.gz
# generate index of reference genome fasta file for alignment step
bwa index hg19.fa.gz # run this step now as it takes ~45 mins, making
troubleshooting of the steps later in the pipeline easier
```

```
# Download and setup Annovar database for variant annotation
cd ~/annovar
tar -zxvf annovar.latest.tar.gz
# N.B. this file must be downloaded from the Annovar website as it is
not opensource. D/L link =
http://download.openbioinformatics.org/annovar_download_form.php

# setup databases
./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 \
snp138 humandb/
# due to the size of the dbSNP 138 database, I had to run this step
in my local environment
```

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)

```
workdir=~/ngs_assignment # set workdir variable, enabled me to use my
script from the module workshop more easily

# run FastQC to generate quality metrics on untrimmed reads
fastqc -t 4 $workdir/data/untrimmed_fastq/*.fastq.gz

# Move FastQC results to a new directory
mkdir $workdir/results/fastqc_untrimmed_reads
mv $workdir/data/untrimmed_fastq/*fastqc*
$workdir/results/fastqc_untrimmed_reads/

# use Trimmomatic to trim away adapters and filter out poor quality
score reads. This Drops reads below 50 bp in length, and trims bases
from the end of reads, if below a threshold quality (25). Also
removes Illumina adapter sequences.
trimmomatic PE  \
  -threads 4 \
  -phred33 \
  $workdir/data/untrimmed_fastq/NGS0001.R1.fastq.gz \
$workdir/data/untrimmed_fastq/NGS0001.R2.fastq.gz \
  -baseout $workdir/data/trimmed_fastq/NGS0001_trimmed_R \
  ILLUMINACLIP:/home/ubuntu/anaconda3/pkgs/trimmomatic-0.39-\
1/share/trimmomatic-0.39-1/adapters/NexteraPE-PE.fa:2:30:10 \
  TRAILING:25 MINLEN:50
```

Measure	Value
Filename	NGS0001.R1.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	9007781
Sequences flagged as poor quality	0
Sequence length	100
%GC	46

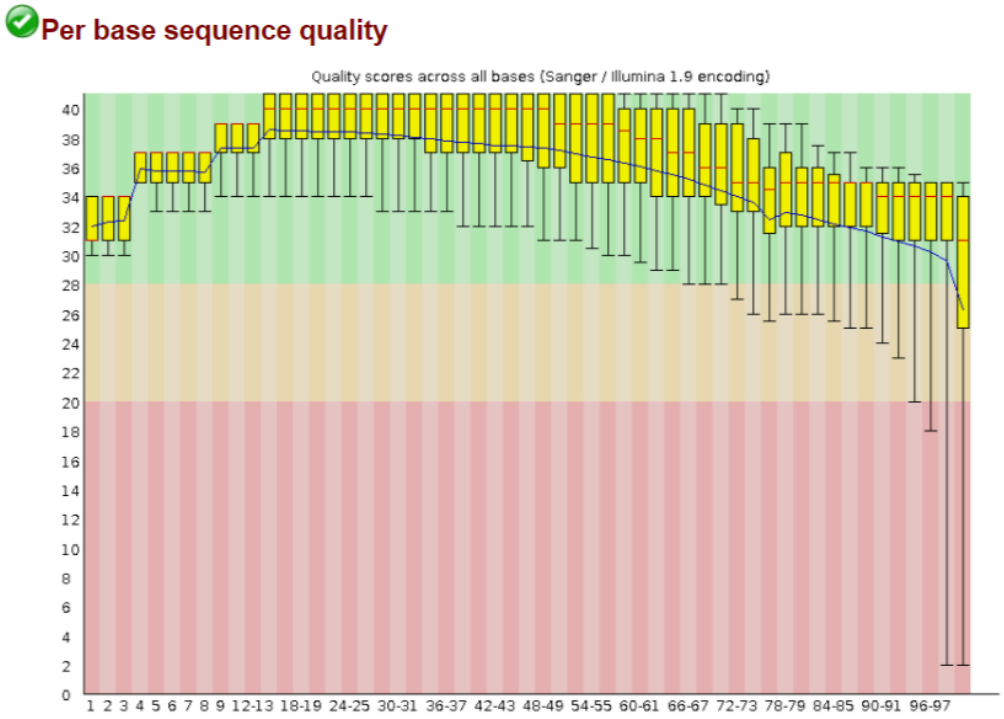


Figure 1: Basic statistics and per base sequencing quality from FastQC analysis on raw sequencing data in NGS0001.R1.fastq prior to trimming.

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

```
# run FastQC to generate quality metrics on trimmed reads
fastqc -t 4 $workdir/data/trimmed_fastq/NGS0001_trimmed_R_1P \
  $workdir/data/trimmed_fastq/NGS0001_trimmed_R_2P

# Move FastQC results to a new directory
mkdir $workdir/results/fastqc_trimmed_reads
mv $workdir/data/trimmed_fastq/*fastqc*
$workdir/results/fastqc_trimmed_reads/
```

Measure	Value
Filename	NGS0001_trimmed_R_1P
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	8692365
Sequences flagged as poor quality	0
Sequence length	50-100
%GC	46

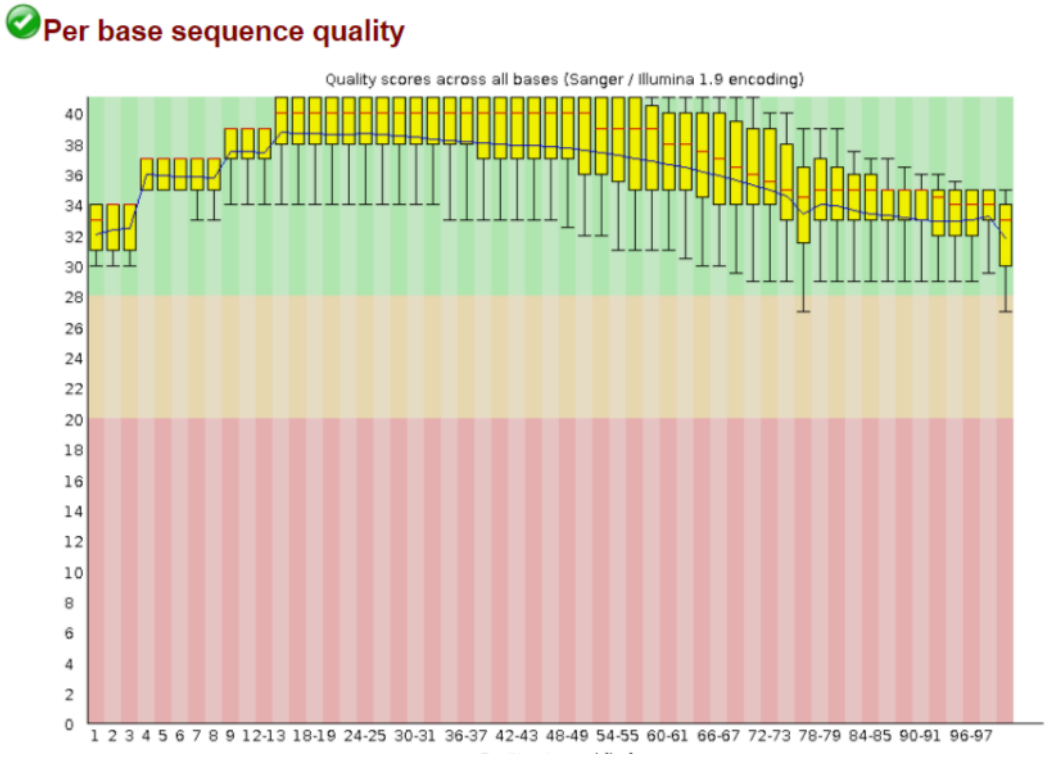


Figure 2: Basic statistics and per base sequencing quality from FastQC analysis on NGS0001.R1.fastq after trimming. Note the overall improvement in Phred quality scores in bases towards the end of reads compared to Figure 1.

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)

```
mkdir $workdir/data/aligned_data
alignment_dir=$workdir/data/aligned_data

# alignment of trimmed reads to hg19 reference genome, using BWA-MEM algorithm.
# read group info obtained from the raw read FASTQ files.
bwa mem -t 4 -v 1 -R
'@RG\tID:11V6WR1:111:D1375ACXX:1.NGS0001\tSM:NGS0001\tPL:ILLUMINA\' \
  -I 250,50  $workdir/data/reference/hg19.fa.gz \
$workdir/data/trimmed_fastq/NGS0001_trimmed_R_1P \
  $workdir/data/trimmed_fastq/NGS0001_trimmed_R_2P > \
$workdir/data/aligned_data/NGS0001.sam
```

```
# covert SAM to BAM file:
samtools view -h -b $alignment_dir/NGS0001.sam > \
$alignment_dir/NGS0001.bam
# sort the BAM file:
samtools sort $alignment_dir/NGS0001.bam > \
$alignment_dir/NGS0001_sorted.bam
# generate index:
samtools index $alignment_dir/NGS0001_sorted.bam
```

- Perform duplicate marking (2pts)

```
# this step locates and marks duplicated reads in the BAM file
picard MarkDuplicates I=$alignment_dir/NGS0001_sorted.bam \
O=$alignment_dir/NGS0001_sorted_marked.bam \
  M=$alignment_dir/marked_dup_metrics.txt

samtools index $alignment_dir/NGS0001_sorted_marked.bam # generate
index
```

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

```
# Filter reads below a minimum MAPQ score (-q 20). The samtools flag
-F 1796 filters reads that are unmapped, not in the primary
alignment, that fail platform/vendor quality checks, or that are
PCR/optical duplicates. For more info see:
https://broadinstitute.github.io/picard/explain-flags.html
samtools view -F 1796  -q 20 -o \
$alignment_dir/NGS0001_sorted_marked_filtered.bam \
  $alignment_dir/NGS0001_sorted_marked.bam

# generate index
samtools index $alignment_dir/NGS0001_sorted_marked_filtered.bam
```

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

```
mkdir $alignment_dir/alignment_stats # make a new directory
stats_dir=$alignment_dir/alignment_stats

# Generate flagstats
samtools flagstats \
$alignment_dir/NGS0001_sorted_marked_filtered.bam \
  > $stats_dir/flagstats_output.txt

# view the BAM file in the command line
# samtools view -h $alignment_dir/NGS0001_sorted_marked_filtered.bam
| less

# Generate alignment statistics per chromosome with idxstats
samtools idxstats $alignment_dir/NGS0001_sorted_marked_filtered.bam \
  > $stats_dir/idxstats_output.txt
```

```
# Determine the distribution of insert sizes between read pairs with
Picard tools
picard CollectInsertSizeMetrics -H \
$stats_dir/CollectInsertSizeMetrics_histogram.pdf \
      -I $alignment_dir/NGS0001_sorted_marked_filtered.bam -O \
$stats_dir/CollectInsertSizeMetrics_output.txt

# Calculate Depth of coverage. First get the all BAM regions that
overlap with the input, then use this output to calculate the
coverage.(This saves memory usage)
bedtools intersect -bed -a NGS0001_sorted_marked_filtered.bam -b \
$workdir/data/annotation.bed \
  | bedtools coverage -d -a $workdir/data/annotation.bed -b - \
  > $stats_dir/coverageBed_output.txt
```

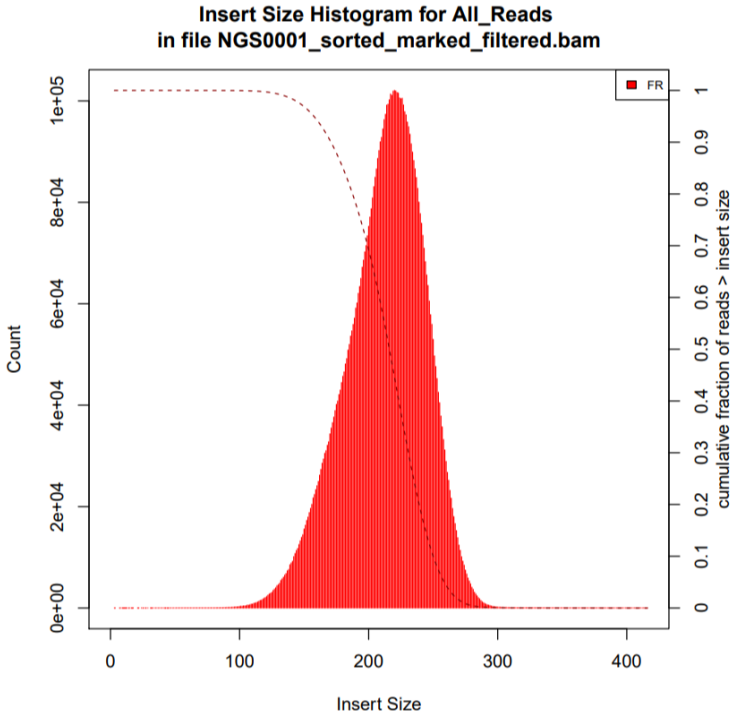


Figure 3: Distribution of insert sizes between read pairs, determined with Picard tools.

2.4. Variant Calling (4pts)

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

```
# uncompress reference genome, as this is required by freebayes
zcat $workdir/data/reference/hg19.fa.gz > \
$workdir/data/reference/hg19.fa samtools faidx
# generate index
$workdir/data/reference/hg19.fa

# use FreeBayes to report putative variants in the sequencing data
compared to the reference allele
freebayes --bam $alignment_dir/NGS0001_sorted_marked_filtered.bam \
  --fasta-reference $workdir/data/reference/hg19.fa --vcf \
$workdir/results/NGS0001.vcf

bgzip -f $workdir/results/NGS0001.vcf # compress VCF
```

```
tabix -p vcf $workdir/results/NGS0001.vcf.gz # index VCF
```

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

```
# remove "bad" variant calls from the VCF.
# "QUAL > 1" removes horrible sites, "QUAL / AO > 10" additional
contribution of each obs should be 10 log units (~ Q10 per read),
"SAF > 0 & SAR > 0" reads on both strands, "RPR > 1 & RPL > 1" at
least two reads "balanced" to each side of the site.
vcffilter -f "QUAL > 1 & QUAL / AO > 10 & SAF > 0 & SAR > 0 & RPR > 1
& RPL > 1" \
  $workdir/results/NGS0001.vcf.gz > \
$workdir/results/NGS0001_filtered.vcf

# use bedtools to filter VCF for regions present in the annotation
bed file.
bedtools intersect -header -wa -a \
$workdir/results/NGS0001_filtered.vcf -b \
$workdir/data/annotation.bed \
  > $workdir/results/NGS0001_filtered_in_bedfile.vcf

# compress filtered VCF
bgzip -f $workdir/results/NGS0001_filtered_in_bedfile.vcf
# index filtered VCF
tabix -p vcf $workdir/results/NGS0001_filtered_in_bedfile.vcf.gz
```

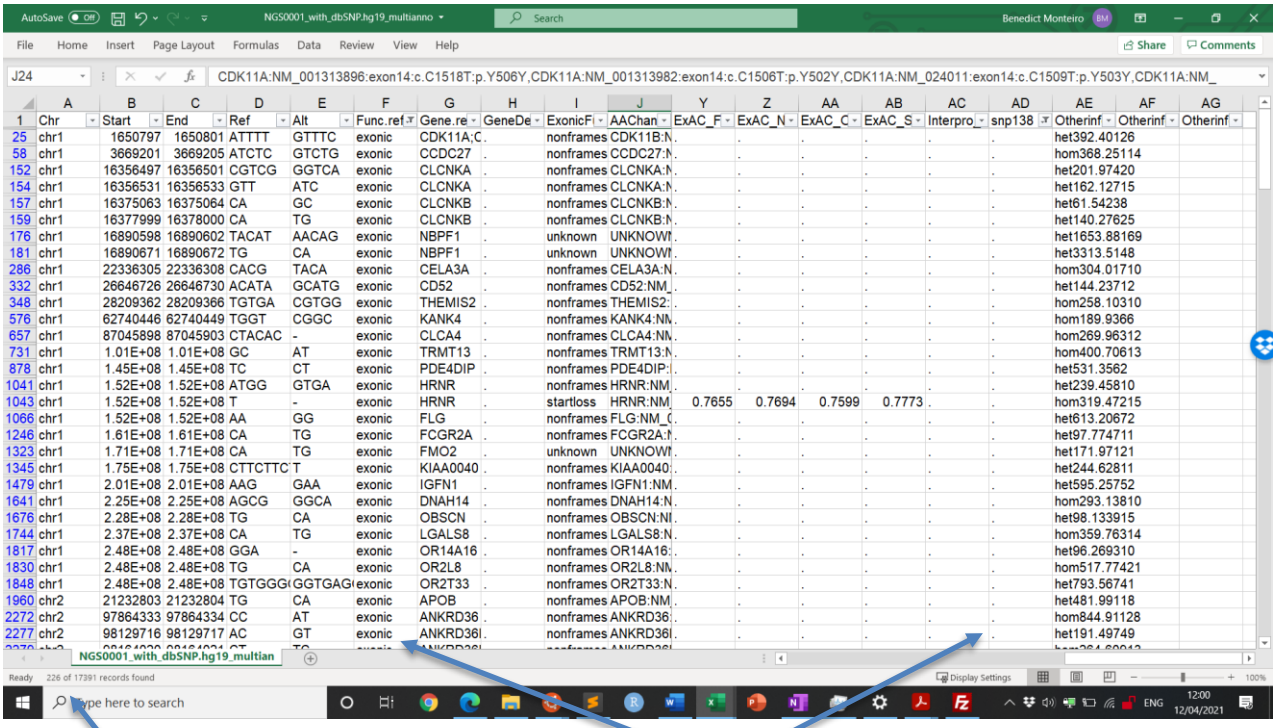
2.5. Variant Annotation and Prioritization (10pts)

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

```
# Convert VCF to annovar input
~/tools/annovar/convert2annovar.pl -format vcf4 \
$workdir/results/NGS0001_filtered_in_bedfile.vcf.gz  \
  > $workdir/results/NGS0001_filtered_in_bedfile.avinput

# Run Annovar to annotate variants with database
frequencies/functional consequences. Output is a .csv that can be
opened in MS Excel.
~/tools/annovar/table_annovar.pl \
$workdir/results/NGS0001_filtered_in_bedfile.avinput \
~/tools/annovar/humandb/ -buildver hg19 \
  -out $workdir/results/NGS0001_filtered_in_bedfile -remove \
  -protocol \
refGene,ensGene,clinvar_20180603,exac03,dbnsfp31a_interpro \
-operation g,g,f,f,f -otherinfo -nastring . -csvout


# To add dbSNP database annotations, I also performed this step in my
local environment:
input_file=~/../../mnt/c/Users/redacted/Dropbox/Masters/8_Advanced_Bi
oi\nformatics/Assignment/NGS0001_filtered_in_bedfile.avinput

output_file=~/../../mnt/c/Users/redacted/Dropbox/Masters/8_Advanced_B
io\informatics/Assignment/annovar_out/NGS0001_with_dbSNP

~/Annovar/annovar/table_annovar.pl $input_file \
/Annovar/annovar/humandb/ -buildver hg19 \
  -out $output_file -remove \
  -protocol \
```

```
refGene,ensGene,clinvar_20180603,exac03,dbnsfp31a_interpro,snp138 -\
operation g,g,f,f,f,f -otherinfo -nastring . -csvout
```

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



Filtering by exonic variants not in the dbSNP 138 database (marked by ".")

This leaves 226 prioritised variants

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)

Advanced Bioinformatics 2021 R assessment

Candidate Number: 18040

15/04/2021

Task 3.1

Using the `sum()` function to evaluate the sum of all integers between 5 and 55.

```
sum(c(5:55))
```

```
## [1] 1530
```

Task 3.2

A function that calculates the sum of all integers between 5 and n.

```
sumfun = function(n){
  output = sum(c(5:n))
  return(output)
}
sumfun(10)
```

```
## [1] 45
```

```
sumfun(20)
```

```
## [1] 200
```

```
sumfun(100)
```

```
## [1] 5040
```

Task 3.3

Loop to calculate and print the first 5 terms of the Fibonacci sequence.

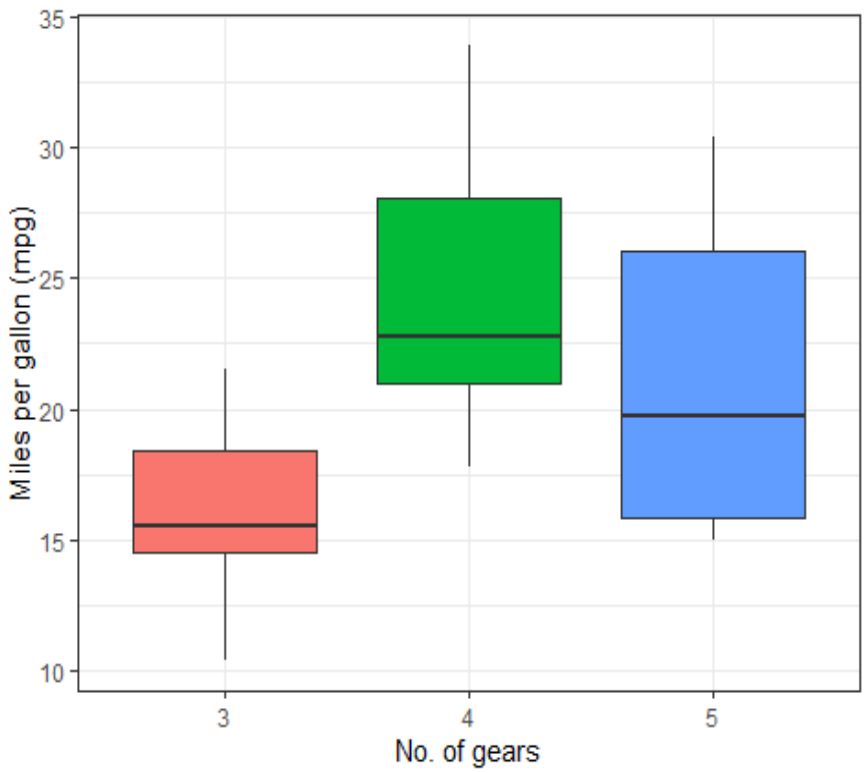
```
for(i in 1:12){
  if(i == 1){
    current_num = i
    prev_num = 0 # as no previous num in the sequence
  }else{
    current_num = prev_num + prev_prev_num
  }
    print(current_num) # print current entry
    # update values for next iteration
    prev_prev_num = prev_num
    prev_num = current_num
}
```

```
## [1] 1
## [1] 1
## [1] 2
## [1] 3
## [1] 5
## [1] 8
## [1] 13
## [1] 21
## [1] 34
## [1] 55
## [1] 89
## [1] 144
```

Task 3.4

Boxplot to show miles per gallon (in the variable mpg) as a function of the number of gears. Data taken from the `mtcars` dataset.

```
library(tidyverse)

ggplot(mtcars) +
  geom_boxplot(aes(x = factor(gear), y = mpg, fill = factor(gear)))
+
  ylab("Miles per gallon (mpg)") + xlab("No. of gears") +
  theme_bw() + theme(legend.position = "none")
```



Task 3.5

Performing a linear regression of car speed versus breaking distance using the `lm()` function. Data taken from the `cars` dataset.

```
# assign linear regression to a variable
reg_output = summary(lm(formula= dist~speed, data = cars))

# assign linear regression coefficients to variables
reg_coeffs = reg_output$coefficients

# extract and print slope values
slope = reg_coeffs["speed","Estimate"]
slope_se = reg_coeffs["speed","Std. Error"]

print(paste("The fitted slope of the line is", round(slope, digits =
 2), "with a standard error of", round(slope_se, digits = 2)))
```

```
## [1] "The fitted slope of the line is 3.93 with a standard error o
f 0.42"
```

```
# extract and print intercept values
intercept = reg_coeffs["(Intercept)","Estimate"]
intercept_se = reg_coeffs["(Intercept)","Std. Error"]

print(paste("The intercept of the line is", round(intercept,digits =
 2), "with a standard error of",  round(intercept_se,digits = 2)))
```

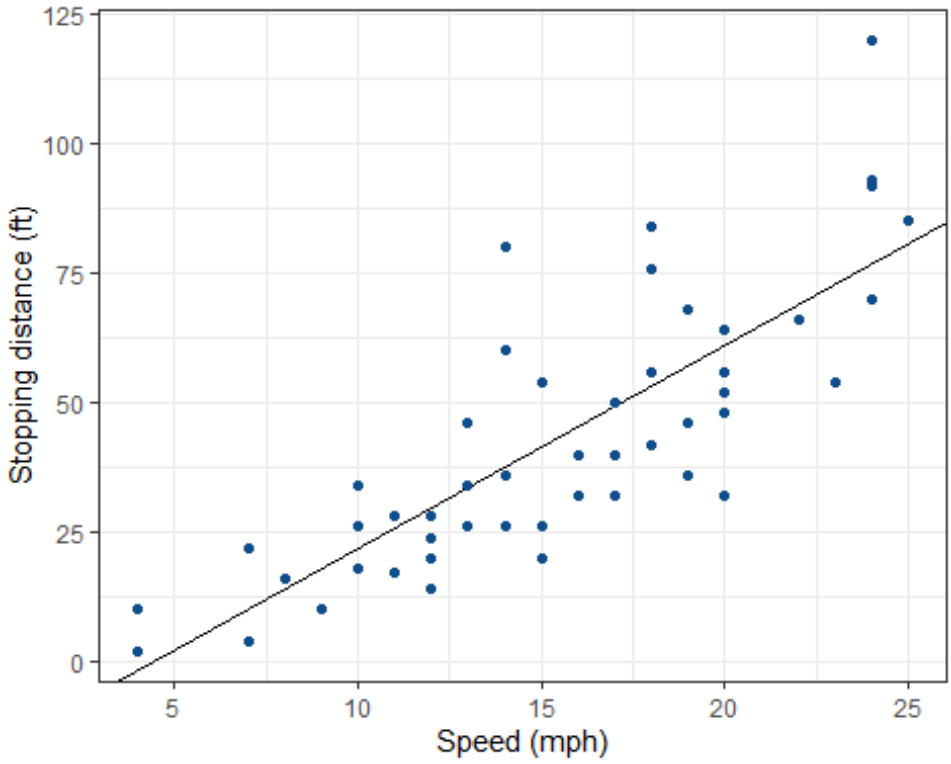
```
## [1] "The intercept of the line is -17.58 with a standard error of
 6.76"
```

The units in the `cars` dataset are miles per hour (mph) for speed and feet (ft) for stopping distance (dist). Indentified in the `?cars` help documentation.

Task 3.6

Plotting the data points from `cars` and the linear fit calculated in Task 3.5 with ggplot.

```
ggplot(cars, mapping = aes(x = speed, y = dist)) +
  geom_abline(intercept = intercept, slope = slope, colour = "black"
) + # these values are taken directly from task 3.5
  geom_point(colour = "dodgerblue4") +
  xlab("Speed (mph)") + ylab("Stopping distance (ft)") +
  theme_bw()
```



Task 3.7

Using linear regression to estimate the average reaction time for a driver to start braking (in seconds). Data taken from the `cars` dataset. To simplify matters it is assumed that once braking commences, braking distance is proportional to the square of the speed.

As braking distance is proportional to the square of the speed, this can be modelled in a linear regression:

```
cars_edit = cars %>% # Adding the square of the speed to the dataset
  mutate(speed_squared = speed^2)

reaction_reg = summary(lm(formula = dist ~ speed_squared, data = car
s_edit))
```

The slope of this linear regression describes the relationship between stopping distance and speed squared. The y-intercept (where mph2 is equal to 0) therefore describes the theoretical mean value of braking distance when speed is equal to 0, i.e. the portion of braking distance that is not dependent on speed: the reaction distance.

```
rxn_intercept = reaction_reg$coefficients["(Intercept)","Estimate"]
rxn_intercept_se = reaction_reg$coefficients["(Intercept)","Std. Err
or"]

print(paste("The intercept of the linear fit is", round(rxn_intercep
```

```
t,digits = 2), "ft with a standard error of",  round(rxn_intercept_s
e,digits = 2),"ft."))

## [1] "The intercept of the linear fit is 8.86 ft with a standard e
rror of 4.09 ft."
```

As this value is taken from the linear fit of the data points in `cars`, we can assume that this value is the average distance a car going at average speed in the dataset travels before starting to brake (i.e. the reaction distance).

```
mean_speed = mean(cars_edit$speed)
print(paste("The mean average speed of the datapoints in cars is", r
ound(mean_speed,digits = 2), "mph."))

## [1] "The mean average speed of the datapoints in cars is 15.4 mph
."
```

We can now use the mean speed and reaction distance to estimate the average reaction time, by rearranging the equation "speed = distance/time". As our intercept is in ft, we will convert our mean speed from mph into ft per second, by multiplying our speed value by 1.47 (as there are 5280 ft in a mile, and 3600 seconds in an hour, 5280/3600 = 1.47).

```
rxn_time = rxn_intercept / (mean_speed*1.47)
print(paste("The estimated average of reaction time in cars is", rou
nd(rxn_time,digits = 2), "seconds."))

## [1] "The estimated average of reaction time in cars is 0.39 secon
ds."
```

However, due to the significant standard error of the intercept in the linear fit, the average distance before braking commences could be anywhere between the upper and lower limits of this standard error. This can be used to calculate the upper and lower limits of average reaction:

```
# Calculate upper limit
rxn_intercept_upper_lim = rxn_intercept + rxn_intercept_se
rxn_time_upper_lim = rxn_intercept_upper_lim / (mean_speed*1.47)

print(paste("The upper limit of the estimated average of reaction ti
me in cars is", round(rxn_time_upper_lim,digits = 2), "seconds."))

## [1] "The upper limit of the estimated average of reaction time in
 cars is 0.57 seconds."

# Calculate lower limit
rxn_intercept_lower_lim = rxn_intercept - rxn_intercept_se
rxn_time_lower_lim = rxn_intercept_lower_lim / (mean_speed*1.47)

print(paste("The lower limit of the estimated average of reaction ti
```

```
me in cars is",
            round(rxn_time_lower_lim,digits = 2), "seconds."))

## [1] "The lower limit of the estimated average of reaction time in
 cars is 0.21 seconds."
```

Mean average reaction time from 81 million online tests is 0.284 seconds (source: https://humanbenchmark.com/tests/reactiontime/statistics), which is within the range of our estimation. As our estimate was based on lots of assumptions and data from the 1920s, we can conclude that the use of linear regression here gives a reasonable estimation of reaction time.

Finally, we can plot this linear regression on a graph of braking distance as a function of speed squared from `cars`:

```
rxn_slope = reaction_reg$coefficients["speed_squared","Estimate"]

ggplot(cars_edit, mapping = aes(x= speed_squared, y = dist)) +
  geom_point() +
  xlab("Speed (mph) squared") + ylab("Stopping distance (ft)") +
  geom_abline(intercept = rxn_intercept, slope = rxn_slope, colour =
 "red") +
  theme_bw()
```