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

**Student ID: m1700635**

## 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

# The correct answer is number C.#

1.2 What does cd / mean in UNIX? Please explain what the cd command does.  
cd stands for change directory and sometimes (known also as chdir). cd is a command line shell, and it is used to change the current working directory in many operating systems including UNIX. cd is used for navigation into any directory. It can direct you to the folder you want to be in such as cd data means it will take you to the data folder. cd / means change to root directory or take you to the root directory of your UNIX (including home), which always has more documents than the user directory. It is used in batch files and shell scripts.

|  |  |
| --- | --- |
| **Option** | **Description** |
| $ cd | Change to home directory |
| $ cd ~ | Change to home directory |
| $ cd / | Change to root directory |
| $ cd .. | Change to parent directory |
| $ cd data | Change to subdirectory data |
| $ cd dnaseq/data | Change to subdirectory dnaseq/data |
| $ cd /home/usr | Change to a directory with absolute path /home/usr |

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

cp stands for copy and it is a Linux/UNIX shell command. It is used to copy files or directories. It is creating the exact file or directory by copying from location to another location or copying within the same location but in a different name. I have to use man cp command that stands for manual copy. This command is used to get help about the command cp. We can use: $ cp --help to display this help and exit.

In case of no space, we can use the command $ mv to move the data instead of $ cp

For example:

$ mv mynewfile.txt myoldfile.txt

$ cp mynewfile.txt myoldfile.txt

**cp Command Options and Examples:**

|  |  |
| --- | --- |
| **cp Command Options Examples** | **Description** |
| $ cp dnaseq data | Copy from dnaseq to data file |
| $ cp -a | Archive files |
| $ cp -f | Force file copying to directory - even if you have to remove the destination file (e.g cp -f hg19.gz data) |
| $ cp -i | Interactive and overwrite copying |
| $ cp -l | Copy files with linking together |
| $ cp -L | Copy the symbolic links |
| $ cp -n | Copy without overwrite |
| $ cp -R | Copying all files including the hidden files |
| $ cp -u | Copy with updating ($ cd -u \* data means update all files in current directory with coping newer files to destination directory) |
| $ cp -v | Copy with verbose |
| $ cp dnaseq.gz data | Copy single file to destination directory |
| $ cp dnaseq.gz hg19.gz /home/user | Copy 2 files (dnaseq.gz and hg19.gz) to the absolute path directory /home/user |
| $ cp \*.gz data | Copy all .gz files in the current subdirectory data |
| $ cp data /home/usr | Copy directory data to the absolute path directory /home/usr |
| $ cp -u | Copying everything including the hidden files |

1.4 What does the command pwd do?

pwd stands for print working directory tell us which directory I am currently located at (e.g $ /home/ubuntu). It is a LINUX/UNIX command to locate the current working directory. It is used to print the full path of the current working directory starting from the root. Another example: learner@host01:~$ pwd

/home/learner

$ pwd command is used when you want to make sure that you are in your home 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)  
I have to use the command ls, which is a LINUX shell command that lists all contents and details of files and directories. The list option here is to use ls -l to list file details in a directory such as date, size, time, and access permission (e.g $ ls -l). Another example:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ ls -l

total 6089980

-rw-rw-r-- 1 ubuntu ubuntu 5694450 Apr 3 17:54 bad\_reads.txt

-rw-rw-r-- 1 ubuntu ubuntu 2328336161 Apr 3 17:47 NGS0001.R1.fastq

-rw-rw-r-- 1 ubuntu ubuntu 778118175 Feb 28 2017 NGS0001.R1.fastq.gz

-rw-rw-r-- 1 ubuntu ubuntu 2328336161 Apr 3 17:50 NGS0001.R2.fastq

-rw-rw-r-- 1 ubuntu ubuntu 795624686 Feb 28 2017 NGS0001.R2.fastq.gz

|  |  |
| --- | --- |
| **ls Command Options Examples** | **Description** |
| $ ls dnaseq/data | List directory with relative path |
| $ ls /home/usr/dnaseq/data | List directory with absolute path |
| $ ls / | List root directory |
| $ ls .. | List parent directory |
| $ ls ~ | List user’s home directory |
| $ ls \* | List all subdirectories |
| $ ls -R | List recursive directory tree or list all file in subdirectories |
| $ ls -a | List all files including the hidden files |
| $ ls -d | List all directories |
| $ ls -i | List files that have index number |
| $ ls -l | List files with long format and show permission |
| $ ls -la | List files with long format including hidden files |
| $ ls -lh | List files with long format and with readable format file size |
| $ ls -ls | List files with long format and with file size |
| $ ls -lt | List files with long format and sorted by time and date |
| $ ls -r | List files in reverse order |
| $ ls -n | List files in numeric order |
| $ ls -s | List file size |
| $ ls -S | Sort by file size |
| $ ls -t | Sort by time and date |
| $ ls -X | Sort by extension name |
| $ ls \*.txt | List text files only |
| $ ls > output.txt | Redirection to output file |
| $ ls -d \*/ | List directories only |
| $ ls -d $pwd/\* | List all files and directories with full path |
| $ ls --version | To check the version of ls command |
| $ ls --help | To get the help page of ls command |

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

I have to use head command to print, display, or type the first 15 lines of all files in text format calling \*.txt (e.g $ head -15 \*.txt). There are other commands that used to print other type of files such as /dnaseq/data:

1. sed command (e.g sed -n 1, 15p /dnaseq/data),
2. awk command (e.g awk ‘FNR <= 15’ /dnaseq/data), and
3. perl command (e.g perl -ne`1..15 and print` /dnaseq/data

1.7 How do you rename a file from new to old? (please provide an example command)  
mv command stands for move and it is used to move, copy or rename only one file at a time to a single file or directory (e.g $ mv newfile.txt oldfile.txt). To rename a file from new to old, I should specify the source and destination of a single file and then follow this: type mv, a space, name of the new file, a space, and name of the old file. Then press enter!

There is also rename command that used to rename multiple files. Here, we must install first rename by using this command: $ sudo apt install rename (then use: $ rename \*.html \*.txt).

1.8 How do you display the contents of a file myfile.txt? (please provide an example command)  
type command is used to display the contents of a text file without modifying it (e.g $ type myfile.txt). Write type, a space, myfile.txt and then press enter!

cat command stands for concatenate and it is used to display and visualize the contents of a file (e.g $ cat myfile.txt). It is also used to combine multiple files to one file.

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

mkdir stands for make directory. It is used to make or create one or more new directories (e.g $ mkdir flower).

1.10 How do you change the current directory to /usr/local/bin? (please provide an example command)  
I have to use cd command to change current directory to bin directory as it shown in this example: (e.g $ 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)

I have to use ls -a command to display a list of all files in the current directory, including the hidden files (e.g $ ls -a \*.fastq.gz).

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

I have to use cd .. command. It is used to navigate from the current directory to the previous or parent directory. It is up one directory level:

(e.g ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ cd .. and then press enter. This will take us to ubuntu@mai:~/ngs\_course/dnaseq/data).

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

I have to use mkdir command to create a subdirectory in my home directory:

e.g ubuntu@mai:~/ngs\_course/dnaseq/data$ mkdir untrimmed\_fastq

ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq

We can use this command: e.g $ mkdir -p ngs\_course/dnaseq, if I want to create multiple subdirectories in my home directory, the command will be like this: $ mkdir -p ngs\_course/dnaseq/data. We have to add -p here to make sure that mkdir adds any missing parent directory to my home directory.

e.g ubuntu@mai:~/ngs\_course/dnaseq/ data $ mkdir ~/ngs\_course/dnaseq/ data /untrimmed\_fastq

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

head -1 command is used to list the first line in a text file (myfile.txt) (e.g $ head -1 myfile.text).

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

tail command is used to display or print the last number of lines of a certain file (e.g $ tail file1.txt), which can print the last 10 lines of our text file1 without certain option (10 lines by default). If there is a certain number of lines the command will be like this: $ tail -n file1.txt. Here, we use the command like this: $ tail -1 file1.txt

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

cut is a command line used in LINUX/UNIX. It is used to extract sections from file (e.g $ cut file.txt). Here, cut command is used to extract a column from a file in text format. To specify this character, -c will be used like this: $ cut -c file.txt. Another example is:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data$ cut -f1,2,3 annotation.bed | head -n 5

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

cp -r is a command that used to copy a directory or folder from one place to another (e.g $ 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 command line, which is used for searching or finding string in a text file. In UNIX, grep stands for searching globally through a file for a regular expression and then printing those lines. The command can be like this: $ grep Hypertension diseases.txt

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

I have to use the command ls -a (e.g $ ls -a). This will help us to view all hidden filles in home directory. ubuntu@mai:~/ngs\_course/dnaseq/data$ ls -a

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

(1) We have to use the command screen (e.g $ screen). To detach a screen from remote terminal, we have to press ctrl+a together immediately followed by d. Now, we can come back to the terminal and see the message that the screen detached. Then, we can safely logout and the session will be alive and not dead. If we want to resume a detached screen, we can login and type screen -r (e.g $ screen -r) / (e.g $ screen -S myjobname and $ screen -R myjobname).

Other commands are used to keep running even after we logged out such as:

(2) $ nohub

(3) $ tmux / $ tmux detach / $ snux attach

(4) $ disown %1

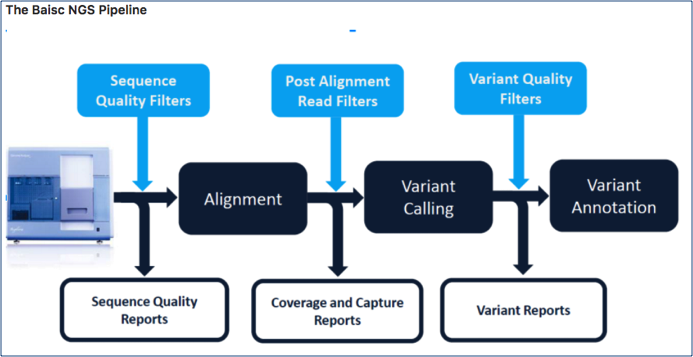
(5) $ setsid

## 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)

# We have to download the raw\_fastq data (raw reads) and bed file, we have to use the following commands:

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R1.fastq.qz

This will provide NGS0001.R1.fastq.qz file.

Then, changed to NGS0001.R1.fastq.gz file

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R2.fastq.qz

This will provide NGS0001.R2.fastq.qz file.

Then, changed to NGS0001.R2.fastq.gz file

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/annotation.bed

This will provide annotation.bed file.

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

# We have to download the raw\_fastq data (raw reads) and bed file, we have to use the following commands:

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R1.fastq.qz

This will provide NGS0001.R1.fastq.qz file.

Then, changed to NGS0001.R1.fastq.gz file

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/NGS0001.R2.fastq.qz

This will provide NGS0001.R2.fastq.qz file.

Then, changed to NGS0001.R2.fastq.gz file

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/annotation.bed

This will provide annotation.bed file.

# To download the human reference genome for mapping, we have to use the following command:

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

# Installing Anaconda will help us to install other tools such as trimmomatic, Fastqc, Samtools, Picard, Bedtools, BWA, Freebayes and any other tools required to make our pipeline:

$ wget https://repo.anaconda.com/archive/Anaconda3-2020.02-Linux-x86\_64.sh

# These are the tools required to make our pipeline:

$ conda install -c bioconda samtools

$ conda install samtools

$ conda install -c bioconda bwa

$ conda install bwa

$ conda install -c bioconda freebayes

$ conda install freebayes

$ conda install -c bioconda picard

$ conda install picard

$ conda install -c bioconda bedtools

$ conda install bedtools

$ conda install -c bioconda trimmomatic

$ conda install trimmomatic

$ conda install -c bioconda fastqc

$ conda install fastqc

$ conda install -c bioconda vcflib

$ conda install vcflib

# Before to use the command tabix for variant calling data, we have to install it. The command below is used to install tabix:

$ sudo apt install tabix

# To Download the ANNOVAR software for annotation of variants, we generate this command:

$ wget <http://www.openbioinformatics.org/annovar/download/0wgxR2rIVP/annovar.latest.tar.gz>

#The following commands were used to download the databases that used for annoation the variants via ANNOVAR software:

$ ./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/

# To Download snpEFF software to annotate the variants data, we have to use this command to download it:

$ wget <https://snpeff.blob.core.windows.net/versions/snpEff_latest_core.zip>

***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)

# These are all the command lines necessary to download and organize the input data during pre-alignment stage, which include:

$ pwd

/home/ubuntu

$ mkdir ngs\_course

$ mkdir ngs\_course/dnaseq

$ cd ngs\_course/dnaseq

$ mkdir data meta results logs

$ ls -lF

$ cd ~/ngs\_course/dnaseq/data

$ mkdir untrimmed\_fastq

$ mkdir trimmed\_fastq

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

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mv NGS0001.R2.fastq.qz NGS0001.R2.fastq.gz

$ mv \*fastq.gz ~/ngs\_course/dnaseq/data/untrimmed\_fastq

$ mv annotation.bed ~/ngs\_course/dnaseq/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

$ wget https://s3-eu-west-1.amazonaws.com/workshopdata2017/annotation.bed

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

$ mv hg19.fa.gz ~/ngs\_course/dnaseq/data

$ cd

$ nano README.txt

$ ls

$ cd ~/

$ 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

$ cd ~/ngs\_course/dnaseq/data/untrimmed\_fastq

$ zcat NGS0001.R1.fastq.gz > NGS0001.R1.fastq

$ mkdir ../other/

$ mv bad\_reads.txt ../other/

$ cp annotation.bed annotation\_plus5.bed

$ head -5 annotation.bed >> annotation\_plus5.bed/

$ cat annotation\_plus5.bed | wc -l

$ sort -u annotation\_plus5.bed | wc -l

**To check the quality of our input data:**

# This section is for assessing the quality of our data using fastQC tools:

# The main functions of FastQC are:

(1) Import of data from BAM, SAM or FastQ files (any variant)

(2) Providing overview to tell us in which areas there may be problems

(3) provide us with summary graphs and tables to quickly assess our data

(4) Export of results to an HTML based permanent report

(5) Offline operation to allow automated generation of reports without running the interactive application

# Quality control of fastq files is a critical first step in the analysis of our NGS data

# To run fastQC, we have to change directory to untrimmed\_fastq directory using the command below:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ cd ~/ngs\_course/dnaseq/data/untrimmed\_fastq

# We have to use the command Fastqc to check the quality control of all fastq.gz files

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ fastqc \*.fastq.gz

# We used the multi-threading functionality of FastQC to run 4 jobs at once

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ cd ~/ngs\_course/dnaseq/data/untrimmed\_fastq

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ fastqc -t 4 \*.fastq.gz

# To know more about fastqc command, we run the command below to help us understand the arguments

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ fastqc –help

# We have make a home for our results called fastqc\_untrimmed\_reads and move them there.

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mkdir ~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads

mkdir: cannot create directory ‘/home/ubuntu/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads’: File exists

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mv \*fastqc\* ~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads/

# In results directory:

# We used the command below to show us the contents and size of fastqc\_untrimmed\_reads file

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ ls -lh ~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads/

# The .html files contain the final reports generated by fastqc

# Using FileZilla software, we can transfer the html files over to our personal computer via FileZilla.

# There are are two of the most important analysis modules in FastQC, the "Per base sequence quality" plot and the "Overrepresented sequences" table.

# The "Per base sequence quality" plot provides the distribution of quality scores across all bases at each position in the reads.

# We have to change directory into fastqc\_untrimmed\_reads

# The other output of FastQC is a .zip file, we have to unzip these files

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ cd ~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads/

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ unzip \*.zip

# It didn't work because unzip expects to get only one zip file.

# We have to use shell for loop to iterate through the list of files in \*.zip format and it will run unzip once for each file as shown below:

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ for zip in \*.zip

> do

> unzip $zip

> done

# To list the contents and size of fastqc files:

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ ls -l NGS0001.R1\_fastqc

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ ls -l NGS0001.R2\_fastqc

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ head NGS0001.R1\_fastqc/summary.txt

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ head NGS0001.R2\_fastqc/summary.txt

# To save all generated records, we have to use the command cat for all fastqc summary.txt files to save it into one full\_report.txt:

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ cat \*/summary.txt > ~/ngs\_course/dnaseq/logs/fastqc\_summaries.txt

**# Quality control pre-alignment and before trimming are shown in the links below (fastQC Reports):**

1. [NGS0001.R1.fastq.gz FastQC Report](file:///E:\Advanced%20Bioinformatics%20Assignment\Advanced%20Bioinformatics%202019%20assessment_files\NGS0001.R1_fastqc\fastqc_report.html)
2. [NGS0001.R2.fastq.gz FastQC Report](file:///E:\Advanced%20Bioinformatics%20Assignment\Advanced%20Bioinformatics%202019%20assessment_files\NGS0001.R2_fastqc\fastqc_report.html)

**Quality Control\_Trimming:**

# To improve the quality of fastq reads, different commands are used to perform trimming stratigies. Trimming is important to help in allignment with the reference genome and remove all bad bases and any contaminating vector or adapter sequences from the reads as well as filter out poor quality score reads.

# We have to run our raw reads through FastQC to assess the quality of our sequencing reads (R1 and R2 reads). Now we are going to improve the quality of our reads, by trimming off any "bad" bases using the following trimmomatic commands (for paired-end fastq files):

(base) ubuntu@mai:~/ngs\_course/dnaseq/results/fastqc\_untrimmed\_reads$ cd ~/ngs\_course/dnaseq/data/untrimmed\_fastq

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ trimmomatic PE \

> -threads 4 \

> -phred33 \

> /home/ubuntu/ngs\_course/dnaseq/data/untrimmed\_fastq/NGS0001.R1.fastq.gz /home/ubuntu/ngs\_course/dnaseq/data/untrimmed\_fastq/NGS0001.R2.fastq.gz \

> ILLUMINACLIP:/home/ubuntu/anaconda3/pkgs/trimmomatic-0.39-hdfd78af\_2/share/trimmomatic-0.39-2/adapters/NexteraPE-PE.fa:2:30:10 \

> TRAILING:25 MINLEN:50

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ trimmomatic PE -threads 4 -phred33 /home/ubuntu/ngs\_course/dnaseq/data/untrimmed\_fastq/NGS0001.R1.fastq.gz /home/ubuntu/ngs\_course/dnaseq/data/untrimmed\_fastq/NGS0001.R2.fastq.gz -baseout /home/ubuntu/ngs\_course/dnaseq/data/trimmed\_fastq/NGS0001\_trimmed\_R ILLUMINACLIP:/home/ubuntu/anaconda3/pkgs/trimmomatic-0.39-hdfd78af\_2/share/trimmomatic-0.39-2/adapters/NexteraPE-PE.fa:2:30:10 TRAILING:25 MINLEN:50

# The above result is the trimmomatic output in the terminal.

**# Quality control post-alignment and after trimming are shown in the links below (fastQC Reports):**

1. [trimmed\_data\_1P FastQC Report](file:///E:\Advanced%20Bioinformatics%20Assignment\Advanced%20Bioinformatics%202019%20assessment_files\trimmed_data_1P_fastqc.html)
2. [trimmed\_data\_2P FastQC Report](file:///E:\Advanced%20Bioinformatics%20Assignment\Advanced%20Bioinformatics%202019%20assessment_files\trimmed_data_2P_fastqc.html)

### 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)
* Perform duplicate marking (2pts)
* Quality Filter the duplicate marked BAM file (2pts)
* Generate standard alignment statistics (i.e. flagstats, idxstats, depth of coverage, insert size) (4pts)

**Alignment with bwa and duplicate marking with picard tools:**

**# For read alignment:**

# Following trimming, the following steps are used to align sequence reads to the reference genome using alignment method BWA.

# Run the commands: $ bwa and $ bwa mem for read alignment

# We need to index the genome with `bwa index' command.

# bwa is a software package for mapping DNA sequences against a large reference genome, such as the human genome. It consists of three algorithms: BWA-backtrack, BWA-SW and BWA-MEM. The first algorithm is designed for Illumina sequence reads up to 100bp, while the rest two for longer sequences ranged from 70bp to a few megabases. BWA-MEM and BWA-SW share similar features such as the support of long reads and chimeric alignment, but BWA-MEM, which is the latest, is generally recommended as it is faster and more accurate. BWA-MEM also has better performance than BWA-backtrack for 70-100bp Illumina reads.

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ bwa

# For bwa mem algorithm:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ bwa mem

# The reference index, BWA will generate several files in ~/ngs\_course/dnaseq/data/reference/ directory

# For making the index step, we have to generate the following steps of different commands:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mkdir -p ~/ngs\_course/dnaseq/data/reference

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mv ~/ngs\_course/dnaseq/data/hg19.fa.gz ~/ngs\_course/dnaseq/data/reference/

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ bwa index ~/ngs\_course/dnaseq/data/reference/hg19.fa.gz

# To list the different type of references. we run the command below:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ ls ~/ngs\_course/dnaseq/data/reference

hg19.fa.gz hg19.fa.gz.amb hg19.fa.gz.ann hg19.fa.gz.bwt hg19.fa.gz.pac hg19.fa.gz.sa

# For read group (RG) information, run bwa mem with RG information using the following commands:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/untrimmed\_fastq$ mkdir ~/ngs\_course/dnaseq/data/aligned\_data

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ bwa mem -t 4 -v 1 -R '@RG\tID:11V6WR1.111.HD1375ACXX.1.NGS0001\tSM:NGS0001\tPL:ILLUMINA\tLB:nextera\tDT:2017-02-28\tPU:11V6WR1' -I 250,50 ~/ngs\_course/dnaseq/data/reference/hg19.fa.gz ~/ngs\_course/dnaseq/data/trimmed\_fastq/NGS0001\_trimmed\_R\_1P ~/ngs\_course/dnaseq/data/trimmed\_fastq/NGS0001\_trimmed\_R\_2P > ~/ngs\_course/dnaseq/data/aligned\_data/NGS0001.sam

# To change directory into aligned\_data, we used the commands below and will help us to convert sam file to bam file format, sort it and generate an index using samtools:

(base) ubuntu@mai:~$ cd ~/ngs\_course/dnaseq/data/aligned\_data

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools view -h -b NGS0001.sam > NGS0001.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools view -h -b -s NGS0001.sam > NGS0001.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools sort NGS0001.bam NGS0001\_sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

NGS0001.bam NGS0001.sam NGS0001\_sorted.bam NGS0001\_sorted.sam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data/NGS0001\_sorted.bam$ cd

(base) ubuntu@mai:~$ source ~/.bashrc

(base) ubuntu@mai:~$ conda install samtools

(base) ubuntu@mai:~$ conda install bwa

(base) ubuntu@mai:~$ conda install picard

(base) ubuntu@mai:~$ conda install trimmomatic

(base) ubuntu@mai:~$ ls

anaconda3 Anaconda3-2020.02-Linux-x86\_64.sh NGS0001.R1.fastq.qz\_fastqc.html ngs\_course README.txt

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools sort NGS0001.bam > NGS0001.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools index NGS0001.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

NGS0001.bam NGS0001.bam.bai NGS0001.sam NGS0001\_sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

NGS0001.bam NGS0001.sam NGS0001\_sorted.bam NGS0001\_sorted.sam sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ cd ~/ngs\_course/dnaseq/data/aligned\_data/NGS0001\_sorted.bam

(base) ubuntu@mai:~$ ls

anaconda3 Anaconda3-2020.02-Linux-x86\_64.sh NGS0001.R1.fastq.qz\_fastqc.html ngs\_course README.txt

(base) ubuntu@mai:~$ cd ~/ngs\_course/dnaseq/data/aligned\_data

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

NGS0001.bam NGS0001.sam NGS0001\_sorted.bam NGS0001\_sorted.sam sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools sort NGS0001.bam > NGS0001\_sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools index NGS0001\_sorted.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

NGS0001.bam NGS0001\_sorted.bam NGS0001\_sorted.bam.bai

**# The commands below are used for post alignment quality control (QC) and filtering. There are 2 steps:**

(1) Using Picard tools to mark duplicated reads and (2) Filtering bam data

**# We use Picard tools to mark duplicated reads:**

This tool examines aligned records in the sam and bam file, where can locate duplicate molecules. All records are then written to the output file with the duplicate records flagged. Two files are generated one is the new bam file with duplicate reads marked and the second is a metrics file summarising the number of duplicate reads found.

# We have to mark duplicates first and then filtering the bam file data.

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ picard MarkDuplicates I=NGS0001\_sorted.bam O=NGS0001\_sorted\_marked.bam M=marked\_dup\_metrics.txt

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools index NGS0001\_sorted\_marked.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools index NGS0001\_sorted\_marked.bam

**# Filtering bam data dependent on mapping quality and bitwise flags using samtools:**

# Filtering is based on different criteria:

Minimum MAPQ quality score : 20 -Filter on bitwise flag: yes a. Skip alignments with any of these flag bits because of: (1) The read is unmapped (2) The alignment or this read is not primary alignment (3) The read fails platform/vendor quality checks (4) The read is a PCR or optical duplicate (5) Supplementary alignment required

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools view -F 1796 -q 20 -o NGS0001\_sorted\_filtered.bam NGS0001\_sort

ed\_marked.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools index NGS0001\_sorted\_filtered.bam

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ ls

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ conda install vcflib

### 2.4. Variant Calling (4pts)

* Call Variants using Freebayes restricting the analysis to the regions in the bed file provided (2pt)
* Quality Filter Variants using your choice of filters (2pt)

**Variant\_Calling\_and\_Filtering:**

# This section for variants calling using freebayes software.

#FreeBayes is a Bayesian genetic variant detector designed to find small polymorphisms, specifically SNPs (single-nucleotide polymorphisms), indels (insertions and deletions), MNPs (multi-nucleotide polymorphisms), and complex events (composite insertion and substitution events) smaller than the length of a short-read sequencing alignment.

# This process requires two inputs: a FASTA reference sequence, and a bam-format alignment file by reference position.

# The output reports positions which it finds polymorphic in variant calling file (VCF) format.

# To convert to text format the reference (as required by samtools faidx) and index it with samtools faidx, we have to use the following commands:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ zcat ~/ngs\_course/dnaseq/data/reference/hg19.fa.gz > ~/ngs\_course/dnaseq/data/reference/hg19.fa

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ samtools faidx ~/ngs\_course/dnaseq/data/reference/hg19.fa

**# The following commands used for calling variants with Freebayes tools and then compress the output (the results are variants in vcf file):**

((base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ freebayes --bam ~/ngs\_course/dnaseq/data/aligned\_data/NGS0001\_sorted\_filtered.bam --fasta-reference ~/ngs\_course/dnaseq/data/reference/hg19.fa --vcf ~/ngs\_course/dnaseq/results/NGS0001.vcf

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ bgzip ~/ngs\_course/dnaseq/results/NGS0001.vcf

# To index the vcf data, we have to use the tabix command as below:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ tabix -p vcf ~/ngs\_course/dnaseq/results/NGS0001.vcf.gz

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ conda install vcflib

**# This section for filtering the vcf data:**

# After we ran Freebayes for variant calling tools with default parameters, which is resulting in vcf file that contains a number of "bad" calls and we have to filter it to provide high quality data.

# This is to explain how the vcf quality performed: "QUAL=probability that there is a polymorphism at the loci described by the record: 1 - P(locus is homozygous given the data). AO=Alternate allele observations, SAF=Number of alternate observations on the forward strand, SAR=Number of alternate observations on the reverse strand, RPL=Reads Placed Left: number of reads supporting the alternate balanced to the left (5’) of the alternate allele, RPR=Reads Placed Right: number of reads supporting the alternate balanced to the right (3’) of the alternate allele"

# This can be performed via this command: $ vcffilter -f "QUAL > 1 & QUAL / AO > 10 & SAF > 0 & SAR > 0 & RPR > 1 & RPL > 1". That means: "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".

# The command for filtering variants is here:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ vcffilter -f "QUAL > 1 & QUAL / AO > 10 & SAF > 0 & SAR > 0 & RPR > 1 & RPL > 1" ~/ngs\_course/dnaseq/results/NGS0001.vcf.gz > ~/ngs\_course/dnaseq/results/NGS0001\_filtered.vcf

# The bed file chr22.genes.b37.bed describes the exome sequences and genes that targeted.

# Using bedtools we can filter the vcf file for the regions in chr22.genes.b37.bed. The output file is compressed by bgzip command and then indexed using tabix command as below:

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ bedtools intersect -header -wa -a ~/ngs\_course/dnaseq/results/NGS0001\_filtered.vcf -b ../annotation.bed > ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.vcf

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ bgzip ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.vcf

(base) ubuntu@mai:~/ngs\_course/dnaseq/data/aligned\_data$ tabix -p vcf ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.vcf.gz

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

* Annotate variants using **ANNOVAR** (4pt) and **snpEFF** (4pt)
* Perform basic variant prioritization: filter to exonic variants not seen in dbSNP (2pts)

**Annotation and Prioritization of variants using ANNOVAR and snpEFF:**

# To Download the ANNOVAR software for annotation of variants, we generate this command:

(base) ubuntu@mai:~$ wget <http://www.openbioinformatics.org/annovar/download/0wgxR2rIVP/annovar.latest.tar.gz>

(base) ubuntu@mai:~$ tar -zxvf annovar.latest.tar.gz

#The following commands were used to download the databases that used for annoation the variants via ANNOVAR software:

(base) ubuntu@mai:~$ 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/

# The following command used to generate VCF file through ANNOVAR software (including all information about indels, transitions, transversions for 1 sample):

(base) ubuntu@mai:~/annovar$ ./convert2annovar.pl -format vcf4 ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.vcf.gz > ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.avinput

# To run the ANNOVAR table function, we generate the following command and the results and output in csv format:

(base) ubuntu@mai:~/annovar$ ./table\_annovar.pl ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.avinput humandb/ -buildver hg19 -out ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22 -remove -protocol refGene,ensGene,clinvar\_20180603,exac03,dbnsfp31a\_interpro -operation g,g,f,f,f -otherinfo -nastring . -csvout

# wANNOAVR is a very fast and easy tool to annotate functional genetic variation from high throughput data. It is web-based access to the ANNOVAR software tools.

# Here, I used wANNOVAR to annotate the variants, which is based on vcf file to upload and my university e-mail. The filtered variant data is generated as csv format and viewed by excel for both genome and exome output (query.output.genome\_summary.csv and query.output.exome\_summary.csv). This step helped me to answer the last question about basic variant prioritization for filtering to exonic variants not seen in dbSNP.

# To Download snpEFF software to annotate the variants data, we have to use this command and make directory first:

(base) ubuntu@mai:~$ mkdir snpEFF

(base) ubuntu@mai:~$ cd snpEFF

(base) ubuntu@mai:~/snpEFF$ wget <https://snpeff.blob.core.windows.net/versions/snpEff_latest_core.zip>

# To unzip the latest version of snpEFF\_latest\_core.zip, we run the command below:

(base) ubuntu@mai:~/snpEFF$ unzip snpEff\_latest\_core.zip

# To generate the variants calling format file using snpEFF, we used these commands:

(base) ubuntu@mai:~/snpEFF$ cd /home/ubuntu/ngs\_course/dnaseq/results

(base) ubuntu@mai:~/ngs\_course/dnaseq/results$ unzip NGS0001\_filtered\_chr22.vcf.gz

(base) ubuntu@mai:~/ngs\_course/dnaseq/results$ cd

(base) ubuntu@mai:~$ cd snpEFF/

(base) ubuntu@mai:~/snpEFF/snpEff$ sudo du -sh

sudo: unable to resolve host mai

131M .

(base) ubuntu@mai:~/snpEFF/snpEff$ java -Xmx8g -jar snpEff.jar hg19 /home/ubuntu/ngs\_course/dnaseq/results/NGS0001\_filte

red.vcf > /home/ubuntu/ngs\_course/dnaseq/results/NGS0001\_filtered.ann.vcf

(base) ubuntu@mai:~/snpEFF/snpEff$ java -Xmx8g -jar snpEff.jar hg19 ~/ngs\_course/dnaseq/results/NGS0001\_filtered\_chr22.vcf.gz > /home/ubuntu/ngs\_course/dnaseq/results/NGS0001\_filtered.ann.vcf

**# Basic variant prioritization: filter to exonic variants not seen in dbSNP:**

1. There are 110 exonic variants not seen in dbsnp.
2. There are 91 exonic variants not seen in dbsnp (unknown exonic function excluded).
3. There are 23 exonic variants with nonsynonymous exonic function not seen in dbsnp.

## 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)

**Answers are provided in the links below that direct you to my GitHub Account:**

# The link below directs you to my GitHub account, where MarkDown and html file are uploaded:

<https://github.com/Mai-beeb/My_MarkDown_2021>

# Another link to My-MarkDown.md

[Advanced Bioinformatics 2019 assessment](file:///C:\Users\maiel\OneDrive\Desktop\Advanced%20Bioinformatics%202019%20assessment.html)

# The link below directs you to my GitHub account, where Advanced-Bioinformatics-Linux-NGS-R-Assessment is uploaded:

<https://github.com/Mai-beeb/Advanced-Bioinformatics-linux-NGS-R-Assessment-2021>

**Answers are also copied, pasted, and shared below:**

# Advanced Bioinformatics 2019 assessment

#### **m1700635**

#### **05/04/2021**

# Question 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.

#How to use sum integers and numbers in R programming: #There are different ways to do so: #I will use sum command to sum every number in the range 5->55. The answer is 1530 #sum(5:55)

sum(5:55)

## [1] 1530

#We can also find the sum from a few numbers, which determined between 5 and 55 sum(5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55)

sum(5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55)

## [1] 1530

#We can generate a sequence of numbers either named #(specified) using from, by, and to: #seq(5,55,1)

seq(5,55,1)

## [1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

## [26] 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54

## [51] 55

#or unnamed (not specified) using: #seq(by=1, from=5, to=55)

seq(by=1, from=5, to=55)

## [1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

## [26] 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54

## [51] 55

#Since the vector is an ordered list of numbers, we have to use the command #sum x <- c(5:55);sum(x)

sum(x <- c(5:55))

## [1] 1530

# Question 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.

sumfun <- **function**(n) {sum(5:n)}

sumfun(10)

## [1] 45

sumfun(20)

## [1] 200

sumfun(100)

## [1] 5040

# Question 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.

#to calculate the Fibonacci series with n values: #fib <- function(n) #or fib <-numeric(n)

fib <-numeric(12)

#fib of the first number is 1 #fib[1] <- 1

fib[1] <- 1

#fib of the second number is 2 #fib[2] <- 1

fib[2] <- 1

#the loop starts from the third number; that means: #i = 3

i = 3

#we have to repeat the loop to the 12th number of fib

fib <- **function**(n) { *# function for calculating fibonnaci of a vector with n values*

fib <-numeric(n)

fib[1] <- 1

*# fib of the first number is 1*

fib[2] <- 1

*# fib of the second number is 2*

i = 3

*# start the loop at the third element*

**while** (i <= n) { *# repeat the loop for the 12th number of fib*

fib[i]=fib[i - 1] + fib[i - 2]

i<- i + 1

}

**return**(fib) *# return the modified vector*

}

fib(12)

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

# Question 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.

str(mtcars)

## 'data.frame': 32 obs. of 11 variables:

## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...

## $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...

## $ disp: num 160 160 108 258 360 ...

## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...

## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...

## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...

## $ qsec: num 16.5 17 18.6 19.4 17 ...

## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...

## $ am : num 1 1 1 0 0 0 0 0 0 0 ...

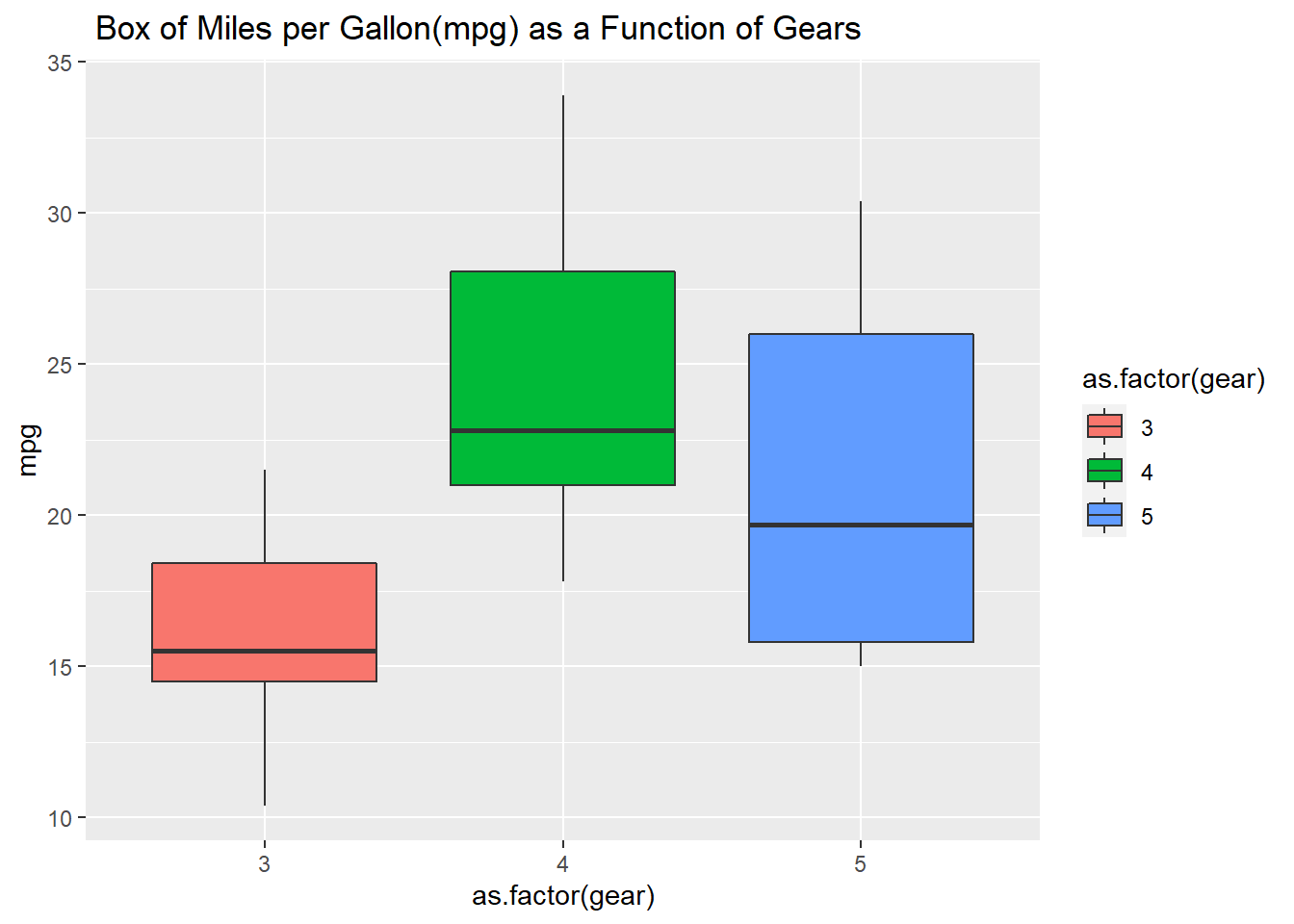
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...

## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...

**library**(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.0.5

ggplot(data = mtcars, aes(x= as.factor(gear), y= mpg)) + geom\_boxplot(aes(fill= as.factor(gear) )) + ggtitle(" Box of Miles per Gallon(mpg) as a Function of Gears")



# Question 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?

#First, we have to set variable for x and y axises. #then, we can set linear model formula and get the summary of model

y <- cars $dist; x <- cars $speed;

model <- lm(formula = "y ~ x")

summary(model)

##

## Call:

## lm(formula = "y ~ x")

##

## Residuals:

## Min 1Q Median 3Q Max

## -29.069 -9.525 -2.272 9.215 43.201

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) -17.5791 6.7584 -2.601 0.0123 \*

## x 3.9324 0.4155 9.464 1.49e-12 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 15.38 on 48 degrees of freedom

## Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438

## F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

#To answer these questions from the coefficient dataset provided above, we can determine: #The fitted slope is 3.9324 #The intercept of the line is -17.5791 #Their standard errors are 6.7584 (for intercept (dist)) #and 0.4155 (for X (speed)) #The units used for the variables in the dataset to #measure the speed and breaking distance is feet

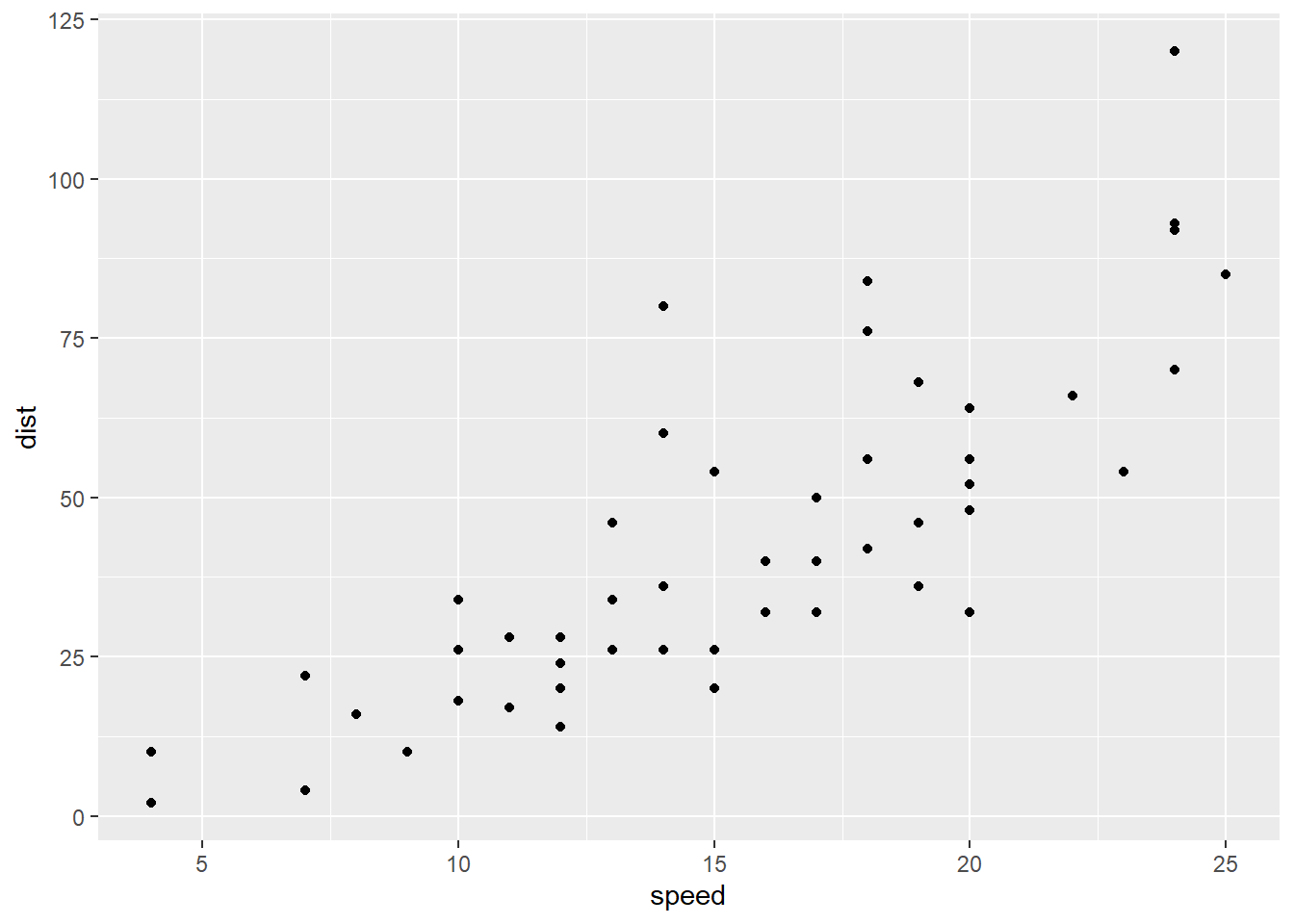
# Question 3.6. Use ggplot to plot the data points from Task 6 and the linear fit.

#we have to download: library(ggplot2) to help us run the plots

**library**(ggplot2)

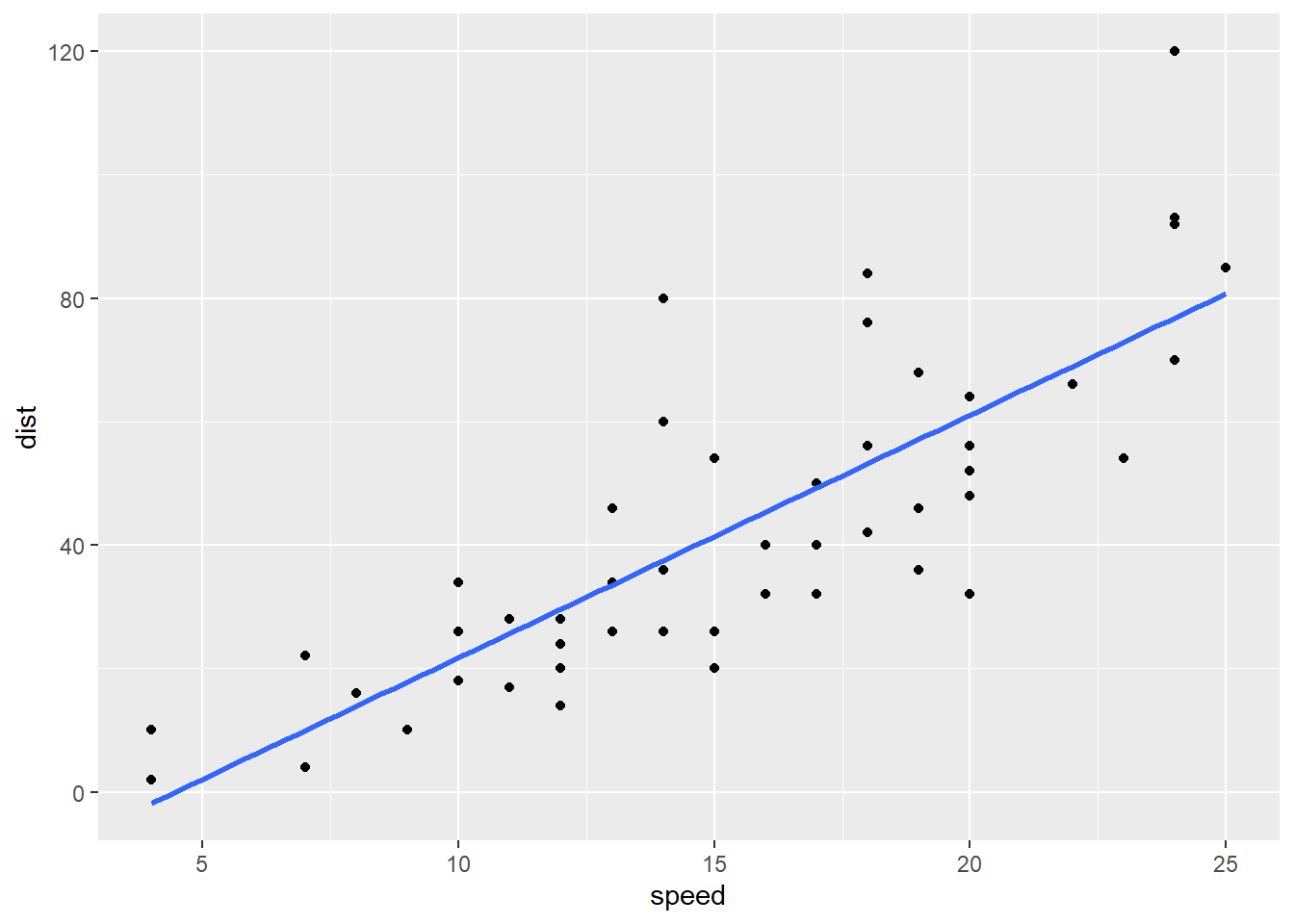
#we have to identify the variables to run the plots

qplot(speed, dist, data = cars, geom ="point")



qplot(speed, dist, data = cars, geom ="point") + geom\_smooth(method= "lm", se = FALSE)

## `geom\_smooth()` using formula 'y ~ x'

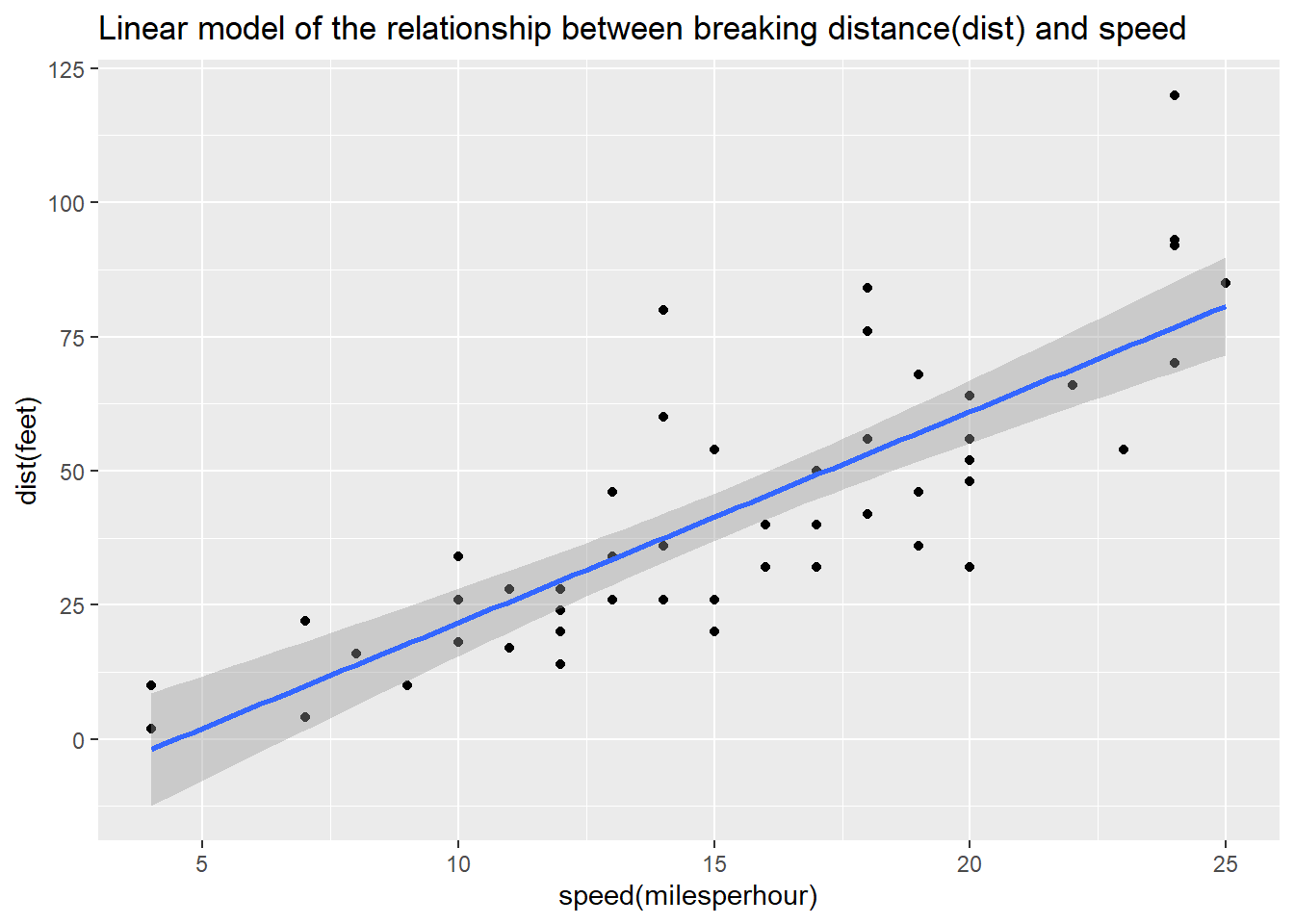


#we have to create new variables named ggplot\_new and add a title, x, and y lables it to the previous plot

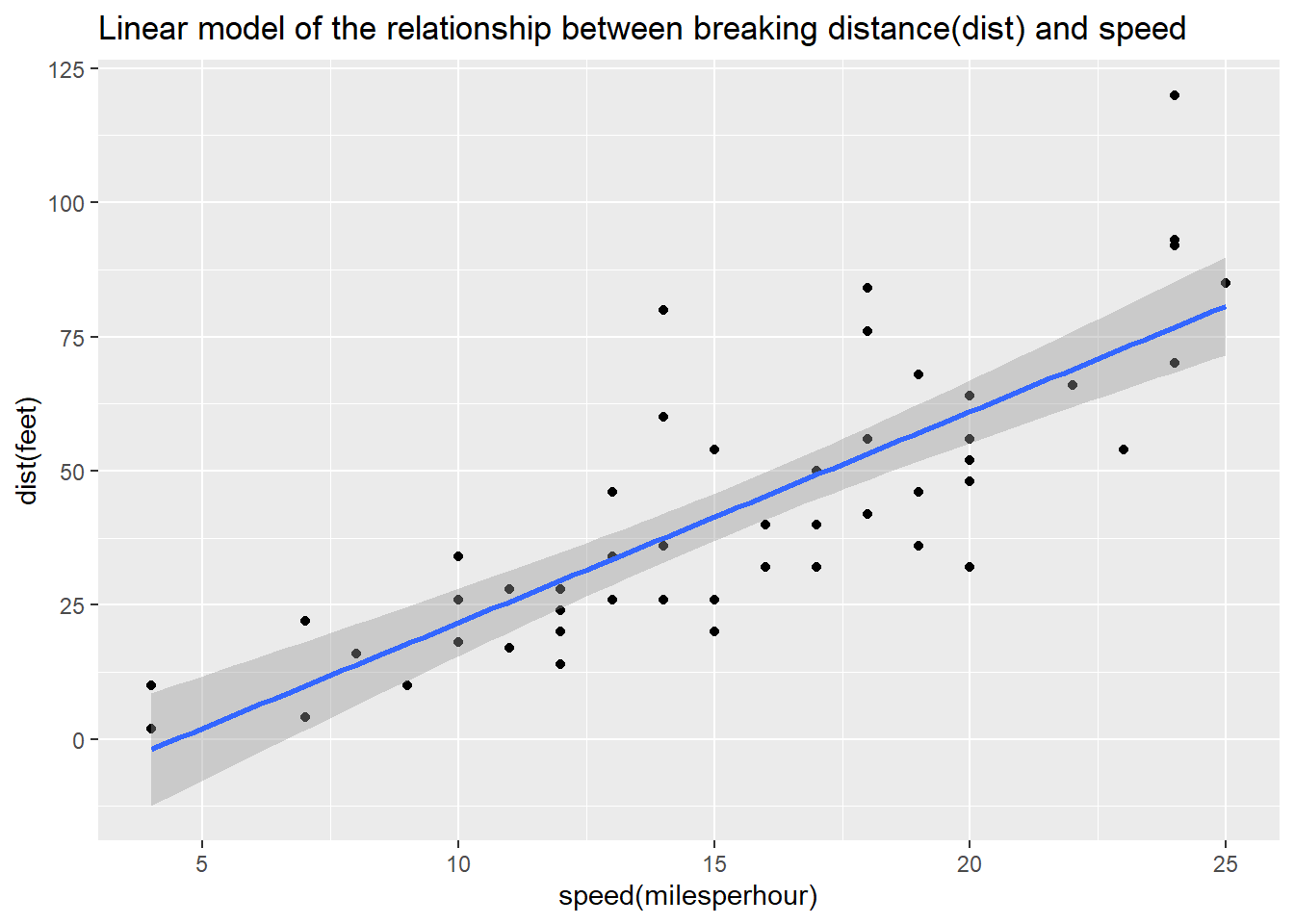
ggplot <- ggplot(data = cars, aes(x= speed, y=dist)) + geom\_point() + geom\_smooth(method = "lm",formula = "y ~ x")

ggplot\_new <- ggplot + ggtitle ("Linear model of the relationship between breaking distance(dist) and speed")+ xlab("speed(milesperhour)")+ ylab("dist(feet)")

ggplot\_new



ggplot\_new



# Question 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.

model$residuals

## 1 2 3 4 5 6 7

## 3.849460 11.849460 -5.947766 12.052234 2.119825 -7.812584 -3.744993

## 8 9 10 11 12 13 14

## 4.255007 12.255007 -8.677401 2.322599 -15.609810 -9.609810 -5.609810

## 15 16 17 18 19 20 21

## -1.609810 -7.542219 0.457781 0.457781 12.457781 -11.474628 -1.474628

## 22 23 24 25 26 27 28

## 22.525372 42.525372 -21.407036 -15.407036 12.592964 -13.339445 -5.339445

## 29 30 31 32 33 34 35

## -17.271854 -9.271854 0.728146 -11.204263 2.795737 22.795737 30.795737

## 36 37 38 39 40 41 42

## -21.136672 -11.136672 10.863328 -29.069080 -13.069080 -9.069080 -5.069080

## 43 44 45 46 47 48 49

## 2.930920 -2.933898 -18.866307 -6.798715 15.201285 16.201285 43.201285

## 50

## 4.268876

View <- lm(dist ~ speed, data = cars)

model <- lm(dist ~ speed, data = cars)

summary(model)

##

## Call:

## lm(formula = dist ~ speed, data = cars)

##

## Residuals:

## Min 1Q Median 3Q Max

## -29.069 -9.525 -2.272 9.215 43.201

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) -17.5791 6.7584 -2.601 0.0123 \*

## speed 3.9324 0.4155 9.464 1.49e-12 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 15.38 on 48 degrees of freedom

## Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438

## F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

cars$residuals <- model$residuals

View(cars)

cars$spridicted <- model$fitted.values

predict(model, data.frame(speed = c(12.5, 15.5, 17)))

## 1 2 3

## 31.57601 43.37324 49.27185

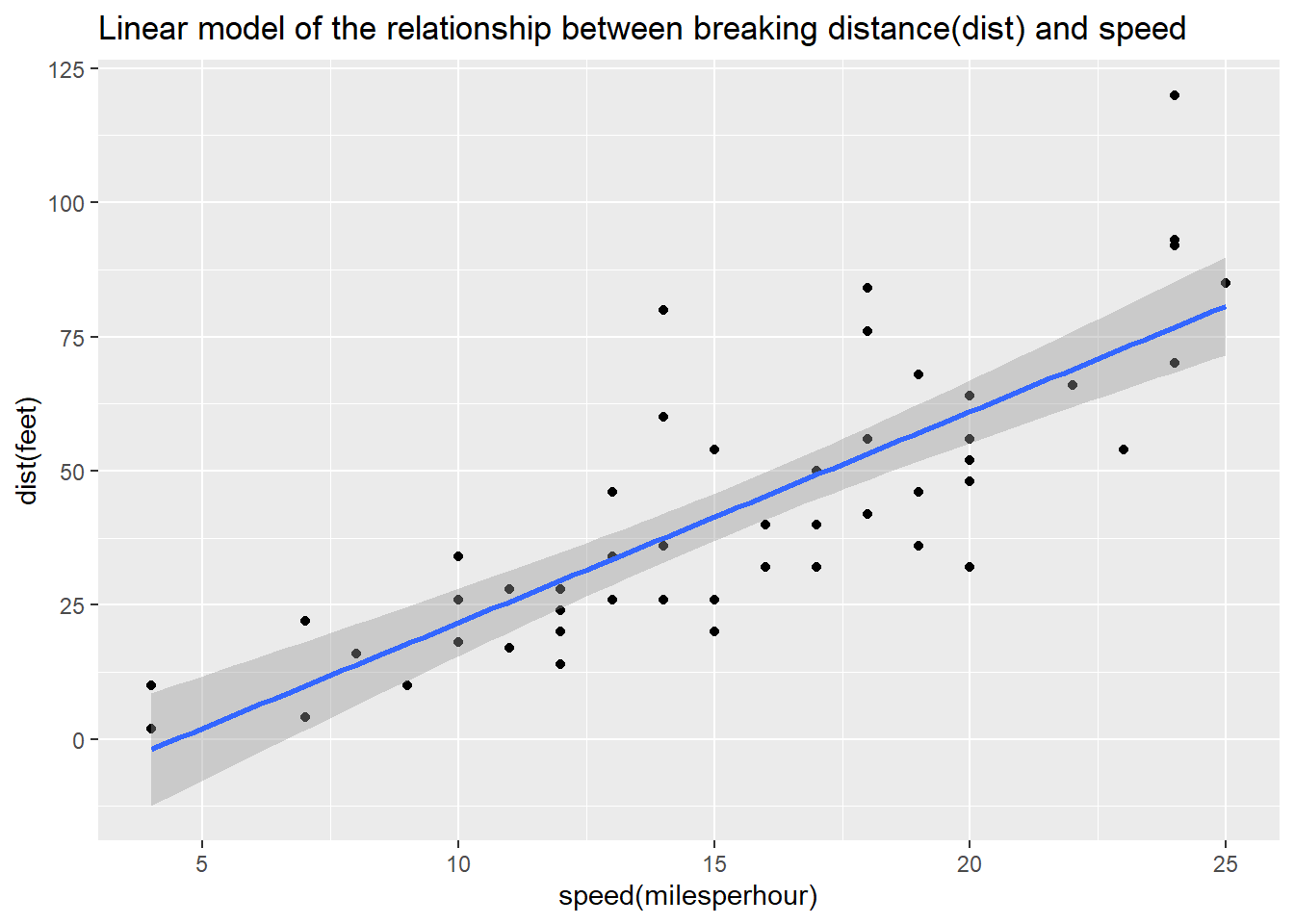
predict(model, data.frame(speed = c(12, 15, 18)))

## 1 2 3

## 29.60981 41.40704 53.20426

ggplot <- ggplot(data = cars, aes(x= speed, y=dist)) + geom\_point() + geom\_smooth(method = "lm",formula = "y ~ x")

ggplot\_new



#First: we have to idenify the variables: #breaking distance in miles = dist\_m #one foot in miles = 0.000189394 #we have convert distance to miles #dist\_m <- cars$dist\*0.00018939 #Second: we have to idenify the variables: #speed in miles per hour = speed\_m\_h #the breaking distance is propotional to square of #the speed: speed\_m\_h <- cars $speed^2 #Now ready for the formula of linear model (lm): #lm(formula = dist\_m ~ speed\_m\_h)

dist\_m <- cars$dist\*0.000189394

speed\_m\_h <- cars $speed^2

lm(formula = dist\_m ~ speed\_m\_h)

##

## Call:

## lm(formula = dist\_m ~ speed\_m\_h)

##

## Coefficients:

## (Intercept) speed\_m\_h

## 1.678e-03 2.443e-05

#since the distance and speed are constant and the #slope means half the reaction time, we have to #convert the reaction time in one hour to seconds: #16060 = 3600 seconds #reaction\_time <- 2.443e-05\*2 #the slope reaction time is twice or double the #reaction time

reaction\_time <- 2.443e-05\*2

converted\_reaction\_time <- reaction\_time/3600

converted\_reaction\_time

## [1] 1.357222e-08

#the results are not significant for the reaction #time and they are negative for the reaction time. #the results expected to be proportional and double #the value of the slope reaction\_time

**library**(ggplot2)

plot\_1 <- ggplot(data = cars, aes(speed\_m\_h, dist\_m))+geom\_point()

plot\_2 <- plot\_1 + geom\_smooth(method = "lm", formula = dist\_m ~ speed\_m\_h)

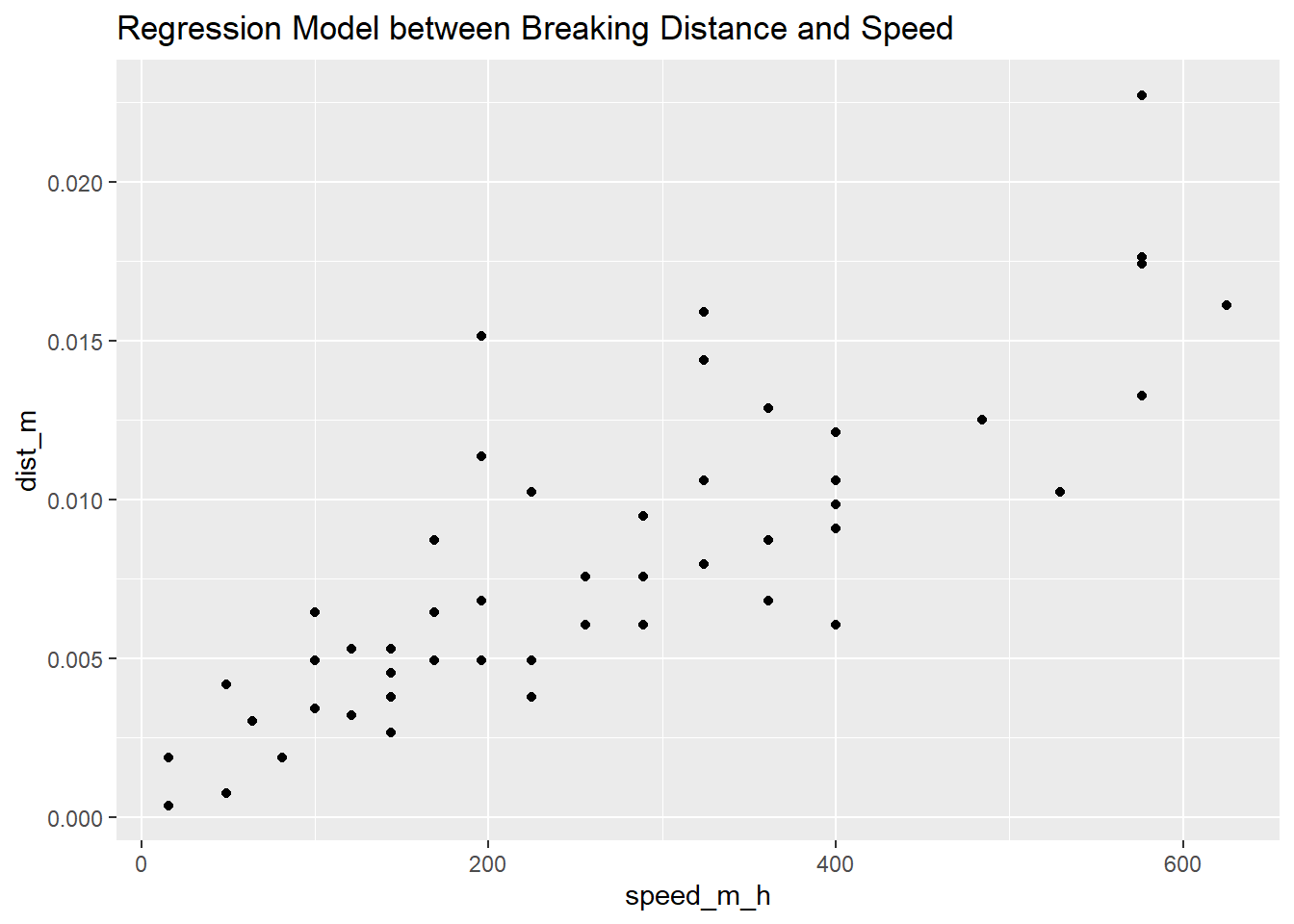
plot\_3 <- plot\_2 + ggtitle("Regression Model between Breaking Distance and Speed")

plot\_3

## Warning: 'newdata' had 80 rows but variables found have 50 rows

## Warning: Computation failed in `stat\_smooth()`:

## arguments imply differing number of rows: 80, 50



**References:**

<https://www.r-graph-gallery.com/>

<https://www.tidyverse.org/>

<https://cran.r-project.org/doc/manuals/r-release/R-intro.html>

<https://www.r-graph-gallery.com/>

<https://www.markdown.rstudio.com>

<https://www.rstudio.com/products/RStudio/>

Lecture Notes