2022/01
e Training Mannual

Online Workshop on Life science meets Programming

13 - 15 September 2022

TRAINING MANUAL

Hands-on

Organized by

Bioinformatics Centre
ICAR-Indian Institute of Spices Research
Kozhikode, Kerala

Online workshop on “Life science meets Programming”
September 13-15, 2022

Training Manual: Hands-on Exercise

Compiled by

Sona Charles

Mukesh Sankar S

Jayarajan K

Fayad M A

Organized by

Bioinformatics Centre,

ICAR-Indian Institute of Spices Research,

Kozhikode, Kerala, India

2022

Published by

Dr. CK Thankamani
Director, ICAR- Indian Institute of Spices Research

Citation:

Charles.S *et al.* (2022) Life science meets programming –Training Manual: Hands-on exercise. ICAR-Indian Institute of Spices Research, Kozhikode, Kerala, India (pp.)

Manuscript No.: Training Manual 2022/01

Workshop Convenor

Ms. Sona Charles,
Scientist (Agricultural Bioinformatics),
Bioinformatics Centre,
ICAR- Indian Institute of Spices Research,
Kozhikode, Kerala-673012.

Workshop co-convenors

Mr. Mukesh Sankar. S,
Scientist (Crop Improvement & Biotechnology),
ICAR- Indian Institute of Spices Research,
Kozhikode, Kerala-673012.

Mr. Jayarajan. K,
Chief Technical Officer,
ICAR- Indian Institute of Spices Research,
Kozhikode, Kerala-673012.

Published by:

ICAR-Indian Spices Research Institute, Kozhikode, Kerala.
http://www.spices.res.in/

ONLINE WORKSHOP ON "Life science meets Programming"

PROGRAM SCHEDULE

Day 1: 13-09-2022		
10:00 am	Welcome Address	Dr. Mukesh Sankar S, Scientist, ICAR-Indian Institute of Spices Research
10:10 am	Introductory Remarks and Release of Training Manual	Dr. CK Thankamani, Director, ICAR-Indian Institute of Spices Research
10:20 am	Felicitations	Dr. KV Saji, Head, Crop Improvement and Biotechnology Division, ICAR-Indian Institute of Spices Research
10:25 am	Introduction to the course and vote of thanks	Ms. Sona Charles, Scientist (Bioinformatics), ICAR-Indian Institute of Spices Research
Pre-workshop evaluation and photo session		
11:00 am	Inaugural Lecture	**"Coding for decoding secrets of life"** Dr. Santhosh J Eapen, Former Director, ICAR- Indian Institute of Spices Research
12:15 pm	Setting up the computer	Dr. Mukesh Sankar S Mr. Jayarajan Mr. Fayad M
02:00 pm	Utilities in Bioinformatics	Ms. Sona Charles Scientist (Bioinformatics), ICAR- Indian Institute of Spices Research
03:30 pm	Introduction to R	Dr. Mukesh Sankar S, Scientist (Plant Breeding), ICAR-Indian Institute of Spices Research
Day 2: 14-09-2022		
10:00 am	Data Visualization using R	Ms. Sona Charles
02:00pm	Introduction to Linux	Dr. Merlin Lopez, Scientist, Community Agrobiodiversity Centre, MS Swaminathan Research Foundation, Kerala
02:30pm	Linux- Hands on exercise	Dr. Merlin Lopez Mr. Fayad M, Research scholar, ICAR-IISR, Kerala.
Day 3: 15-09-2022		
10:00 am	Introduction to Python	Mr. Subeesh A, Scientist (Computer Applications), ICAR- Central Institute of Agricultural Engineering
02:00 pm	Introduction to Galaxy	Dr. Prashanth N Suravajhala, Principal Scientist, School of Biotechnology, Amrita Vishwa Vidyapeetham
Post-workshop evaluation		
Concluding Session		
04:00 pm	Feedback by participants	
04:15 pm	Concluding Remarks	Dr. Prasath D, HRD Nodal Officer, ICAR-Indian Institute of Spices Research
04:20 pm	Vote of Thanks	Ms. Sona Charles

LIST OF RESOURCE PERSONS INVOLVED IN ONLINE TRAINING

Sl. No.	Name	Designation	Affiliation	email
External Resource Persons				
1	Dr. Santhosh J Eapen	Former Director	ICAR-Indian Institute of Spices Research, Kerala	santhosh.eapen@icar.gov.in
2	Dr. Merlin Lopez	Scientist	Community Agrobiodiversity Centre, MS Swaminathan Research Foundation, Wayanad, Kerala	merlinlettinzha@gmail.com
3	Mr. Subeesh A	Scientist	Computer Applications, ICAR- Central Institute of Agricultural Engineering, Bhopal, Madhya Pradesh, India	subeesh.a@icar.gov.in
4	Dr. Prashanth N Suravajhala	Principal Scientist	School of Biotechnology, Amrita Vishwa Vidyapeetham, Kollam, Kerala	prash@am.amrita.edu
Internal Resource Persons				
1	Ms. Sona Charles	Scientist	ICAR-Indian Institute of Spices Research, Kerala	sona.charles@icar.gov.in
2	Mr. S Mukesh Sankar	Scientist	ICAR-Indian Institute of Spices Research, Kerala	mukesh.genetics@gmail.com
3	Mr. Jayarajan K	Chief Technical Officer	ICAR-Indian Institute of Spices Research, Kerala	Jayarajan.K@icar.gov.in
4	Mr. Fayad M.A	Research Scholar	ICAR-Indian Institute of Spices Research, Kerala	muhd.fayad1994@gmail.com

Contents

Topic 1:

Introduction to R

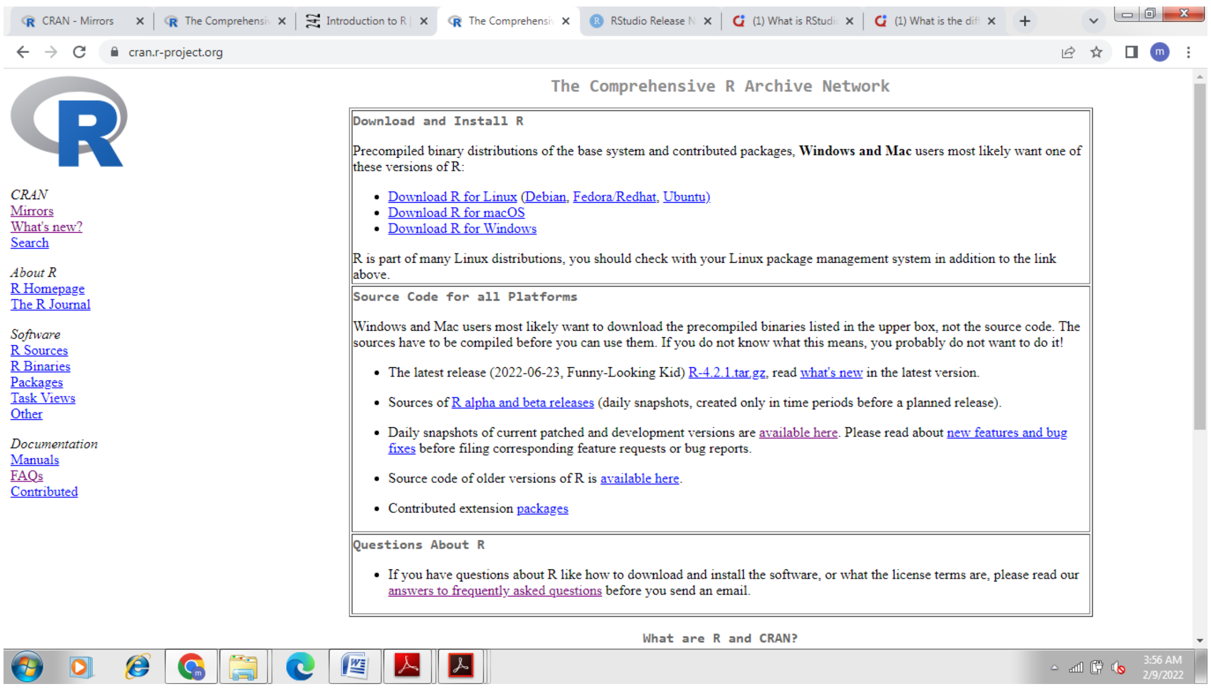
Mr. Mukesh Sankar. S
Scientist (Crop Improvement & Biotechnology),
ICAR-Indian Institute of Spices Research, Kozhikode, Kerala.
Email: mukesh.genetics@gmail.com

General Overview

R is a comprehensive statistical environment and programming language for professional data analysis and graphical display. The R software is free and runs on all common operating systems such as Windows, MacOS and Linux. The key feature of the environment is that it is open source, rapidly evolving, interactivedata analytic platform with large global support system. One of R's strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.

Downloading and Installation of the Software

Precompiled binary distributions of the base system and contributed packages, Windows and Mac users most likely want one of these versions of R: Linux , MacOS X, Windows.



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2022-06-23, Funny-Looking Kid) R-4.2.1.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?

Download and Installation of R for *Windows* is as follows:

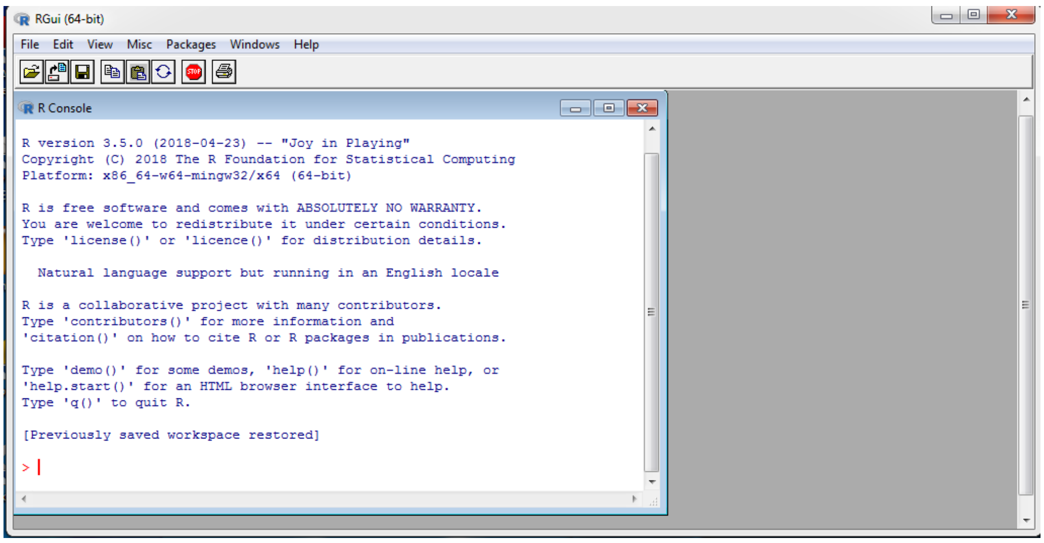
- Visit http://cran.r-project.org/
- Browse Windows
- Click on "base" link - Binaries for base distribution (managed by Duncan Murdoch)
- Click "README on the Windows binary distribution" for Installation and other instructions
- Click "Download R-4.2.1 for Windows (79 megabytes, 64 bit)" for downloading R-4.2.1 software
- Once download is complete, run "R--4.2.1-win32.exe".
- Follow the instructions to install R software.

For Linux:

R can be installed on Ubuntu, using the following Bash script:
sudo apt-get install r-base

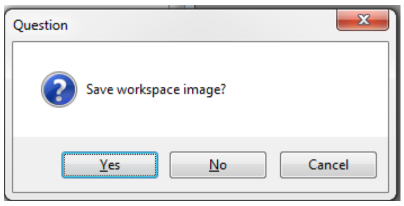
Invoking R

If properly installed, usually R has a shortcut icon on the desktop screen and/or you can find it under Start →All Programs→R menu. Click "R" shortcut icon. A "RGui" based "R Console" will appear.



```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

>
```

To quit R, type q() at the R prompt (>) and press Enter key. A dialog box will ask whether to save the objects you have created during the session so that they will become available next time when R will be invoked.



RStudio

RStudio is an IDE (integrated development environment), that is used to develop R programs more easily and efficiently. It is also available as open source or commercial editions which forms front end editor for R programming. So it means, RStudio in itself is not very useful without R. Now RStudio can also work well with Python.

Installation of RStudio

RStudio requires R 3.0.1+ that means R software should be pre-installed before using RStudio.

RStudio 2022.07.1+554 requires a 64-bit operating system, and works exclusively with the 64 bit version of R. If you are on a 32 bit system or need the 32 bit version of R, you can use an older version of RStudio (https://support.rstudio.com/hc/en-us/articles/206569407-Older-Versions-of-RStudio).

RStudio free desktop version can be downloaded from the following link:

https://www.rstudio.com/products/rstudio/download/#download

Parts of R Studio

The first time RStudio is opened, three windows are seen. A forth window is hidden by default, but can be opened by clicking the File drop-down menu, then New File, and then R Script.

The Script editor pane

The Source Editor can help you open, edit and execute these programs. It is the pane on the top left of your screen.

The R Console Pane

The R Console is where you can type code that executes immediately. This is also known as the command line. It is at the bottom left of your screen. It is the only part of RStudio that is actually R itself.

The R Environment pane

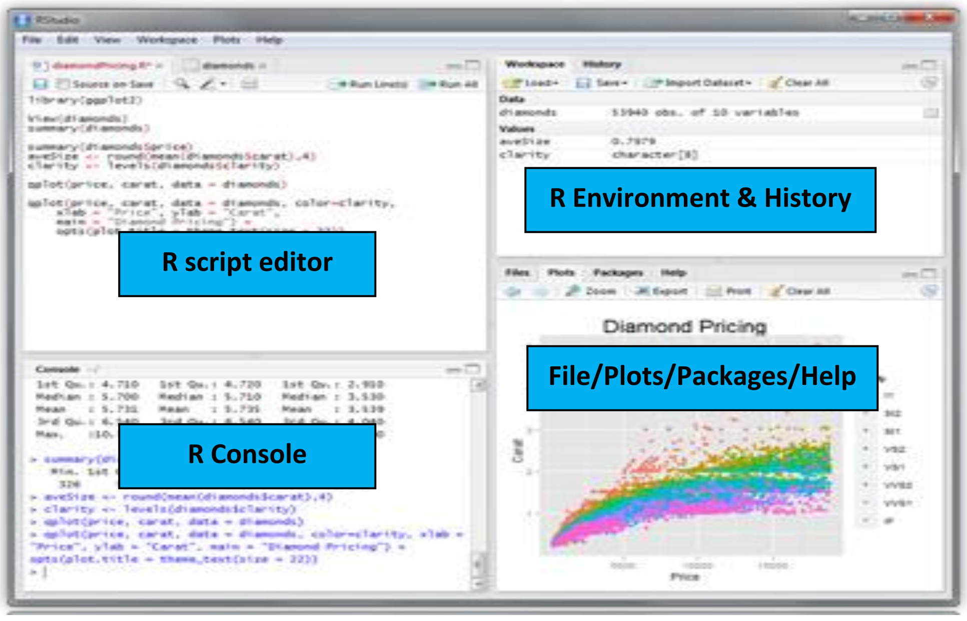
The Environment pane is visible from the top right window as it shows you what objects (i.e., dataframes, arrays, values and functions) you have in your environment (workspace). You can see the values for objects with a single value and for those that are longer, R will tell you their class.

When you have data in your environment that have two dimensions (rows and columns) you may click on them and they will appear in the script editor pane like a spreadsheet. It is at the top right of your screen.

Files/Plots/Packages/Help pane

The last pane appear at bottom right is a basic file browser has a number of different tabs.

- The Files tab has a navigable file manager, just like the file system on your operating system.
- The Plot tab is where graphics you create will appear.
- The Packages tab shows you the packages that are installed and those that can be installed.
- The Help tab allows you to search the R documentation for help and is where the help appears when you ask for it from the Console. It is at the bottom right of your screen.



View of RStudio IDE

```
#################################################################################
###
# R Script for Hands on session : Introduction to R
# Online workshop on "Life science meets programming"
# Created on 06.09.2022 by Mukesh Sankar.S & Sona Charles
# Division of Crop Improvement & Biotectnology, ICAR-IISR, Kozhikode, Kerala, India
############################################################################
```

```
# Install CRAN Package (eg: ggplot2):
install.packages("ggplot2")
install.packages(c("readxl","googlesheets4")) # For multiple packages

# Install Bioconductor packages as follows:
  if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager") # Installs BiocManager if not available yet
BiocManager::version() # Reports Bioconductor version
BiocManager::install("rmelting") # Installs packages specified


#Loading of a specific package
#library("pkg")/require("pkg")
library(BiocManager)
library(rmelting)

#Loading of a set of R package
x<-c("plyr", "psych", "rmelting")

lapply(x, FUN = function(X) {
  do.call("require", list(X))
})

# it a command used in rmelting package
melting(sequence = "CAGTGAGACAGCAATGGTCG", nucleic.acid.conc = 2e-06,
        hybridisation.type = "dnadna", Na.conc = 1)

# To retrieve the manual of Package
browseVignettes(package = 'BiocManager')

#Unloading a specific package (eg:augmentedRCBD)
detach("package:agricolae", unload = TRUE)

#Uninstall a R Package (eg:augmentedRCBD)
remove.packages("augmentedRCBD")

#To avail help
help(mean)
#Or use the command:     ?mean

# Navigating directories

# To know which is our working directory
getwd()
# Give list of all object names that are present in the working directory
dir()
# To change the working directory
setwd("~/R Workspace")
```

```
# Using R as a standard calculator
4 # printing a value
2+3 # adding two value
6-2 # subtraction
2*3 # multiplication
6/2 # Division
2^3 # Power


log(10) # Logarithm
sin(90) #sin() function in R returns the sine of a number in radians.
cos(0)
tan(45)
sqrt(16) # Square-root

max(1,2,4,16,32) # maximum
min(1,2,4,16,32) # minimum
range(1,2,4,16,32) # range
sum(1,2,4,16,32) # sum
prod(1,2,4,16,32) #product
mean(1,2,4,16,32) #arithmetic mean

# Create an object with the assignment operator <- or =
a=1 # equal to assignment
b<-2 # left assignment

# Print command to get output in R console
print(a)

#View function can be used to invoke a spreadsheet-style data viewing.
View(a)

# Lets make R to do some complex expression

a=c(1,2,4,16,32)
#1. Standard deviation of vector a
SD=sqrt(var(a))
#2. Coefficent of Variation in %
CV=(sd(a)/mean(a))*100

# Data Types
#1. Numeric
d <- c(1.5, 2.3, 3.1)
d
class(d)
is.numeric(d) # to check the object whether numeric or not

#2. Character
e <- c ("1.5","2.3","3.1")
e
```

```
class(e)
is.numeric(e)
is.character(e) # to check the object whether character or not

#3. Logical data
f <- 1:10 < 5
f
class(f)

#4. Integer
int <- as.integer(2.2) #Is 2.2 an integer?
int
class(int)


# Data Objects
#1. Scalar (Definition : Scalar object is just a single value like a number or a name.)
a
b="LETTER"

#2. Vector (Definition: ordered collection of numeric, character, complex and logical values)
d
e
f

#3. Factor (Definition: vectors with grouping information)
g= factor(c("dog", "cat", "mouse", "dog", "dog", "cat"))
g
class(g)
levels(g)
nlevels(g)
class(levels(g))


#4. Matrices (Definition: two dimensional structures with data of same type)
#Matrix <- matrix(vector, nrow=r, ncol=c, byrow=TRUE/FALSE,
dimnames=list(char_vector_rownames, char_vector_colnames))

Matrix <- matrix(1:30, nrow=3, ncol=10, byrow = TRUE)
class(Matrix)
print(Matrix)

mat1 <- matrix(1:4, nrow = 2, ncol = 2)
mat1[1,2]
mat1[2, ] #extract 2nd row
mat1[,2 ] #extract 2nd column

mat2 <- matrix(13:16, nrow = 2, ncol = 2)
mat2
```

```
mat1+mat2 #adding two matrices
mat1 - mat2 #subtraction of two matrices
4 * mat1 #multiplication by a constant
(mat1/mat2) #division


M3 = matrix( c('AI','ML','DL','Tensorflow','Pytorch','Keras'), nrow = 2, ncol = 3, byrow = FALSE)#
fill the matrix by column
print(M3)


t(M3) #transpose a matrix



#5. Data frame (Definition: Data frames are two dimensional objects with data of variable types)
Data_frame <- data.frame(Col1=1:10, Col2=10:1)
View(Data_frame)
class(Data_frame)
str(Data_frame)

#6. List (Definition: containers for any object type)
List <- list(name="Fred", wife="Mary", no.children=3, child.ages=c(4,7,9))
List
View(List)

#7. Arrays (Definition: data structure with one, two or more dimensions)
#my_array <- array(data, dim = (rows, colums, matrices, dimnames)

v1=c(1,2,3)
v2=c(4,5,6,7,8,9)
col.names=c("Item", "Serial","Size")
row.names=c("Server","Network","Firewall")
matrix.names=c("Datacentre IN", "Datacentre US")
Array = array(c(v1,v2),dim=c(3,3,2),dimnames = list(row.names,col.names,matrix.names))
Array

#6. Functions (Definition: piece of code)

x<-c("plyr", "psych", "rmelting")
lapply(x, FUN = function(X) {
  do.call("require", list(X))
})

# List out the object saved in workspace
ls()
# To remove the object at workspace
rm(Array)

# Subsetting Data objects
# (1.) Subsetting by positive or negative index/position numbers
myVec <- 1:26; names(myVec) <- LETTERS
View(myVec)
myVec[1:4] #Subsetting by positive index number
myVec[-(5:26)] #Subsetting by negative index number

#(2.) Subsetting by same length logical vectors
myLog <- myVec > 10
myVec[myLog]

#(3.) Subsetting by field names
myVec[c("B", "K", "M")]

#(4.) (4.) Subset with $ sign: references a single column or list component by its name
data("iris")
iris$Species[1:8]


# Reading and Writing External Data

#Import of a Dataset in comma delimited format
iris=read.csv(file="iris.csv",header=TRUE)

# Import of a tab-delimited or comma delimited tabular file
iris <- read.delim("iris.txt", sep="/t", header = T)
iris <- read.delim("iris.csv", sep=",", header = T)

# Import of dataset stored in excel
library(readxl)
iris <- read_excel("iris.xlsx", sheet=iris, header=T)

#Dataset from googlesheet
library(googlesheets4)
gs4_deauth() # Easiest method for reading public access sheets
iris <- read_sheet("https://docs.google.com/spreadsheets/d/12MobcUGmY3uf-
SpJtR8chjdv0PSif8znv0ffmjB95ko/edit?usp=sharing")
myDF <- as.data.frame(iris)
myDF

#Dataset from copied in clipboard
clipboard=read.delim("clipboard")


#writing the output in csv/tab delimited format
write.csv(iris,file="iris.csv")

#Playing with datasets

data(package = "datasets")
data(iris)
covid <-read.delim("covid.txt", header = TRUE)
covid <-read.delim("C:/Users/user/Desktop/Schedule/covid.txt", header = TRUE)
head(covid)
tail(covid)
covid <-read.delim("C:/Users/user/Desktop/Schedule/covid.txt", header = FALSE)
head(covid)

#dataframe indexing
covid[2,3]      #value in second row, third column
covid[,1]       #first column, as a vector
covid[2,]       #second row, as a data.frame
covid[,2:3]     #second and third columns, as a data.frame
covid[1]        #first column, as a data.frame
covid[1:5, c(3,5)]      #rows 1-5, columns 3 and 5
covid[,-1]      #everything but the first column
covid[nrow(covid):1,]#everything, with rows in reverse order
covid[covid[,2] < 10000,]       #rows of covid (with all columns) where the value in the first column
is less than 10000
covid$State.UTs         #State.UTs column, as a vector
covid[,"State.UTs"]     #State.UTs column, as a vector
covid[,c("State.UTs", "Active")]        #State.UTs and Active columns, as a data.frame
covid["10",]    #row named "10", as a data.frame
covid["State.UTs"]      #State.UTs column, as a data.frame
covid[order(covid$Active), c("State.UTs", "Total.Cases", "Deaths", "Active")] #ordering according
to active cases and displaying only 4 columns
nrow(covid)
ncol(covid)
dim(covid)
str(covid)
plot(covid)
summary(covid)
summary(covid$Total.Cases)
min(covid$Total.Cases)
max(covid$Total.Cases)
sd(covid$Total.Cases)
var(covid$Total.Cases)
prod(covid$Total.Cases)
sum(covid$Active)
```

Data Wrangling using Dplyr package

Data analysis can be divided into three parts:

- Extraction: First, we need to collect the data from many sources and combine them.
- Transform: This step involves the data manipulation. Once we have consolidated all the sources of data, we can begin to clean the data.
- Visualize: The last move is to visualize our data to check irregularity.

One of the most significant challenges faced by data scientists is the data manipulation. Data is never available in the desired format. Data scientists need to spend at least half of their time, cleaning and manipulating the data. That is one of the most critical assignments in the job. If the data manipulation process is not complete, precise and rigorous, the model will not perform correctly.

```
#Data wrangling with Dplyr

#install.packages("dplyr")
library(dplyr)

#Selecting columns
select_data <-select(covid, State.UTs, Total.Cases, Deaths)
head(select_data)

head(select(covid, -Discharged)) #To select all the columns except a specific column
head(select(covid, State.UTs:Deaths)) #To select a range of columns
head(select(covid, starts_with("D")))
head(select(covid, ends_with("s")))
head(select(covid, contains("Ratio")))
head(select(covid, contains("hs")))

#Filtering rows
filter(covid, Deaths >= 16000)
filter(covid, Active >= 10000, Deaths >= 10000)

#Pipe operator: %>%
covid %>%
  select(State.UTs, Total.Cases, Deaths) %>%
  head

covid %>% arrange(Active) %>% head


covid %>%
  select(State.UTs, Total.Cases, Deaths) %>%
  arrange(Deaths, Total.Cases) %>%
  head

covid %>%
  select(State.UTs, Total.Cases, Deaths) %>%
  arrange(Total.Cases, Deaths) %>%
  filter(Deaths <= 250)

covid %>%
  mutate(Ratio = Active / Total.Cases) %>%
  head
glimpse(covid)

#summarizing your data
```

```
summarise(covid, mean = mean(Deaths))
summarise(covid, min = min(Deaths))
summarise(covid, max = max(Deaths))
summarise(covid, med = median(Deaths))

#random sampling
# Printing three rows
sample_n(covid, 3) #3 random samples
sample_n(covid, 3) #sample again

# Printing 50 % of the rows
sample_frac(covid, 0.10)
```

Reference

- R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org.
- Zuur, A.F., Ieno, E.N. and Meesters, E.H. (2009). A Beginner's Guide to R (p. 150). New York: Springer.
- http://manuals.bioinformatics.ucr.edu/home/R_BioCondManual#TOC-Introduction
- https://girke.bioinformatics.ucr.edu/GEN242/tutorials/rbasics/rbasics/

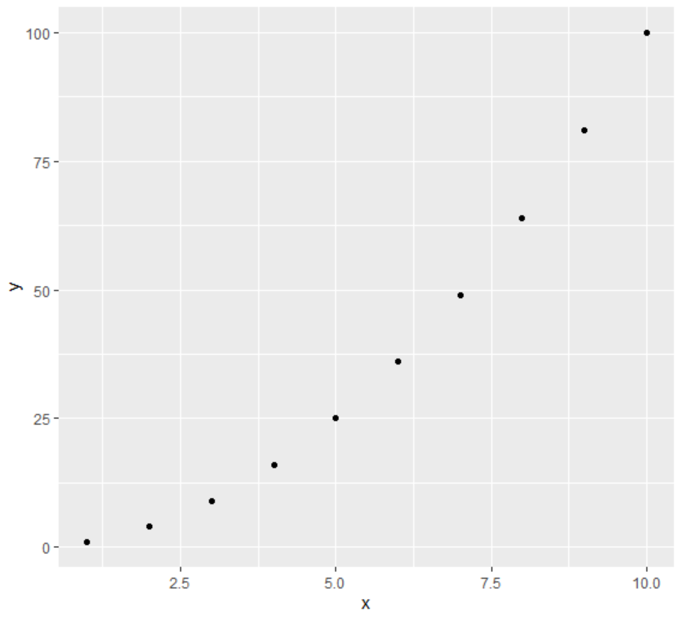
Topic 2: Data Visualization using R

Ms. Sona Charles
Scientist (Bioinformatics),
ICAR-Indian Institute of Spices Research, Kozhikode, Kerala.
Email: sona.charles@icar.gov.in

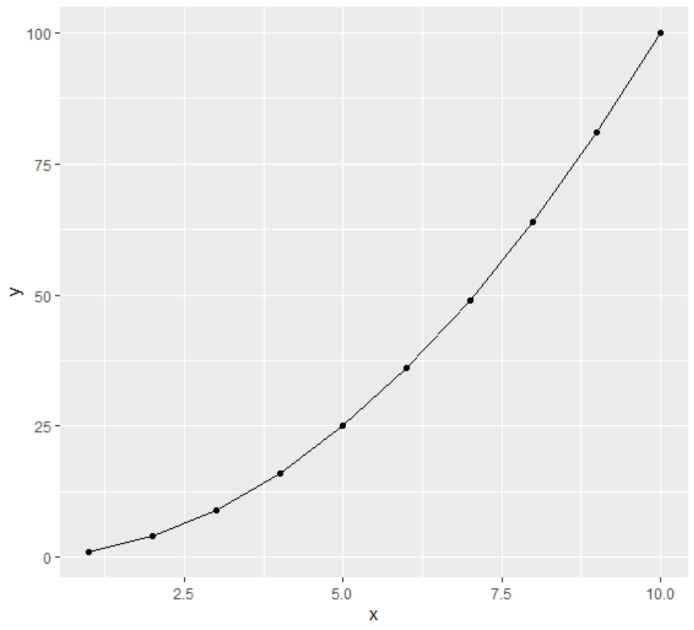
```
>install.packages("ggplot2")
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.0/ggplot2_3.3.5.zip'
Content type 'application/zip' length 4129871 bytes (3.9 MB)
downloaded 3.9 MB

package 'ggplot2' successfully unpacked and MD5 sums checked

The downloaded binary packages are in <PATH>
> library(ggplot2)
Warning message:
package 'ggplot2' was built under R version 4.0.5
```

Your First quick ggplot!

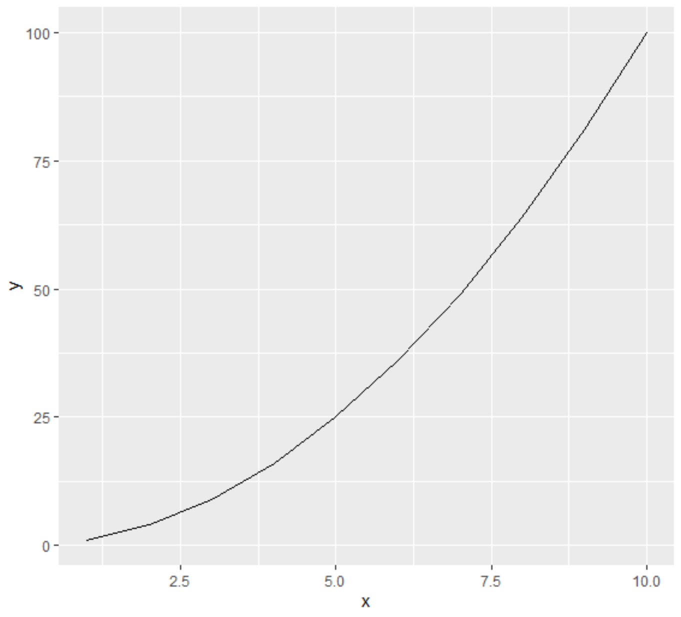
```
> x <- 1:10
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> y = x*x
> y
 [1]   1   4   9  16  25  36  49  64  81 100
>qplot(x,y)
```



```
>qplot(x, y, geom=c("line", "point"))
```



```
>qplot(x, y, geom=c("line"))
```



Scatterplots

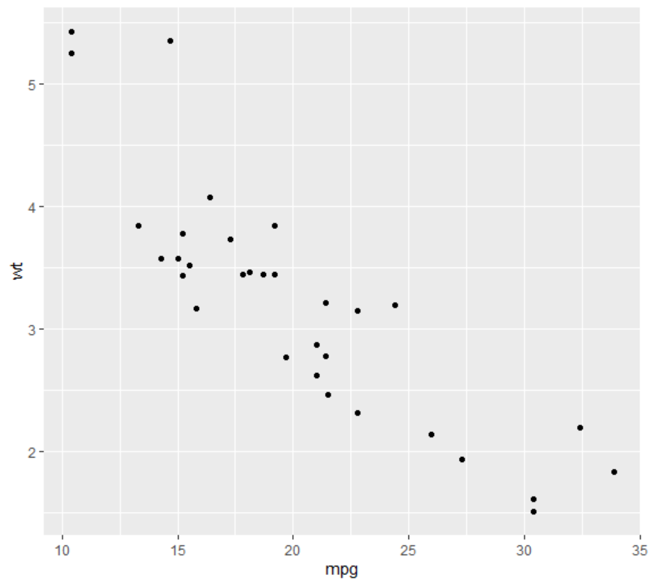
Dataset: mtcars **(**Motor Trend Car Road Tests)

Description: The data comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973 - 74 models).

Format: A data frame with 32 observations on 3 variables.

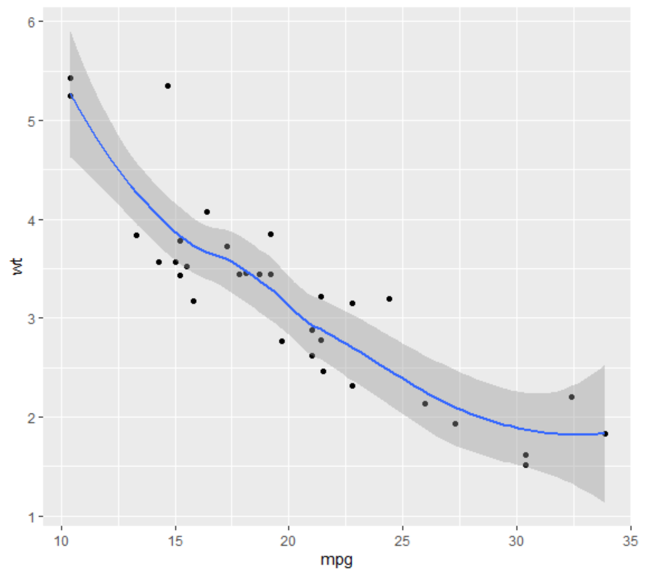
```
> data(mtcars)
> head(mtcars)
                  mpg cyldisphp drat    wtqsecvs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
>df<- mtcars[, c("mpg", "cyl", "wt")]
> head(df)
                  mpg cylwt
Mazda RX4         21.0   6 2.620
Mazda RX4 Wag     21.0   6 2.875
```

```
Datsun 710        22.8   4 2.320
Hornet 4 Drive    21.4   6 3.215
Hornet Sportabout 18.7   8 3.440
Valiant           18.1   6 3.460
>qplot(mpg, wt, data=mtcars)
```



The option "smooth" is used to add a smoothed line with its standard error.

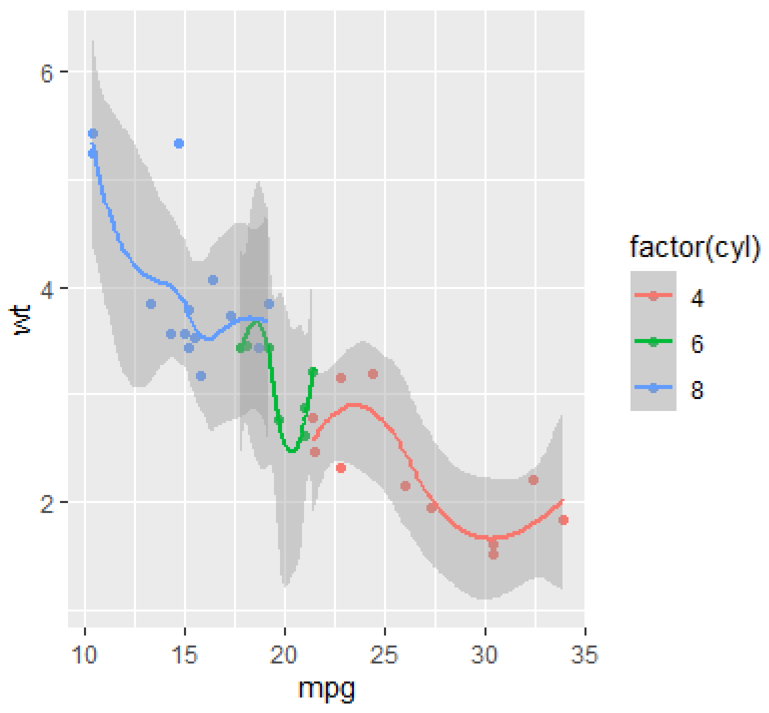
```
#Scatter plots with smoothed line
>qplot(mpg, wt, data = mtcars, geom = c("point", "smooth"))
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



*LOESS is a popular tool used in regression analysis that creates a smooth line through a timeplot or scatter plot to help you to see relationship between variables and foresee trends.

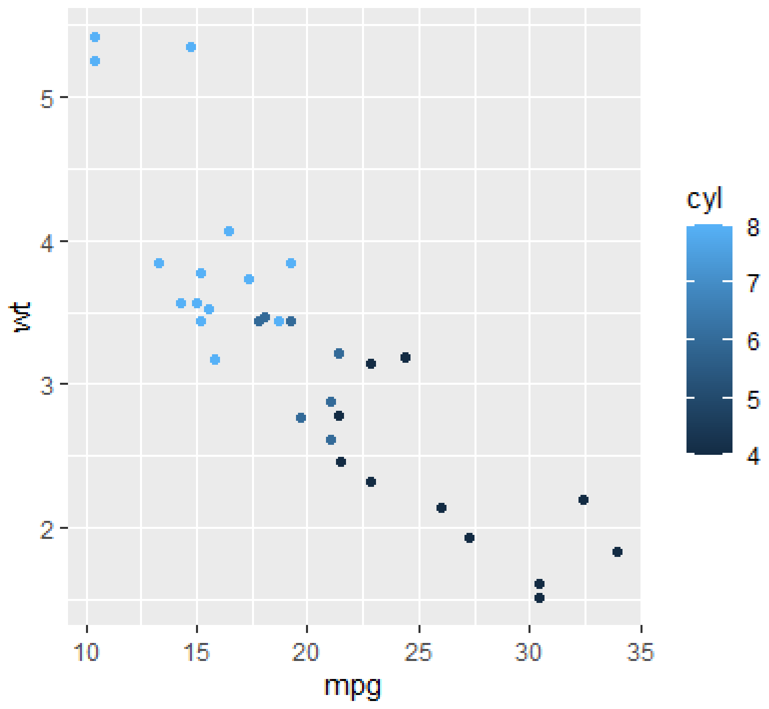
The argument “color” is used to tell R that we want to color the points by groups:

```
>qplot(mpg, wt, data = mtcars, color = factor(cyl),
+      geom=c("point", "smooth"))
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

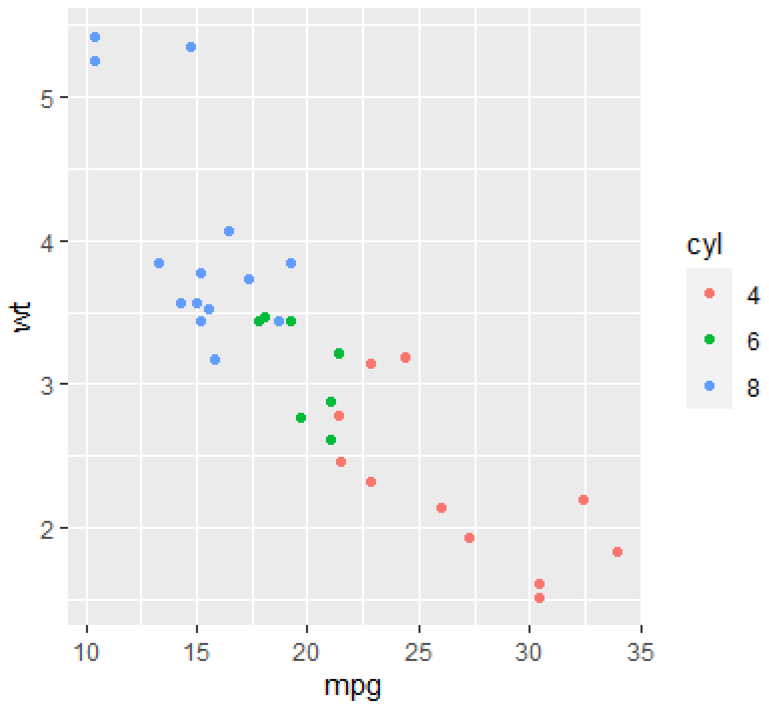


Points can be colored according to the values of a continuous or a discrete variable. The argument “colour” is used.

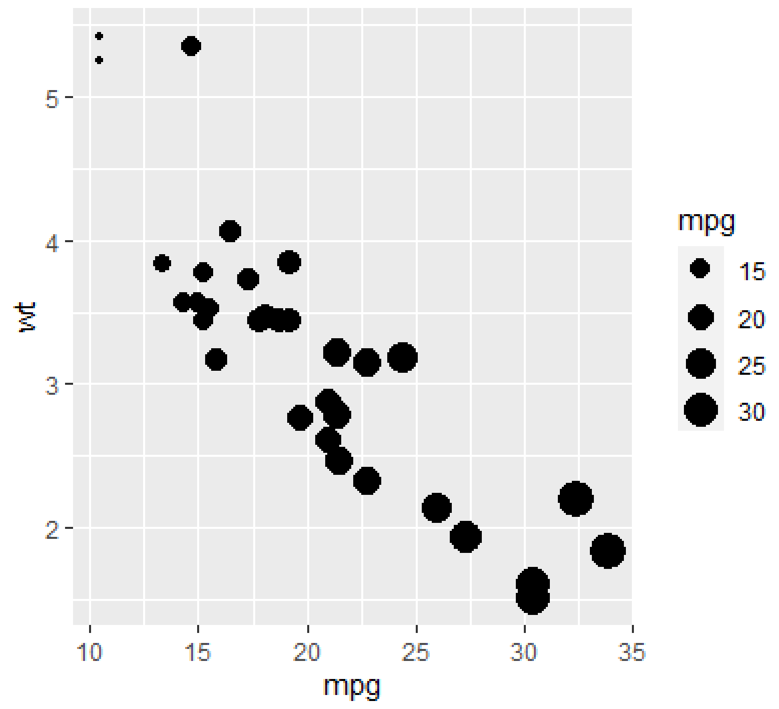
```
>qplot(mpg, wt, data = mtcars, colour = cyl)
```



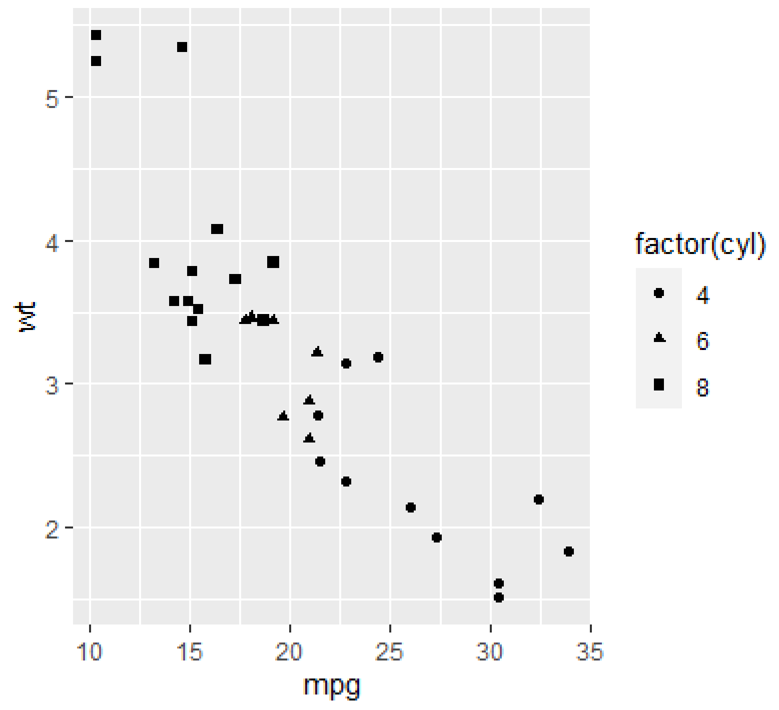
```
> # Change the color by groups (factor)
>df<- mtcars
>df[,'cyl'] <- as.factor(df[,'cyl']) #convert the cyl column to a factor
>qplot(mpg, wt, data = df, colour = cyl)
```



```
# Change the size of points according to  the values of a continuous variable
>qplot(mpg, wt, data = mtcars, size = mpg)
```

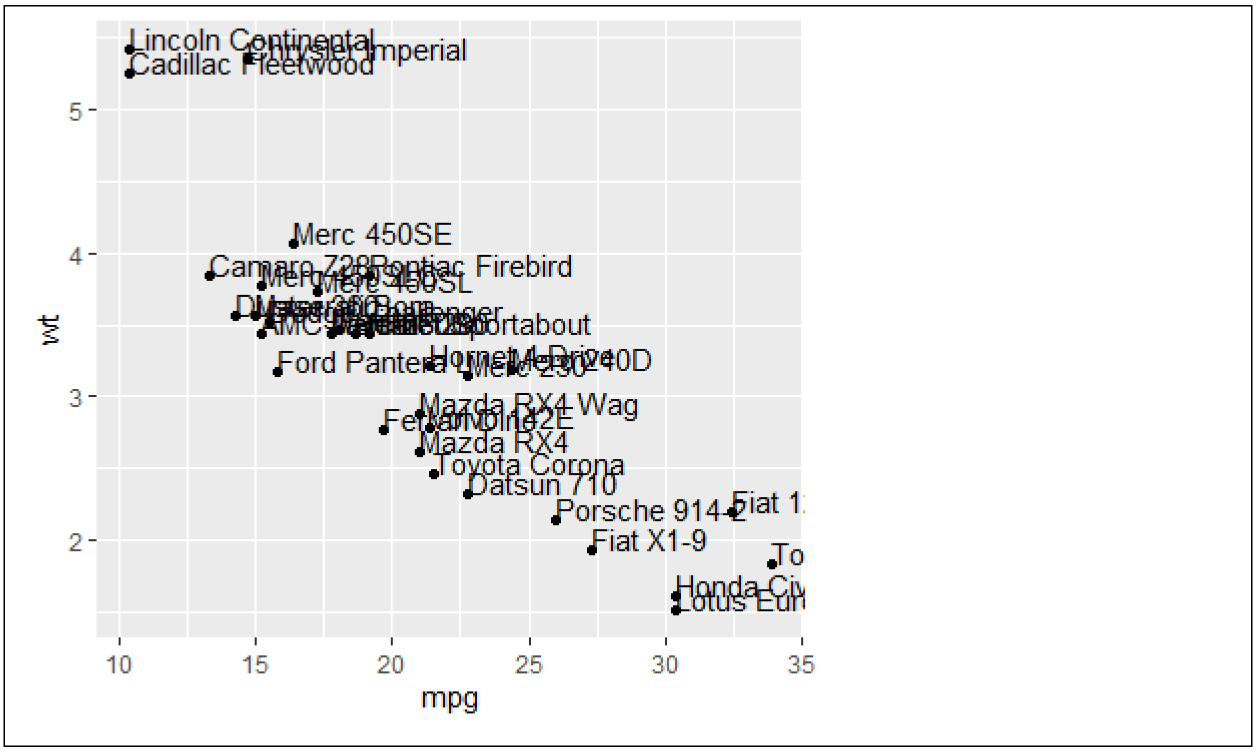


```
> # Change point shapes by groups
>qplot(mpg, wt, data = mtcars, shape = factor(cyl))
```



Scatter plot with texts

```
>qplot(mpg, wt, data = mtcars, label = rownames(mtcars),
+      geom=c("point", "text"),
+      hjust=0, vjust=0)
```



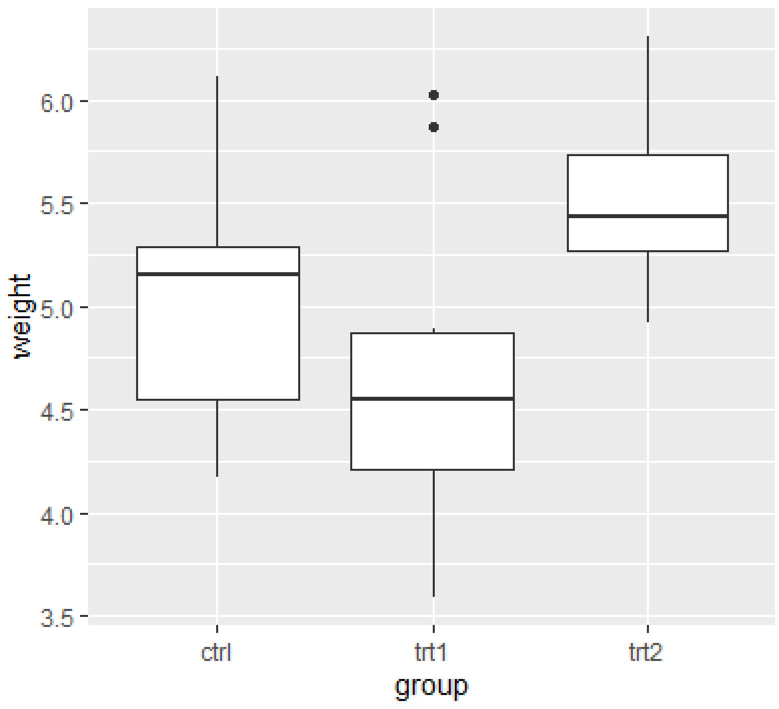
Box Plot

Dataset: PlantGrowth

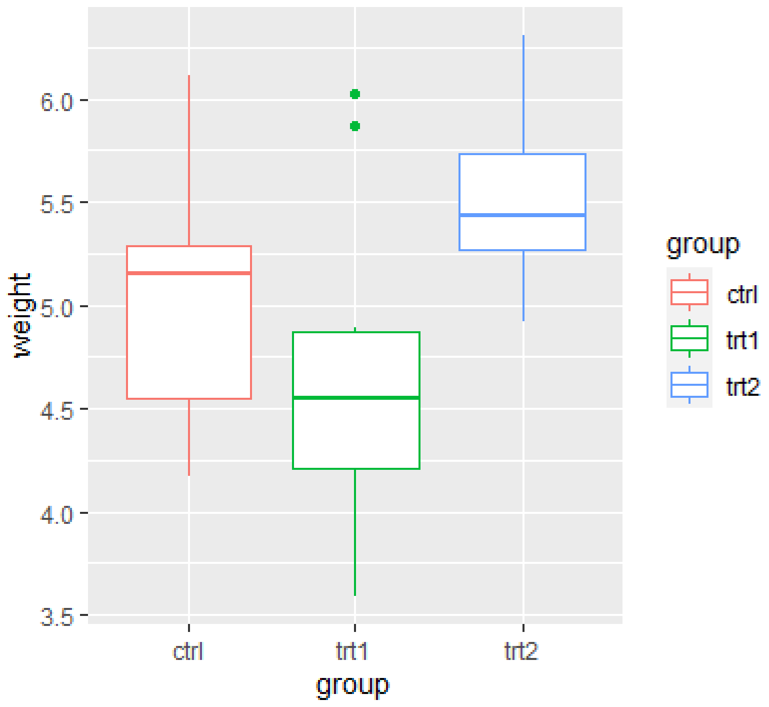
Description: Results from an experiment to compare yields (as measured by dried weight of plants) obtained under a control and two different treatment conditions.

Format:A data frame of 30 cases on 2 variables.

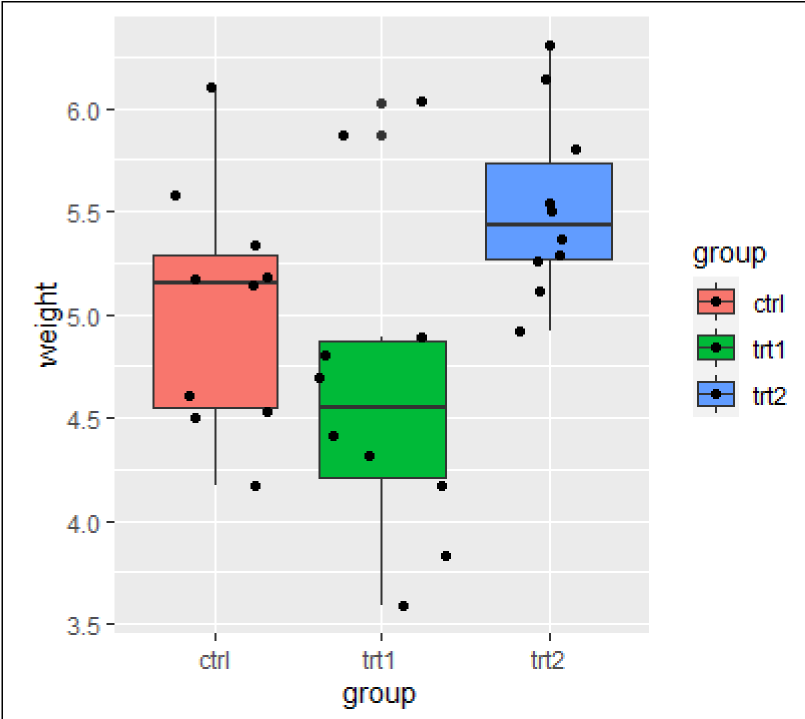
```
> data("PlantGrowth")
> head(PlantGrowth)
 weight group
1  4.17  ctrl
2  5.58  ctrl
3  5.18  ctrl
4  6.11  ctrl
5  4.50  ctrl
6  4.61  ctrl
>qplot(group, weight, data = PlantGrowth,
+      geom=c("boxplot"))
```



```
>qplot(group, weight, data = PlantGrowth, color = group,
+      geom=c("boxplot")
```

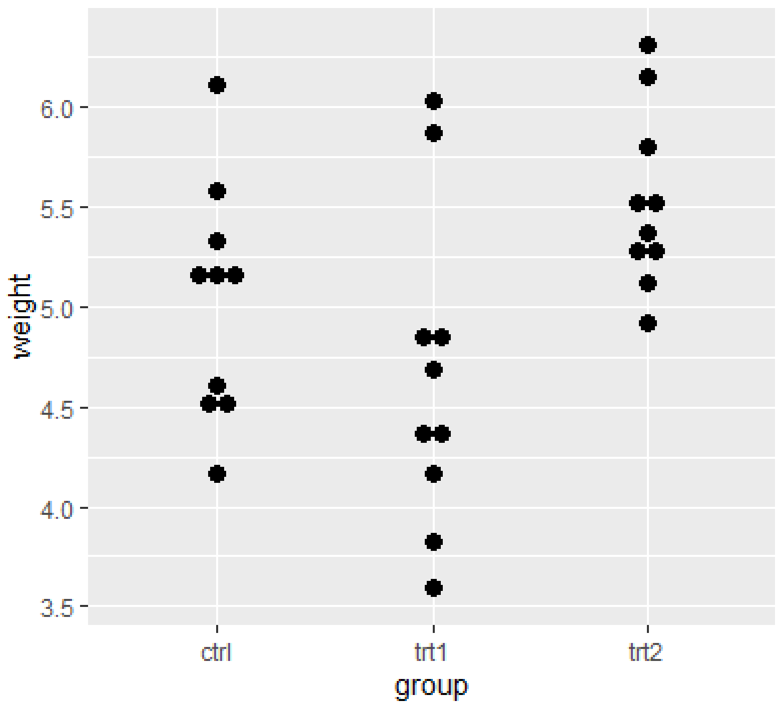


```
>qplot(group, weight, data = PlantGrowth,
+      geom=c("boxplot", "jitter"), fill = group)
```

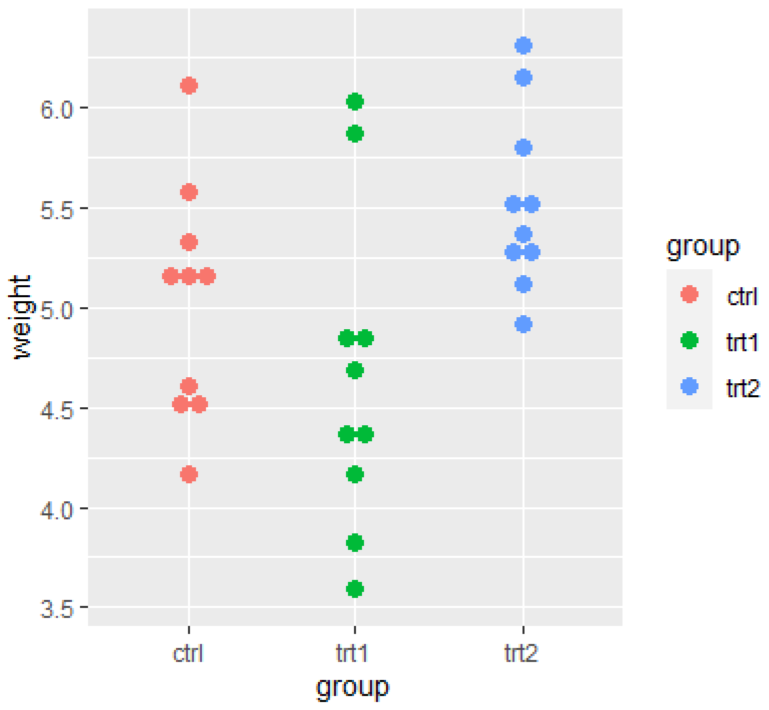


Dot Plot

```
>qplot(group, weight, data = PlantGrowth,
geom=c("dotplot"),
stackdir = "center", binaxis = "y")
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```

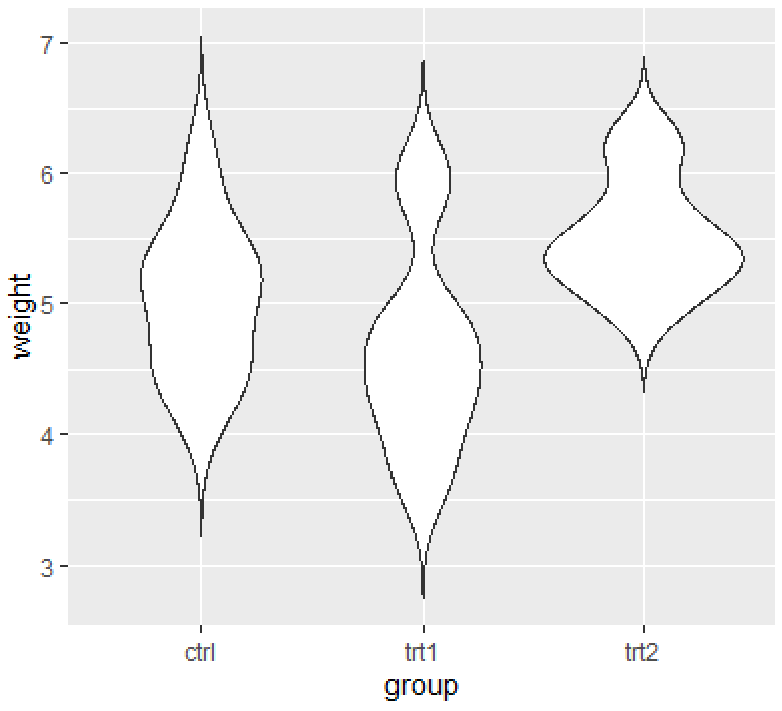


```
>qplot(group, weight, data = PlantGrowth,
+       geom = "dotplot", stackdir = "center", binaxis = "y",
+       color = group, fill = group)
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.
```



Violin Plot

```
>qplot(group, weight, data = PlantGrowth,
+      geom=c("violin"), trim = FALSE)
```



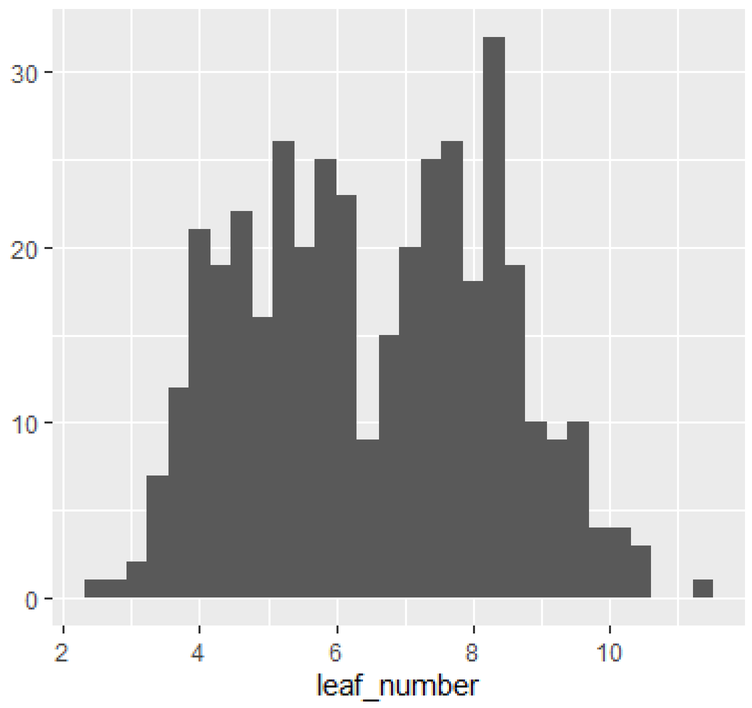
Histogram

Dataset: We will generate some data.

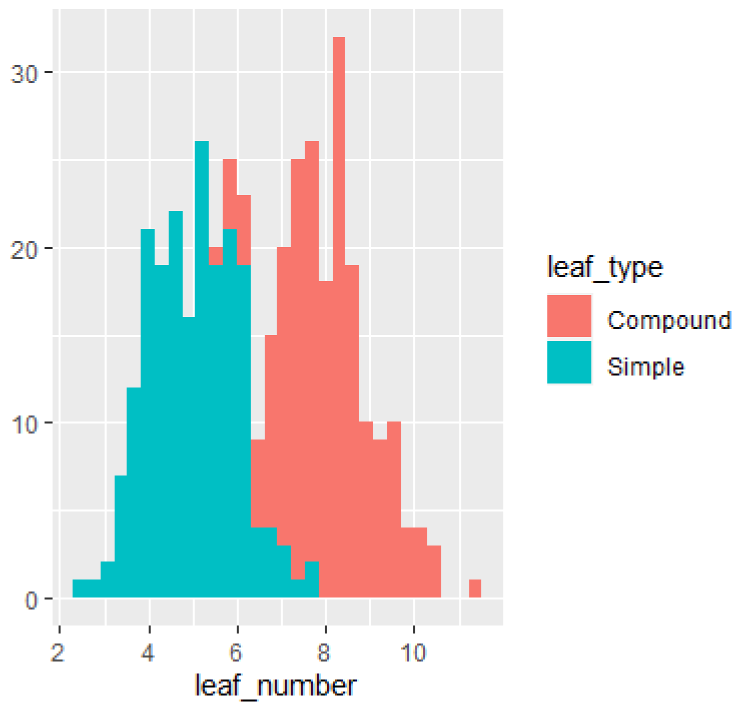
The set.seed() function sets the starting number used to generate a sequence of random numbers

```
>set.seed(3)
> created = data.frame(
+   leaf_type = factor(rep(c("Simple", "Compound"), each=200)),
+   leaf_number = c(rnorm(200, 5), rnorm(200, 8)))
> head(created)
leaf_typeleaf_number
1   Simple    4.038067
2   Simple    4.707474
3   Simple    5.258788
4   Simple    3.847868
5   Simple    5.195783
6   Simple    5.030124
```

```
>qplot(leaf_number, data = created, geom = "histogram")
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
> # Change histogram fill color by group (leaf_type)
>qplot(leaf_number, data = created, geom = "histogram",
+       fill = leaf_type)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

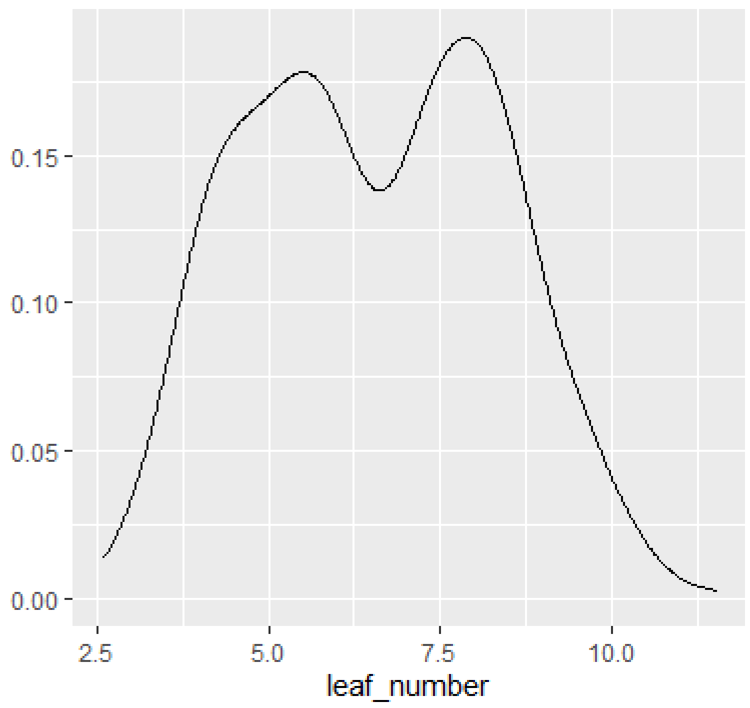


Density Plot

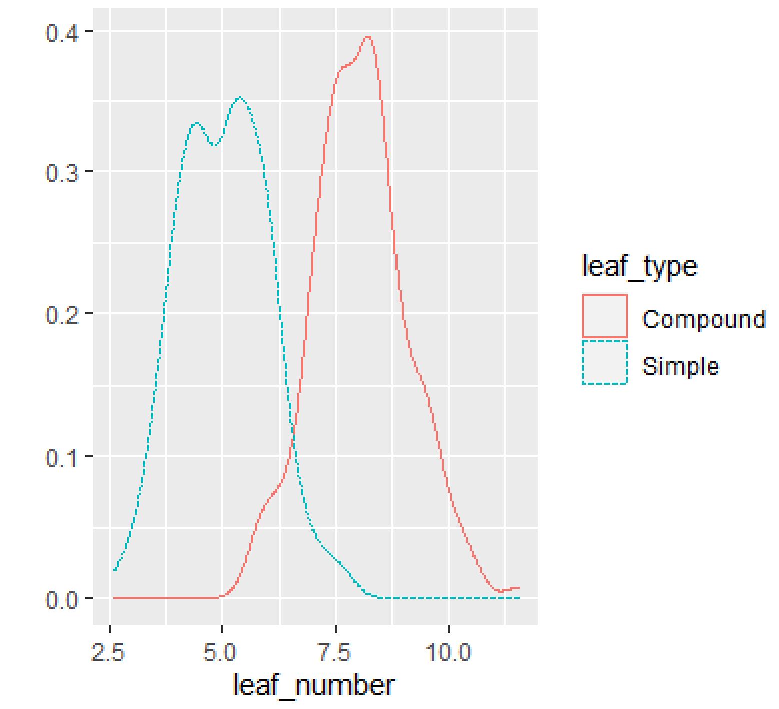
Dataset: Data generated for histogram.

A density plot is a representation of the distribution of a numeric variable. It is a smoothed version of the histogram and is used in the same concept.

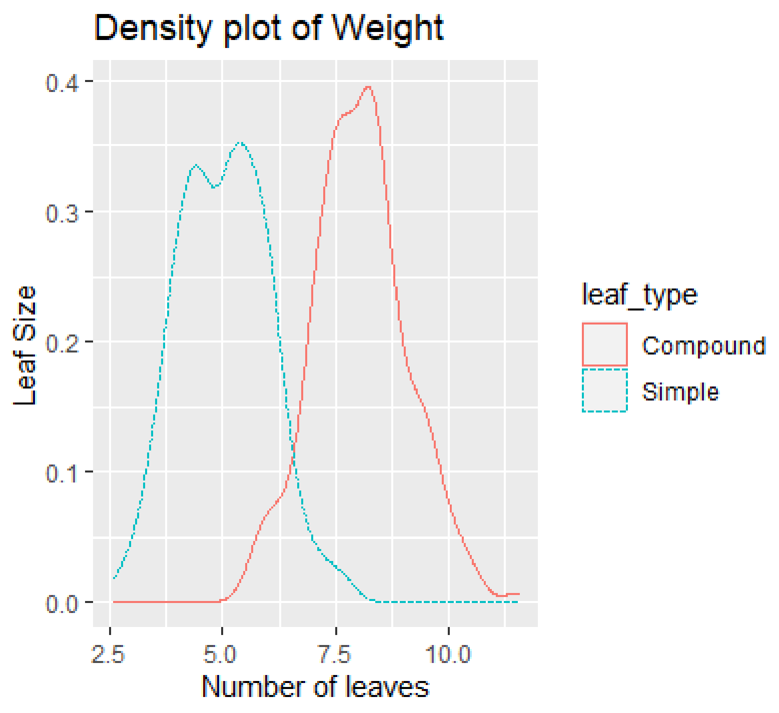
```
>qplot(leaf_number, data = created, geom = "density")
```



```
>qplot(leaf_number, data = created, geom = "density",
+      color = leaf_type, linetype = leaf_type)
```



```
>qplot(leaf_number, data = created, geom = "density",
+      color = leaf_type, linetype = leaf_type,
+      xlab = "Number of leaves", ylab = "Leaf Size",
+      main = "Density plot of Weight")
```



Strip Charts/ Jitter Plot

Dataset: ToothGrowth

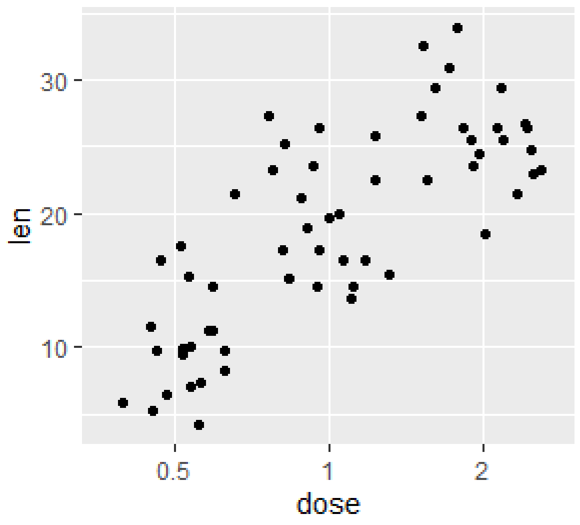
Description: Length of the teeth in each of 10 guinea pigs at three Vitamin C dosage levels (0.5, 1, and 2 mg) with two delivery methods (orange juice or ascorbic acid).

Format: The file contains 60 observations of 3 variables

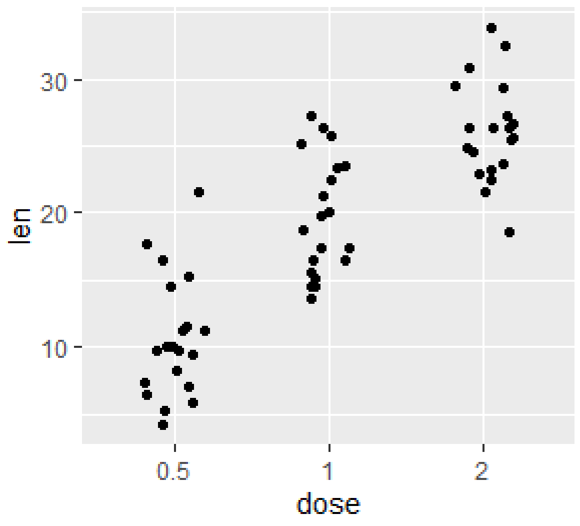
```
#STRIP CHART/ JITTER PLOT
>ToothGrowth
lensupp dose
1   4.2  VC  0.5
2  11.5  VC  0.5
3   7.3  VC  0.5
4   5.8  VC  0.5
5   6.4  VC  0.5
6  10.0  VC  0.5
7  11.2  VC  0.5
8  11.2  VC  0.5
9   5.2  VC  0.5
10  7.0  VC  0.5
11 16.5  VC   1
12 16.5  VC   1
13 15.2  VC   1
14 17.3  VC   1
15 22.5  VC   1
16 17.3  VC   1
17 13.6  VC   1
18 14.5  VC   1
19 18.8  VC   1
20 15.5  VC   1
21 23.6  VC   2
22 18.5  VC   2
23 33.9  VC   2
24 25.5  VC   2
```

25	26.4	VC	2
26	32.5	VC	2
27	26.7	VC	2
28	21.5	VC	2
29	23.3	VC	2
30	29.5	VC	2
31	15.2	OJ	0.5
32	21.5	OJ	0.5
33	17.6	OJ	0.5
34	9.7	OJ	0.5
35	14.5	OJ	0.5
36	10.0	OJ	0.5
37	8.2	OJ	0.5
38	9.4	OJ	0.5
39	16.5	OJ	0.5
40	9.7	OJ	0.5
41	19.7	OJ	1
42	23.3	OJ	1
43	23.6	OJ	1
44	26.4	OJ	1
45	20.0	OJ	1
46	25.2	OJ	1
47	25.8	OJ	1
48	21.2	OJ	1
49	14.5	OJ	1
50	27.3	OJ	1
51	25.5	OJ	2
52	26.4	OJ	2
53	22.4	OJ	2
54	24.5	OJ	2
55	24.8	OJ	2
56	30.9	OJ	2
57	26.4	OJ	2

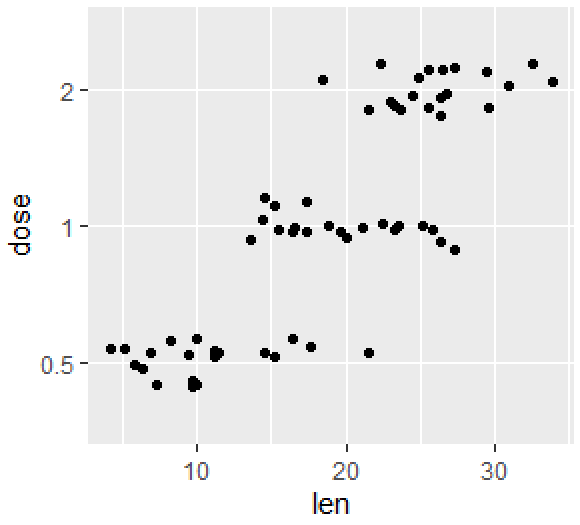
```
58 27.3   OJ    2
59 29.4   OJ    2
60 23.0   OJ    2
>str(ToothGrowth)
'data.frame':  60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: Factor w/ 3 levels "0.5","1","2": 1 1 1 1 1 1 1 1 1 1 ...
>ToothGrowth$dose<- as.factor(ToothGrowth$dose)
>str(ToothGrowth)
'data.frame':  60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: Factor w/ 3 levels "0.5","1","2": 1 1 1 1 1 1 1 1 1 1 ...
>ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter()
```



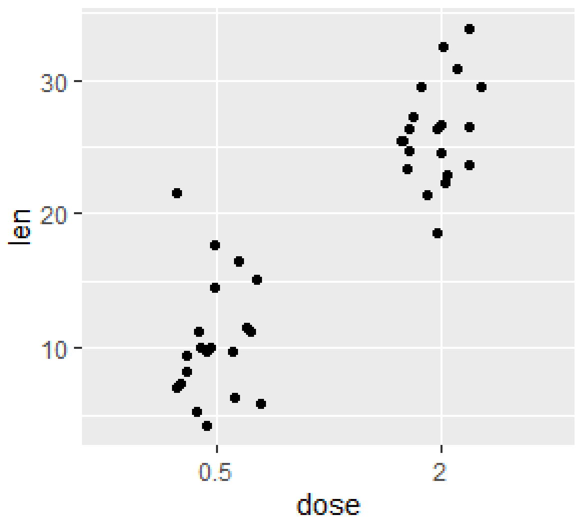
```
> p<-ggplot(ToothGrowth, aes(x=dose, y=len)) +
geom_jitter(position=position_jitter(0.2))
> p
```



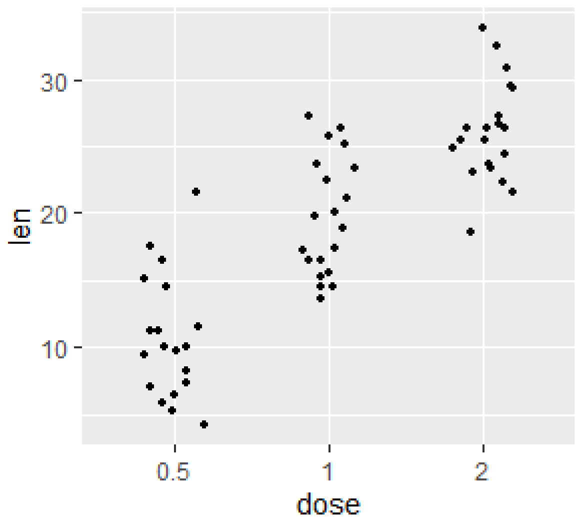
```
> p + coord_flip()
> p
```



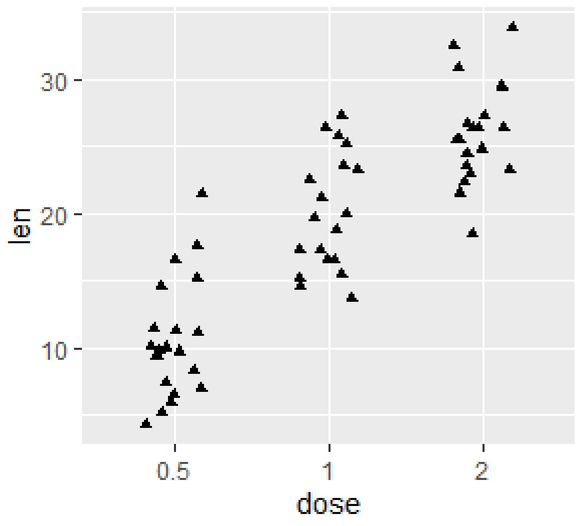
```
> p + scale_x_discrete(limits=c("0.5", "2"))
Warning message:
Removed 20 rows containing missing values (geom_point).
```



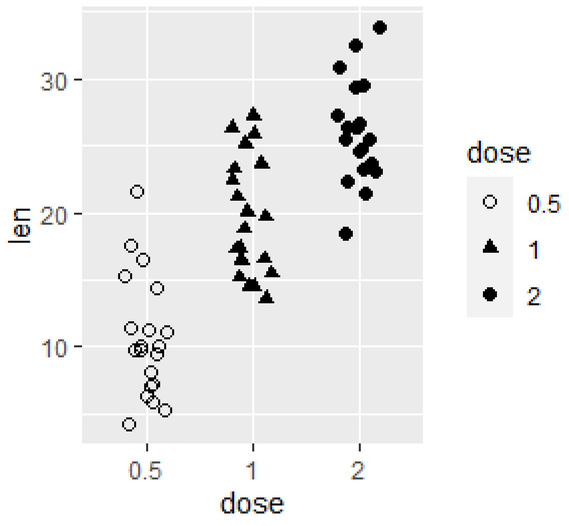
```
> #change the size of points
>ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter(position=position_jitter(0.2), cex=1.2)
```



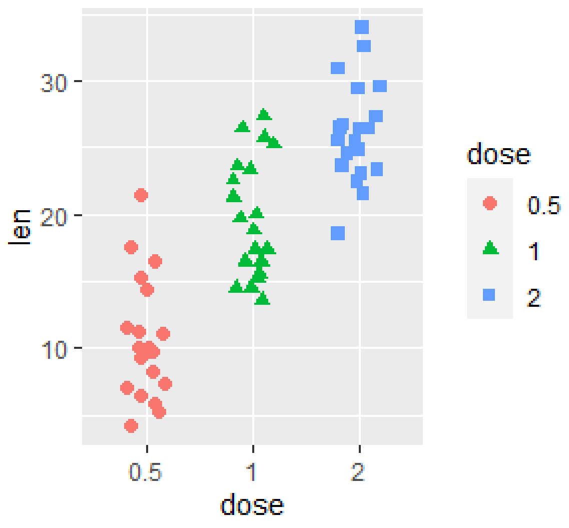
```
> #change shape of points
>ggplot(ToothGrowth, aes(x=dose, y=len)) +
+   geom_jitter(position=position_jitter(0.2), shape=17)
```



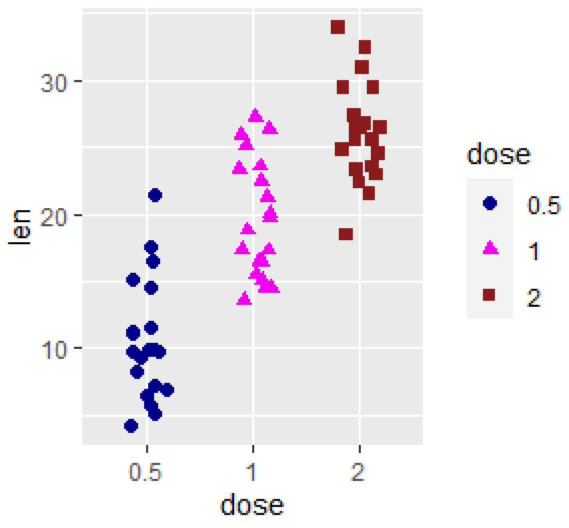
```
> #Change shape according to dose
> p <- ggplot(ToothGrowth, aes(x=dose, y=len, shape=dose)) +
+   geom_jitter(position=position_jitter(0.2), cex=2)
> p + scale_shape_manual(values=c(1,17,19))
```



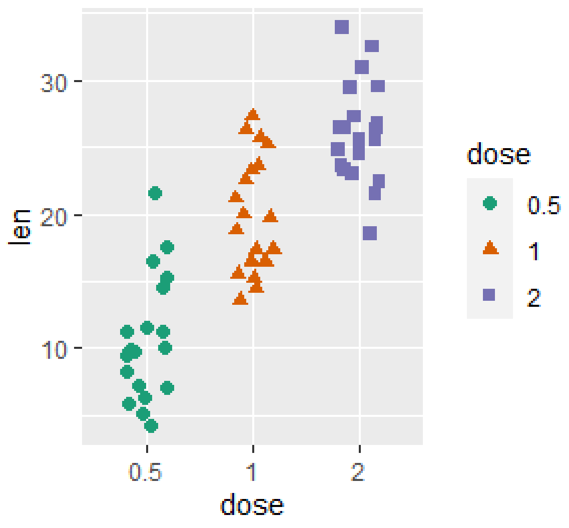
```
> # Change shape and color according to dose
> p<-ggplot(ToothGrowth, aes(x=dose, y=len, shape=dose, color=dose)) +
+   geom_jitter(position=position_jitter(0.2), cex=2)
> p
```



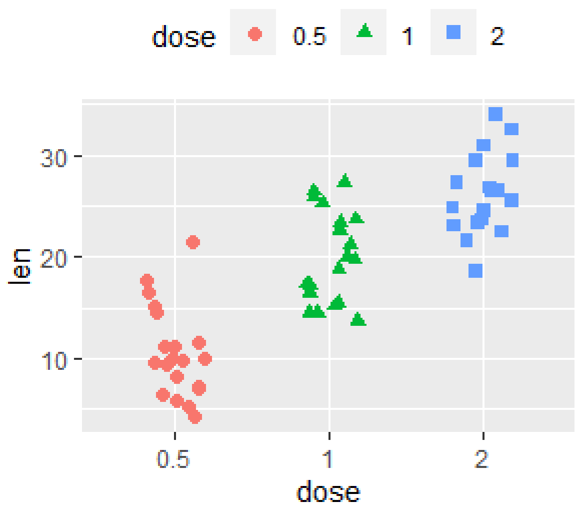
```
> #choose your colors
>p+scale_color_manual(values=c("blue4", "magenta2", "firebrick4"))
```



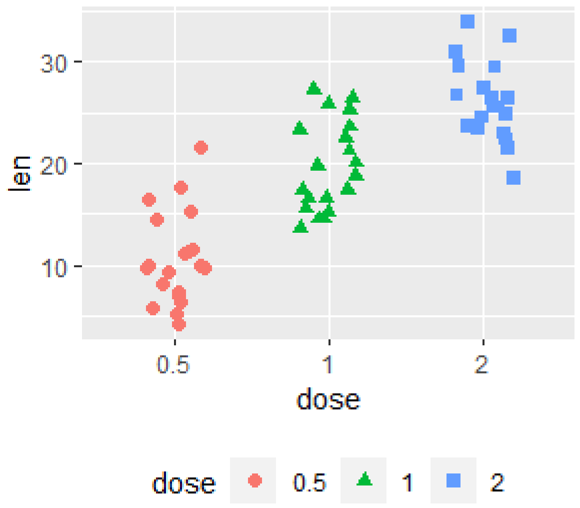
```
> # Choose your palette
>p+scale_color_brewer(palette="Dark2")
```



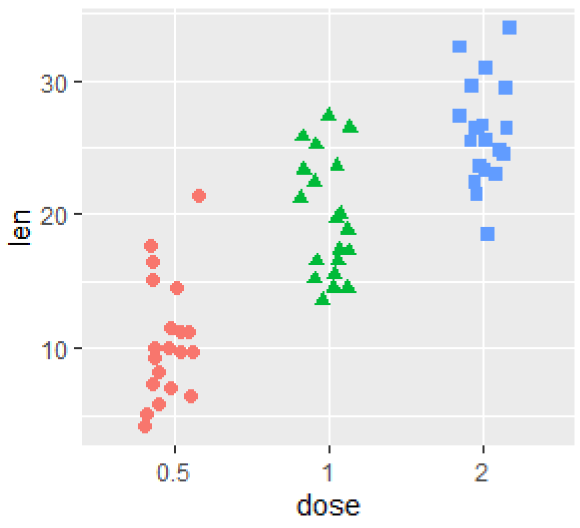
```
> #Change the legend position
> p + theme(legend.position="top")
```



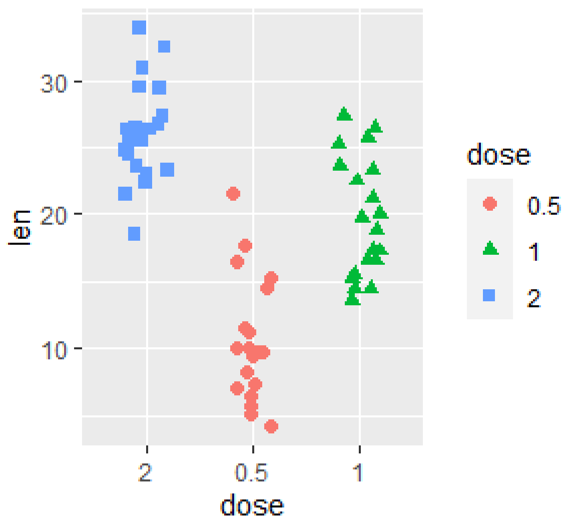
```
> p + theme(legend.position="bottom")
```



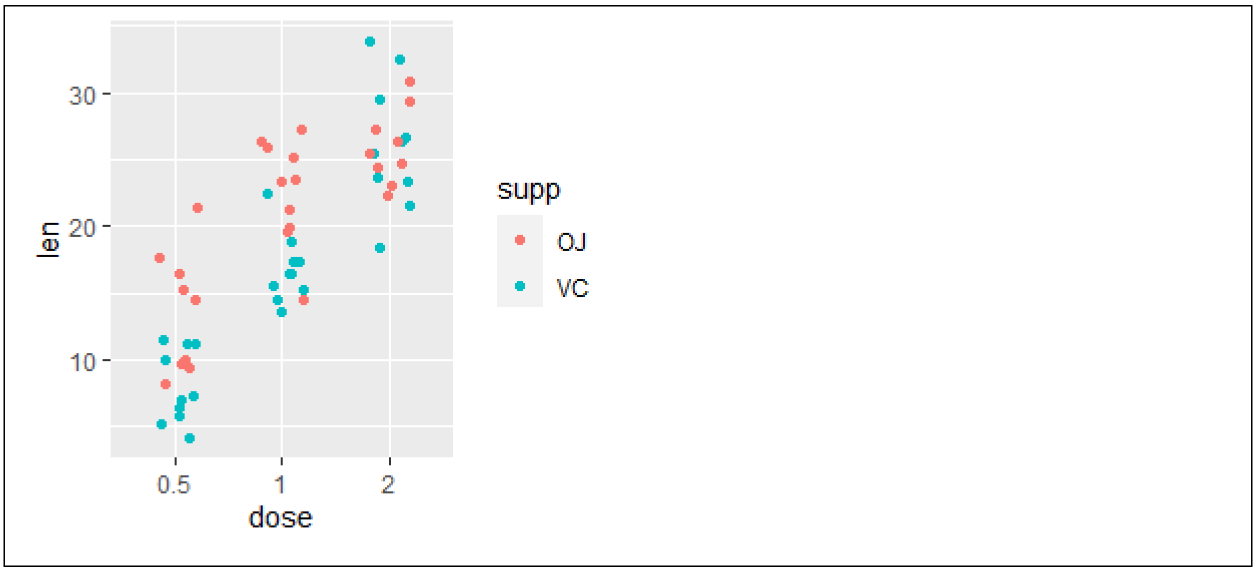
```
> p + theme(legend.position="none")
```



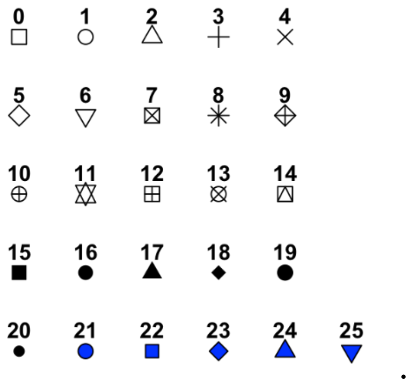
```
> #Change the order of items in the legend
> p + scale_x_discrete(limits=c("2", "0.5", "1"))
```



```
# Change stripchart colors by groups
ggplot(ToothGrowth, aes(x=dose, y=len, color=supp)) +
geom_jitter(position=position_jitter(0.2))
```



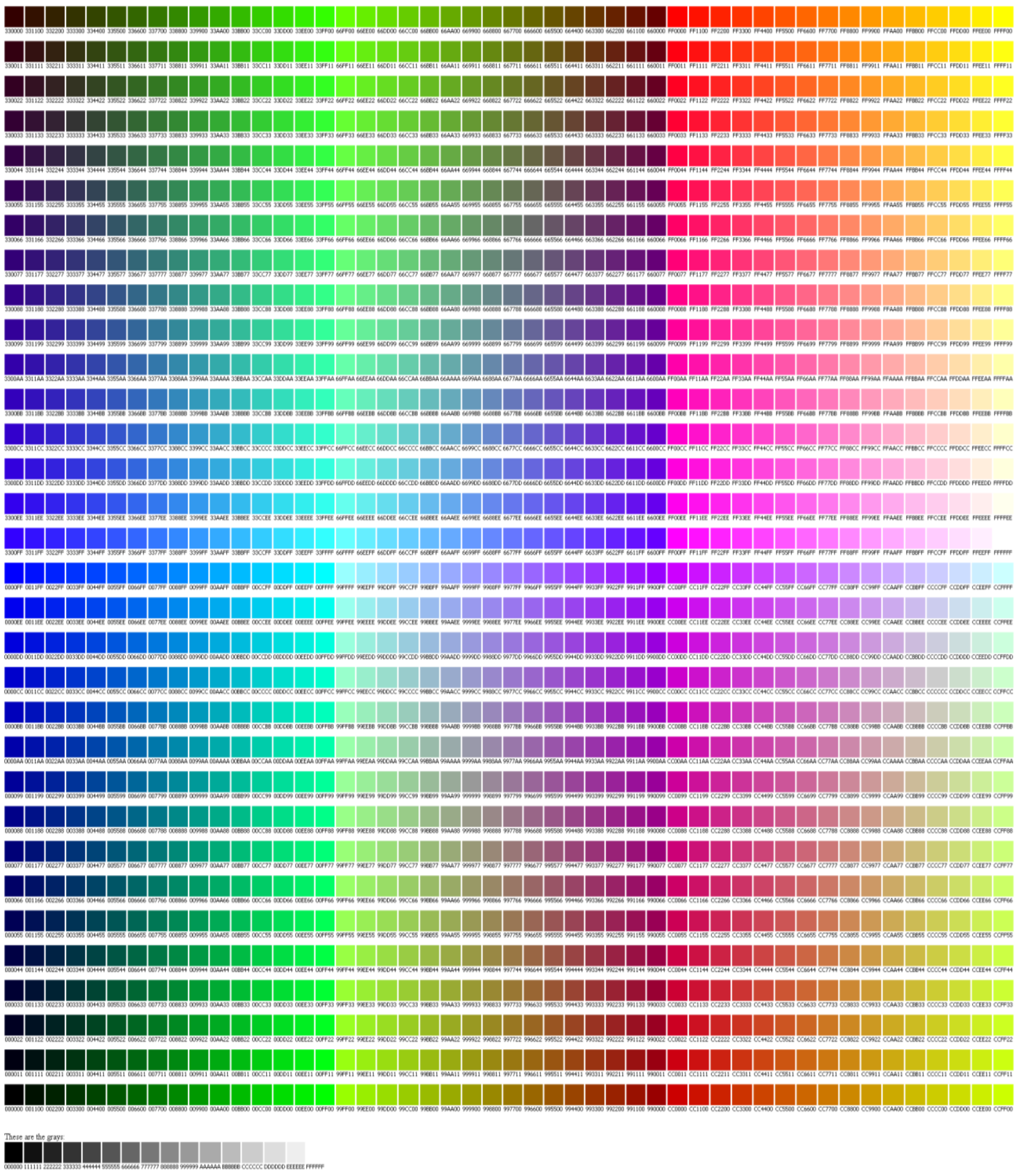
It adds a small amount of random variation to the location of each point, and is a useful way of handling overplotting caused by discreteness in smaller datasets.



You can choose one of the 657 named colors in R

brown4	darkorange4	gray	gray57	hotpink3	lightsalmon4	navajowhite1	plum3	slategray3	antiquewhite
brown3	darkorange3	goldenrod4	gray56	hotpink2	lightsalmon3	navajowhite	plum2	slategray2	aliceblue
brown2	darkorange2	goldenrod3	gray55	hotpink1	lightsalmon2	moccasin	plum1	slategray1	white
brown1	darkorange1	goldenrod2	gray54	hotpink	lightsalmon1	mistyrose4	plum	slategray	yellowgreen
brown	darkorange	goldenrod1	gray53	honeydew4	lightsalmon	mistyrose3	pink4	slateblue4	yellow4
blueviolet	darkolivegreen4	goldenrod	gray52	honeydew3	lightpink4	mistyrose2	pink3	slateblue3	yellow3
blue4	darkolivegreen3	gold4	gray51	honeydew2	lightpink3	mistyrose1	pink2	slateblue2	yellow2
blue3	darkolivegreen2	gold3	gray50	honeydew1	lightpink2	mistyrose	pink1	slateblue1	yellow1
blue2	darkolivegreen1	gold2	gray49	honeydew	lightpink1	mintcream	pink	slateblue	yellow
blue1	darkolivegreen	gold1	gray48	greenyellow	lightpink	midnightblue	peru	skyblue4	whitesmoke
blue	darkmagenta	gold	gray47	green4	lightgrey	mediumvioletred	peachpuff4	skyblue3	wheat4
blanchedalmond	darkkhaki	ghostwhite	gray46	green3	lightgreen	mediumturquoise	peachpuff3	skyblue2	wheat3
black	darkgrey	gainsboro	gray45	green2	lightgray	mediumspringgreen	peachpuff2	skyblue1	wheat2
bisque4	darkgreen	forestgreen	gray44	green1	lightgoldenrodyellow	mediumslateblue	peachpuff1	skyblue	wheat1
bisque3	darkgray	floralwhite	gray43	green	lightgoldenrod4	mediumseagreen	peachpuff	sienna4	wheat
bisque2	darkgoldenrod4	firebrick4	gray42	gray100	lightgoldenrod3	mediumpurple4	papayawhip	sienna3	violetred4
bisque1	darkgoldenrod3	firebrick3	gray41	gray99	lightgoldenrod2	mediumpurple3	palevioletred4	sienna2	violetred3
bisque	darkgoldenrod2	firebrick2	gray40	gray98	lightgoldenrod1	mediumpurple2	palevioletred3	sienna1	violetred2
beige	darkgoldenrod1	firebrick1	gray39	gray97	lightgoldenrod	mediumpurple1	palevioletred2	sienna	violetred1
azure4	darkgoldenrod	firebrick	gray38	gray96	lightcyan4	mediumpurple	palevioletred1	seashell4	violetred
azure3	darkcyan	dodgerblue4	gray37	gray95	lightcyan3	mediumorchid4	palevioletred	seashell3	violet
azure2	darkblue	dodgerblue3	gray36	gray94	lightcyan2	mediumorchid3	paleturquoise4	seashell2	turquoise4
azure1	cyan4	dodgerblue2	gray35	gray93	lightcyan1	mediumorchid2	paleturquoise3	seashell1	turquoise3
azure	cyan3	dodgerblue1	gray34	gray92	lightcyan	mediumorchid1	paleturquoise2	seashell	turquoise2
aquamarine4	cyan2	dodgerblue	gray33	gray91	lightcoral	mediumorchid	paleturquoise1	seagreen4	turquoise1
aquamarine3	cyan1	dimgrey	gray32	gray90	lightblue4	mediumblue	paleturquoise	seagreen3	turquoise
aquamarine2	cyan	dimgray	gray31	gray89	lightblue3	mediumaquamarine	palegreen4	seagreen2	tomato4
aquamarine1	cornsilk4	deepskyblue4	gray30	gray88	lightblue2	maroon4	palegreen3	seagreen1	tomato3
aquamarine	cornsilk3	deepskyblue3	gray29	gray87	lightblue1	maroon3	palegreen2	seagreen	tomato2
antiquewhite4	cornsilk2	deepskyblue2	gray28	gray86	lightblue	maroon2	palegreen1	sandybrown	tomato1
antiquewhite3	cornsilk1	deepskyblue1	gray27	gray85	lemonchiffon4	maroon1	palegreen	salmon4	tomato
antiquewhite2	cornsilk	deepskyblue	gray26	gray84	lemonchiffon3	maroon	palegoldenrod	salmon3	thistle4
antiquewhite1	cornflowerblue	deeppink4	gray25	gray83	lemonchiffon2	magenta4	orchid4	salmon2	thistle3
antiquewhite	coral4	deeppink3	gray24	gray82	lemonchiffon1	magenta3	orchid3	salmon1	thistle2
aliceblue	coral3	deeppink2	gray23	gray81	lemonchiffon	magenta2	orchid2	salmon	thistle1
white	coral2	deeppink1	gray22	gray80	lawngreen	magenta1	orchid1	saddlebrown	thistle
bisque3	coral1	deeppink	gray21	gray79	lavenderblush4	magenta	orchid	royalblue4	tan4
bisque2	coral	darkviolet	gray20	gray78	lavenderblush3	linen	orangered4	royalblue3	tan3
bisque1	chocolate4	darkturquoise	gray19	gray77	lavenderblush2	limegreen	orangered3	royalblue2	tan2
bisque	chocolate3	darkslategrey	gray18	gray76	lavenderblush1	lightyellow4	orangered2	royalblue1	tan1
beige	chocolate2	darkslategray4	gray17	gray75	lavenderblush	lightyellow3	orangered1	royalblue	tan
azure4	chocolate1	darkslategray3	gray16	gray74	lavender	lightyellow2	orangered	rosybrown4	steelblue4
azure3	chocolate	darkslategray2	gray15	gray73	khaki4	lightyellow1	orange4	rosybrown3	steelblue3
azure2	chartreuse4	darkslategray1	gray14	gray72	khaki3	lightyellow	orange3	rosybrown2	steelblue2
azure1	chartreuse3	darkslategray	gray13	gray71	khaki2	lightsteelblue4	orange2	rosybrown1	steelblue1
azure	chartreuse2	darkslateblue	gray12	gray70	khaki1	lightsteelblue3	orange1	rosybrown	steelblue
aquamarine4	chartreuse1	darkseagreen4	gray11	gray69	khaki	lightsteelblue2	orange	red4	springgreen4
aquamarine3	chartreuse	darkseagreen3	gray10	gray68	ivory4	lightsteelblue1	olivedrab4	red3	springgreen3
aquamarine2	cadetblue4	darkseagreen2	gray9	gray67	ivory3	lightsteelblue	olivedrab3	red2	springgreen2
aquamarine1	cadetblue3	darkseagreen1	gray8	gray66	ivory2	lightslategrey	olivedrab2	red1	springgreen1
aquamarine	cadetblue2	darkseagreen	gray7	gray65	ivory1	lightslategray	olivedrab1	red	springgreen
antiquewhite4	cadetblue1	darksalmon	gray6	gray64	ivory	lightslateblue	olivedrab	purple4	snow4
antiquewhite3	cadetblue	darkred	gray5	gray63	indianred4	lightskyblue4	oldlace	purple3	snow3
antiquewhite2	burlywood4	darkorchid4	gray4	gray62	indianred3	lightskyblue3	navyblue	purple2	snow2
antiquewhite1	burlywood3	darkorchid3	gray3	gray61	indianred2	lightskyblue2	navy	purple1	snow1
antiquewhite	burlywood2	darkorchid2	gray2	gray60	indianred1	lightskyblue1	navajowhite4	purple	snow
aliceblue	burlywood1	darkorchid1	gray1	gray59	indianred	lightskyblue	navajowhite3	powderblue	slategrey
white	burlywood	darkorchid	gray0	gray58	hotpink4	lightseagreen	navajowhite2	plum4	slategray4

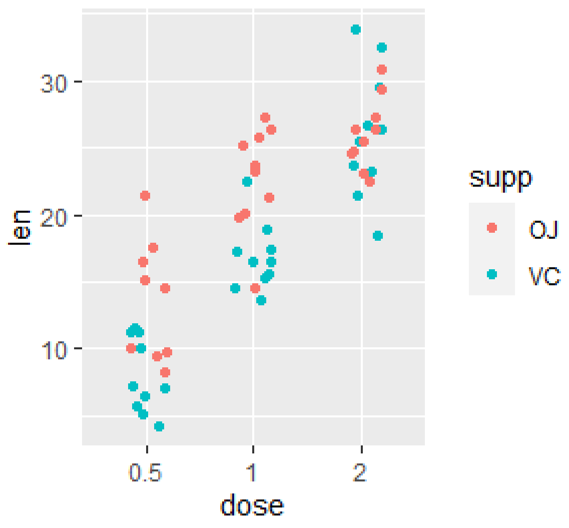
And if you are still in search of colors, you can use the hexadecimal codes.



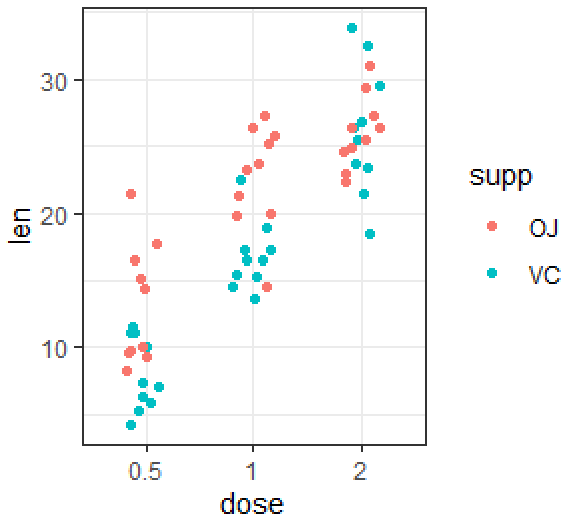
Themes in ggplot

Themes are a powerful way to customize the non-data components of your plots: i.e. titles, labels, fonts, background, gridlines, and legends. Themes can be used to give plots a consistent customized look.

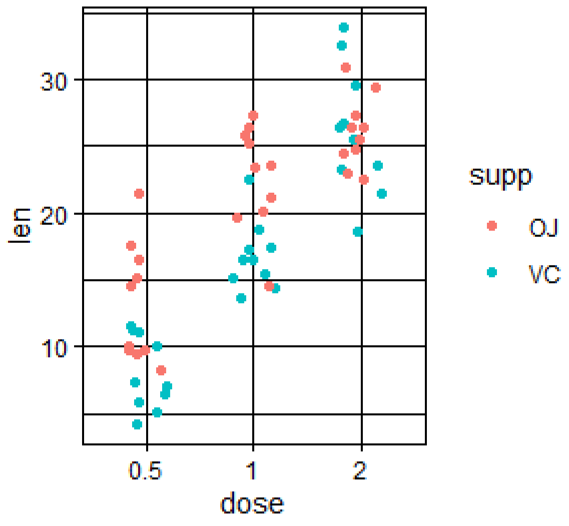
```
> p + theme_gray()
```



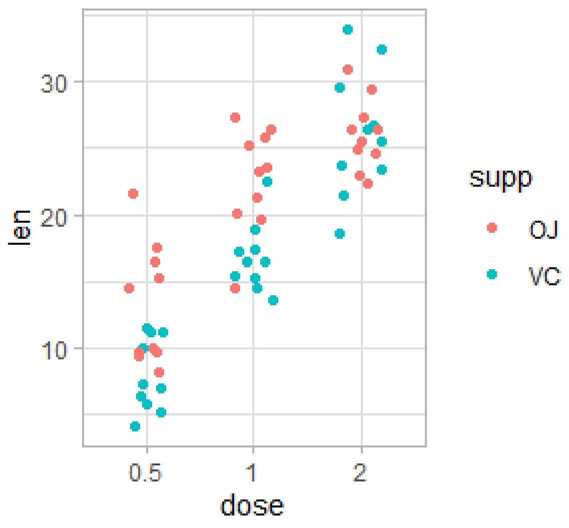
```
> p + theme_bw()
```



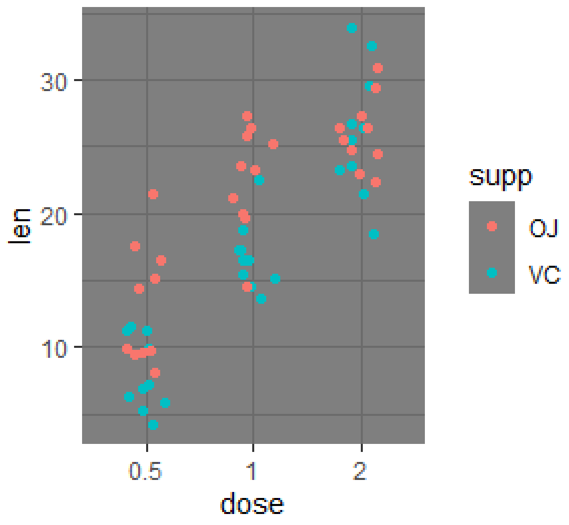
```
> p + theme_linedraw()
```



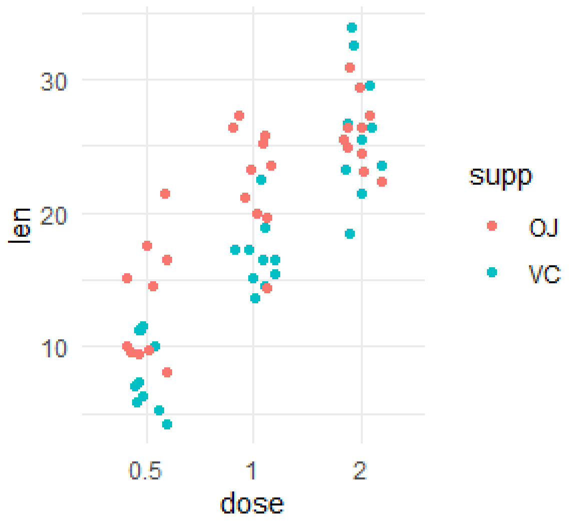
```
> p + theme_light()
```



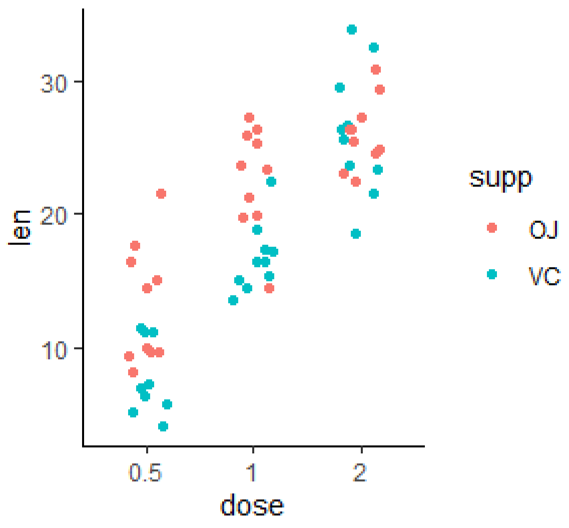
```
> p + theme_dark()
```



```
> p + theme_minimal()
```



```
> p + theme_classic()
```

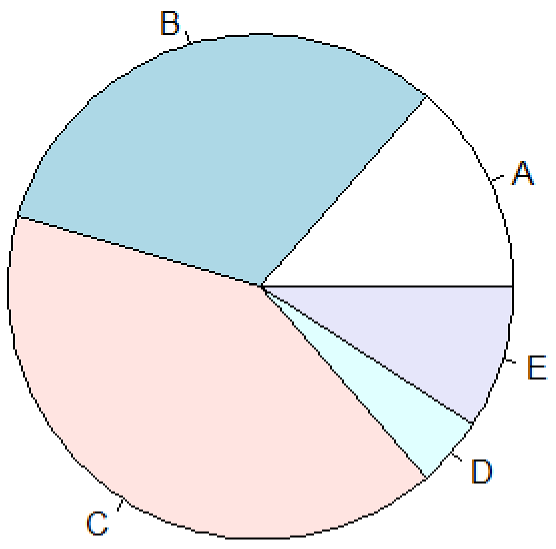


To modify an individual theme component you can use code like `plot + theme (element.name = element_function())`.

Pie Chart

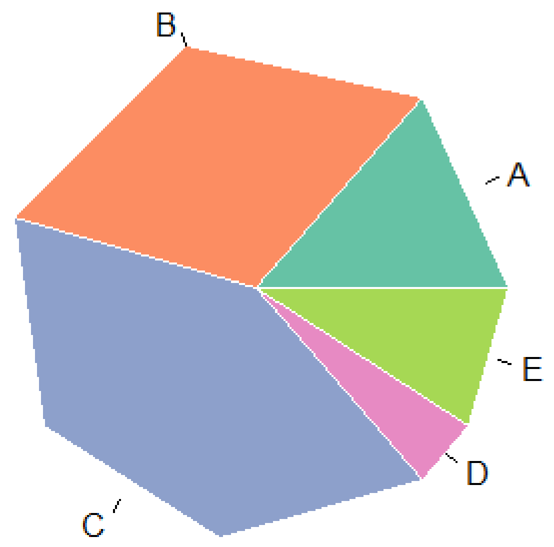
A pie chart is a circle divided into sectors that each represent a proportion of the whole.

```
> # PIE CHART
>countdata<- c(3,7,9,1,2)
> species = c("A","B","C","D","E")
> pie(countdata , species)
```



```
> #Non-circular piechart
```

```
>install.packages("RColorBrewer")
> library(RColorBrewer)
>myPalette<- brewer.pal(5, "Set2")
> pie(countdata , labels = species, border="white", col=myPalette, edges = 10 )
```



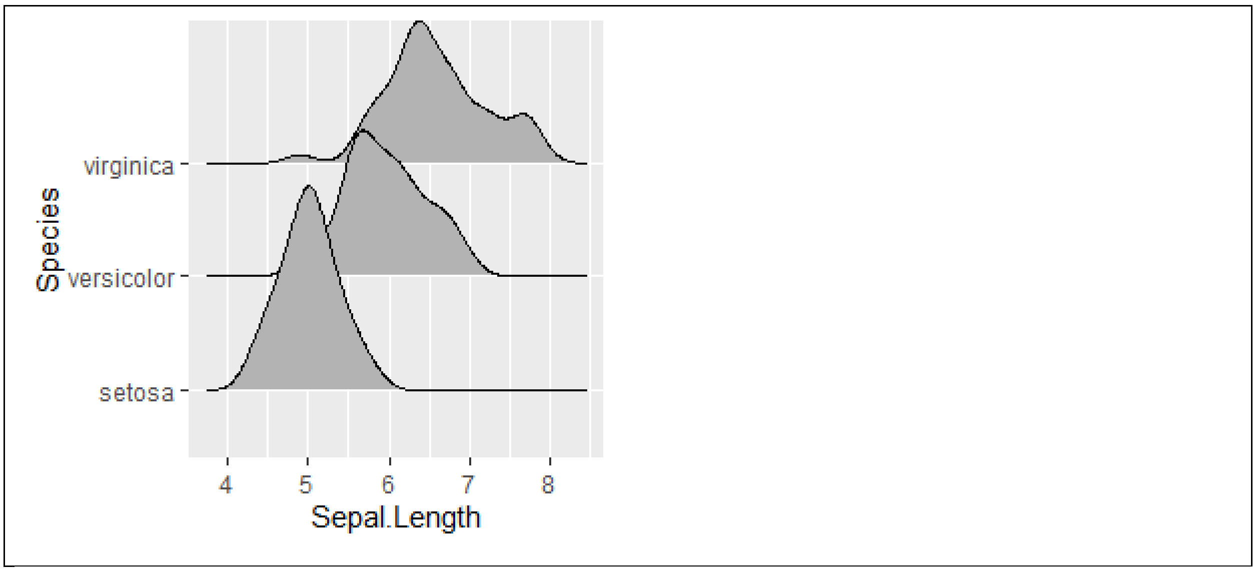
Ridgeline Plot

Dataset: Iris

Description: 50 samples from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals, in centimeters.

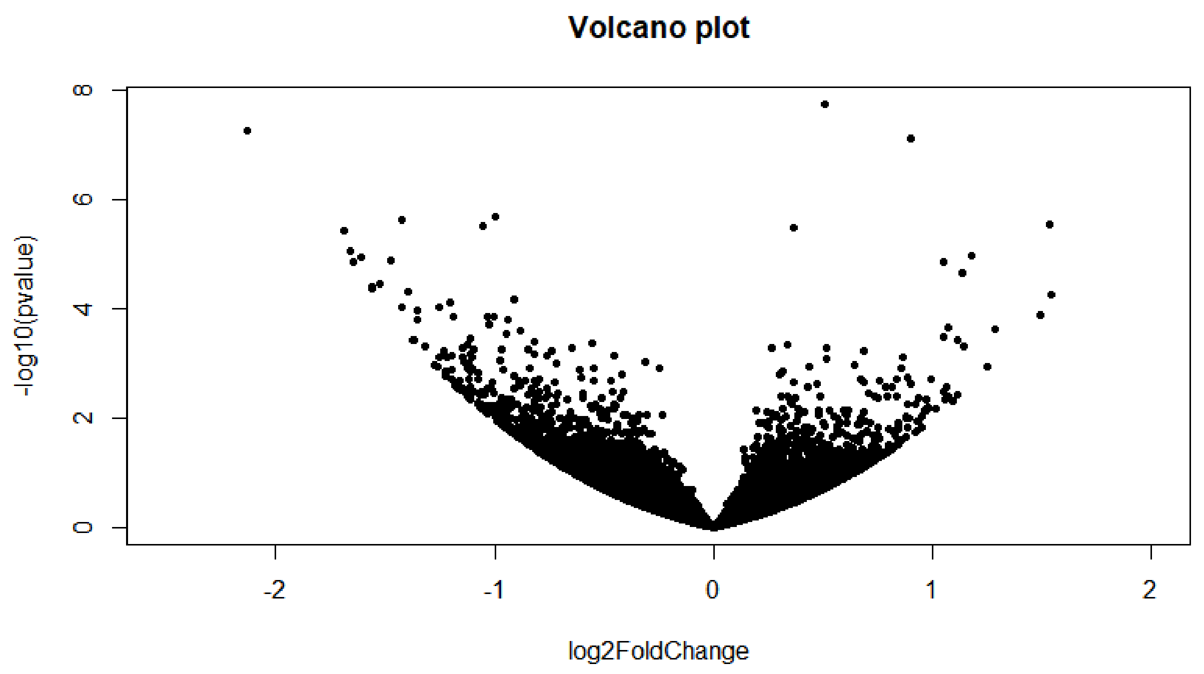
Format: 150 observations of 4 features

```
library(ggridges)
ggplot(iris, aes(x = Sepal.Length, y = Species)) + geom_density_ridges()
>ggplot(iris, aes(x = Sepal.Length, y = Species)) + geom_density_ridges()
Picking joint bandwidth of 0.181
```

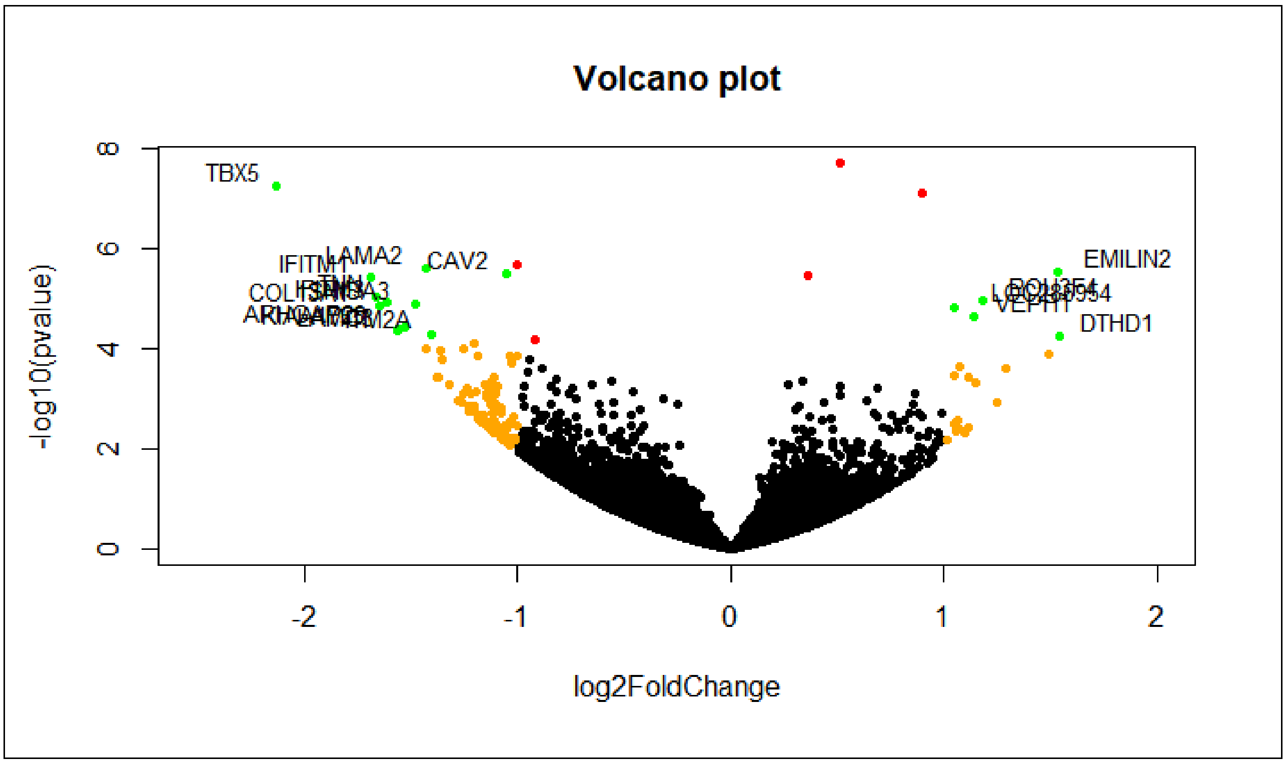


Volcano Plot

```
>res <- read.csv("C:/Users/user/Desktop/Workshop/material/volcano.txt", sep="",
stringsAsFactors=TRUE)
> head(res)
    Gene log2FoldChangepvaluepadj
1    DOK6        0.5100 1.861e-08 0.0003053
2    TBX5       -2.1290 5.655e-08 0.0004191
3 SLC32A1        0.9003 7.664e-08 0.0004191
4  IFITM1       -1.6870 3.735e-06 0.0068090
5   NUP93        0.3659 3.373e-06 0.0068090
6 EMILIN2        1.5340 2.976e-06 0.0068090
# Make a basic volcano plot
with(res, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot", xlim=c(-
2.5,2)))
```

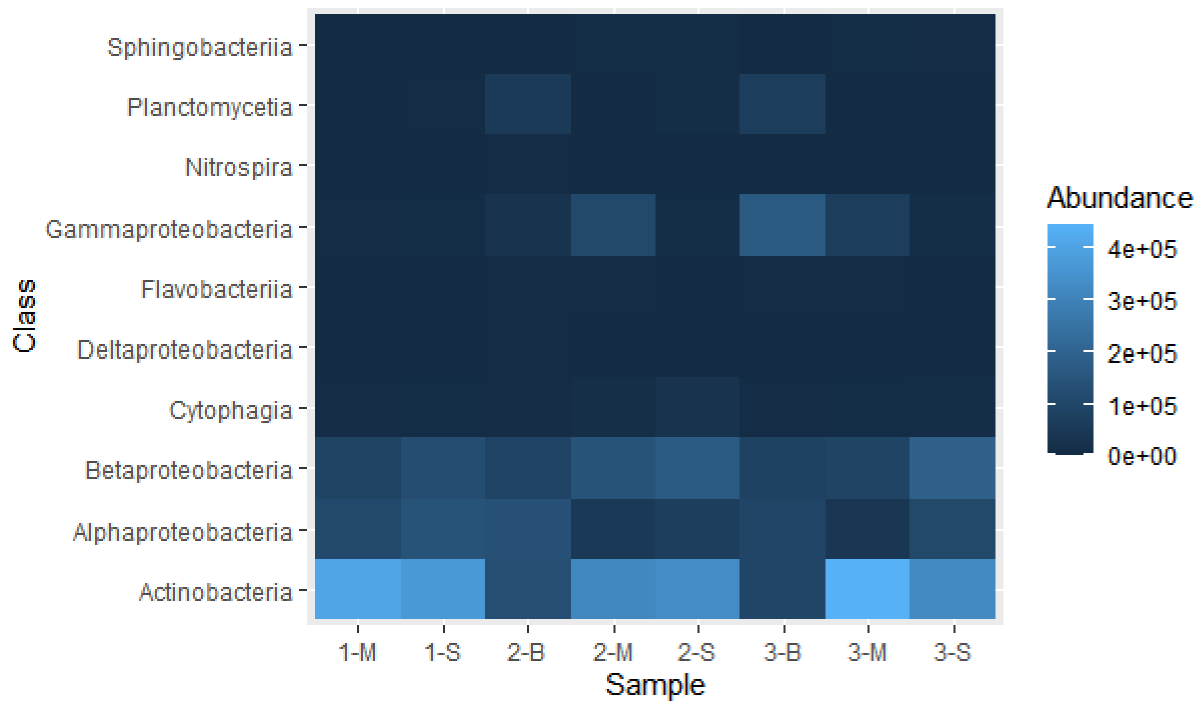


```
# Add colored points: red if padj<0.05, orange of log2FC>1, green if both)
with(subset(res, padj<.05 ), points(log2FoldChange, -log10(pvalue), pch=20,
col="red"))
with(subset(res, abs(log2FoldChange)>1), points(log2FoldChange, -log10(pvalue),
pch=20, col="orange"))
with(subset(res, padj<.05 & abs(log2FoldChange)>1), points(log2FoldChange, -
log10(pvalue), pch=20, col="green"))

# Label points with the textxy function from the calibrate plot
install.packages("calibrate")
library(calibrate)
with(subset(res, padj<.05 & abs(log2FoldChange)>1), textxy(log2FoldChange, -
log10(pvalue), labs=Gene, cex=.8))
```

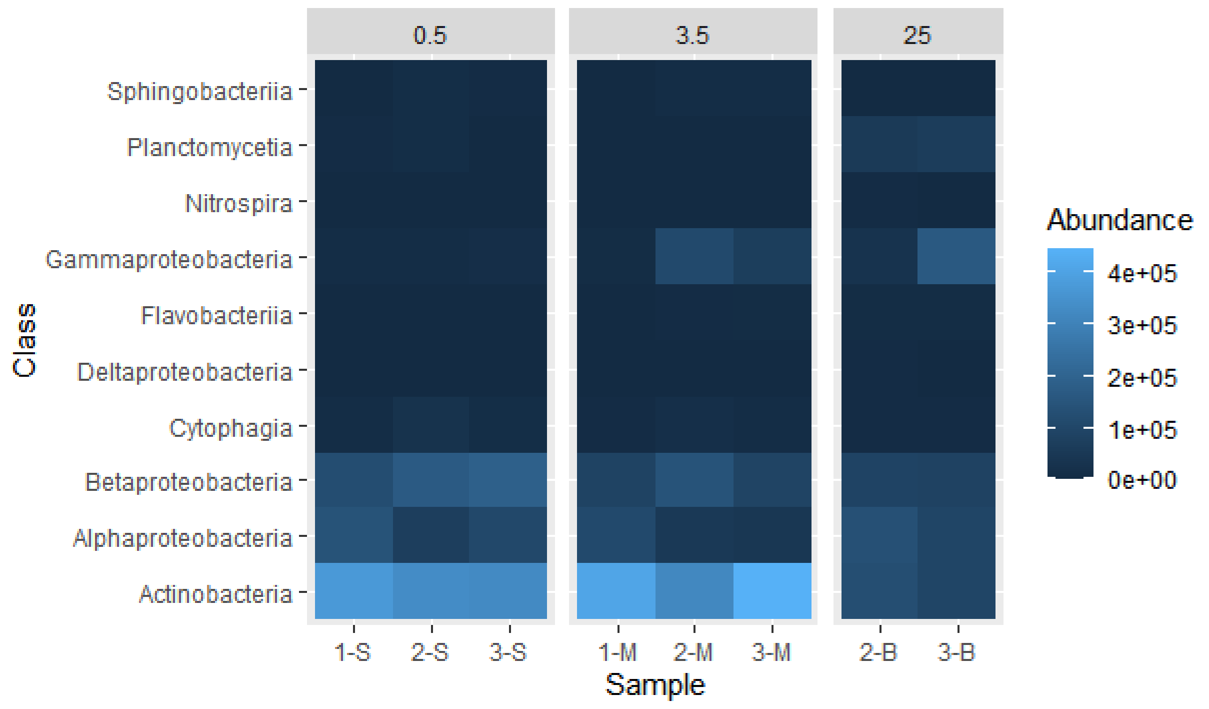


Heatmap

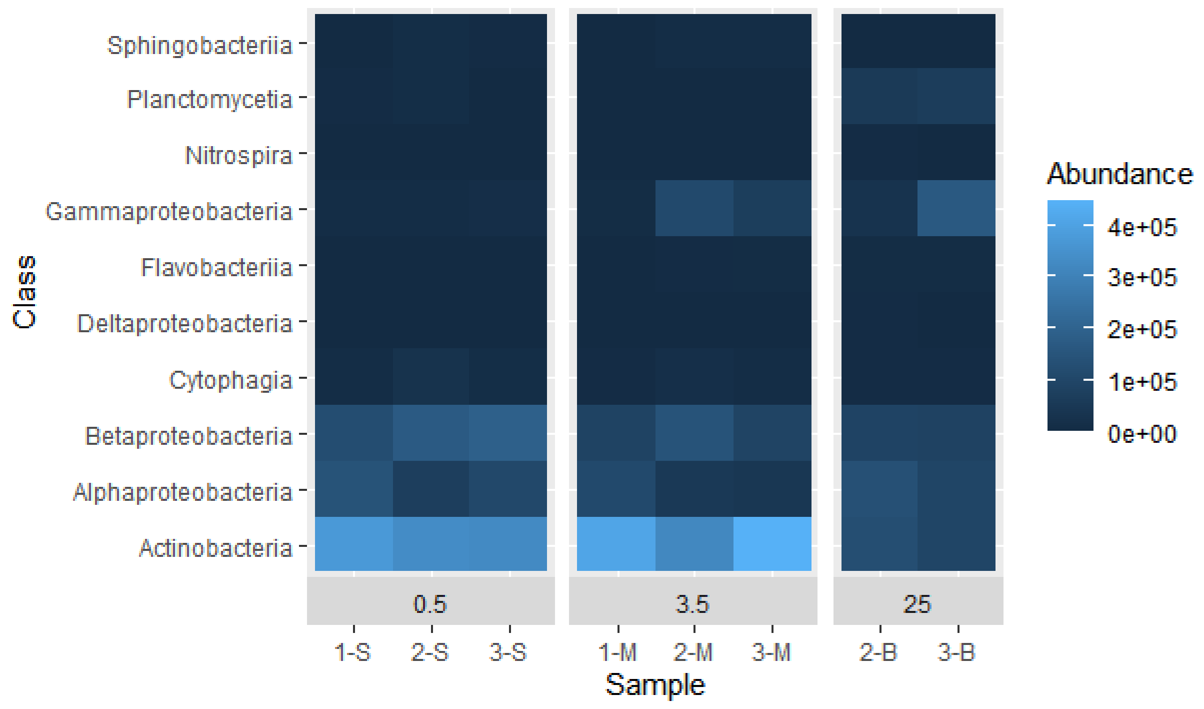
```
library("ggplot2")
>heatdata<- read.csv(file = "C:/Users/user/Desktop/Workshop/material/heat1.csv")
>heatmap<- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                    y = Class,
+                                    fill = Abundance)) +
+   geom_tile() +
+   xlab(label = "Sample")
>heatmap
```



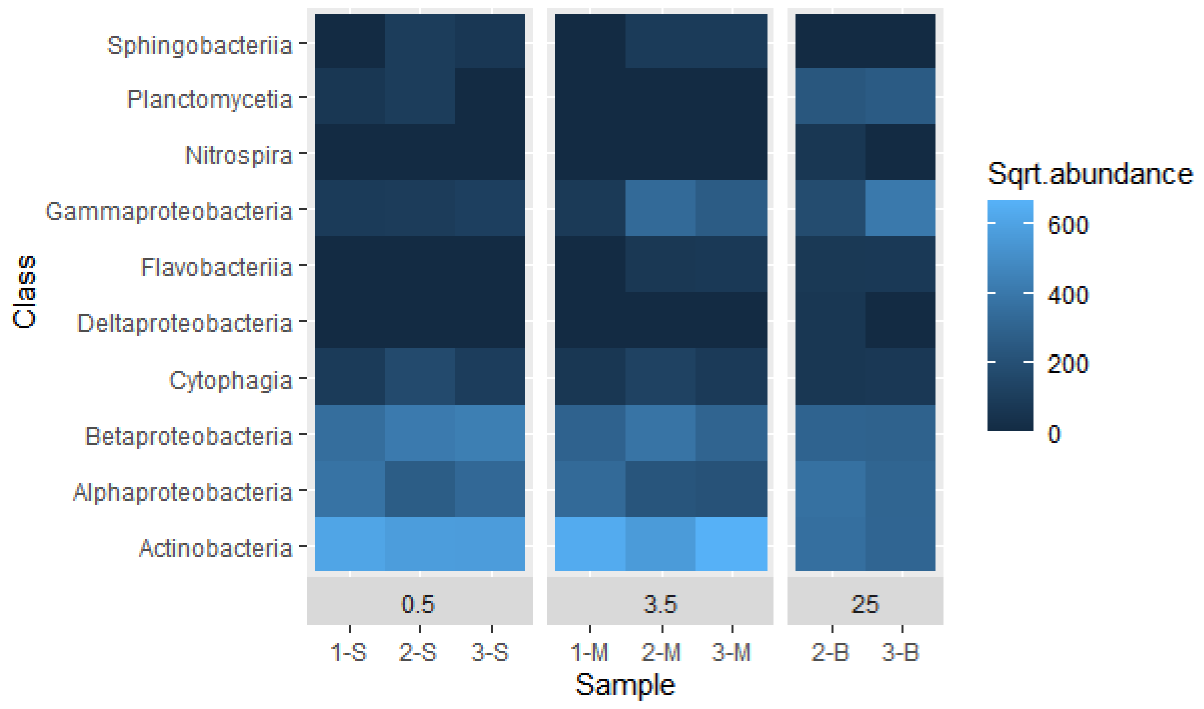
```
> #faceting the heatmap
>facetheatmap<- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                   y = Class,
+                                   fill = Abundance)) +
+  geom_tile() +
+  xlab(label = "Sample") +
+  facet_grid(~ Depth, scales = "free_x", space = "free_x")
>facetheatmap
```



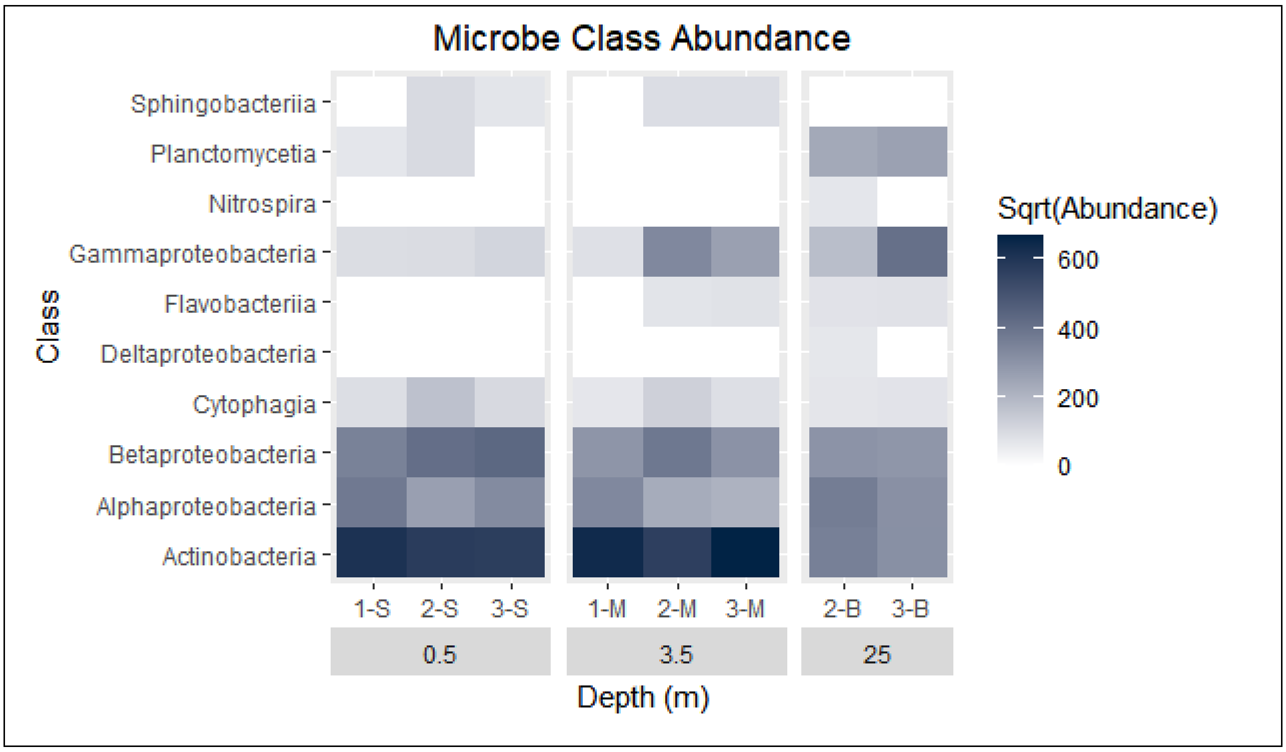
```
> #switching the labels
>labelchange_heatmap<- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                       y = Class,
+                                       fill = Abundance)) +
+   geom_tile() +
+   xlab(label = "Sample") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x")
>labelchange_heatmap
```



```
#scaling the data
>heatdata$Sqrt.abundance<- sqrt(heatdata$Abundance)
>scaleheatmap<- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                      y = Class,
+                                      fill = Sqrt.abundance)) +
+   geom_tile() +
+   xlab(label = "Sample") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x")
>scaleheatmap
```



```
#add title
>heatdata$Sqrt.abundance<- sqrt(heatdata$Abundance)
>titleheatmap<- ggplot(data = heatdata, mapping = aes(x = Sample.name,
+                                          y = Class,
+                                          fill = Sqrt.abundance)) +
+   geom_tile() +
+   xlab(label = "Depth (m)") +
+   facet_grid(~ Depth, switch = "x", scales = "free_x", space = "free_x") +
+   scale_fill_gradient(name = "Sqrt(Abundance)",
+                   low = "#FFFFFF",
+                   high = "#012345") +
+   theme(strip.placement = "outside",
+       plot.title = element_text(hjust = 0.5)) +
+   ggtitle(label = "Microbe Class Abundance")
>titleheatmap
```



Circos Plot/ Idiogram

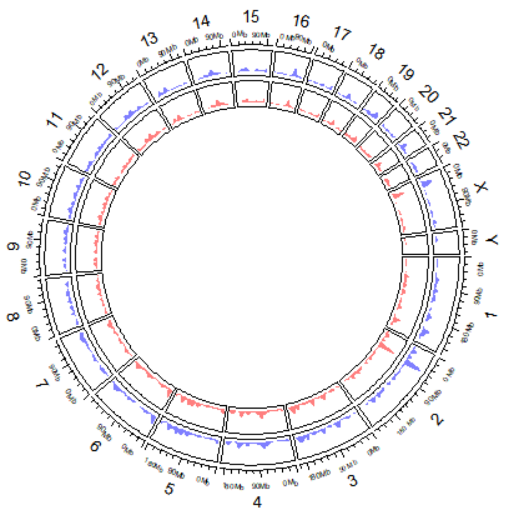
Dataset: Two files with names location_c.txt and location_nc.txt in BED format

Description: Location of coding and non-coding regions in the genome

Format:BED

```
> library(circlize)
>circos.initializeWithIdeogram(plotType = c("labels", "axis"))
>location_nc<-
read.delim("C:/Users/user/Desktop/Workshop/material/location_nc.txt",
stringsAsFactors=TRUE)
>location_c<-
read.delim("C:/Users/user/Desktop/Workshop/material/location_c.txt",
stringsAsFactors=TRUE)
>circos.genomicDensity(location_nc, col = c("#0000FF80"), track.height = 0.1)
Warning message:
Some of the regions have end position values larger than the end of the
chromosomes.
>circos.genomicDensity(location_c, col = c("#FF000080"), track.height = 0.1)
```

```
Warning message:
Some of the regions have end position values larger than the end of the
chromosomes.
```



Topic 3: Installation of Ubuntu 20.04 on Windows

Fayad M A[1] and Merlin Lopez[2]
[1] Research Scholar (Bioinformatics Cell), ICAR-Indian Institute of Spices Research, Kozhikode, Kerala
[2] Scientist (Bioinformatics), Community Agrobiodiversity Centre, MS Swaminathan Research Foundation, Wayanad, Kerala

Introduction

Windows is a pervasive operating system that is used on multiple platforms. However, Linux users, most programmers, and creative professionals tend to use Ubuntu over Windows.

Ubuntu is a very stable and flexible operating system and a Debian-based Linux distribution consisting mainly of free and open-source software. There are different versions of Ubuntu, and we can install any of them on our system. We can install it alone or on a virtual machine. In this writing piece, we will explore how to install "Ubuntu 20.04 on Windows".

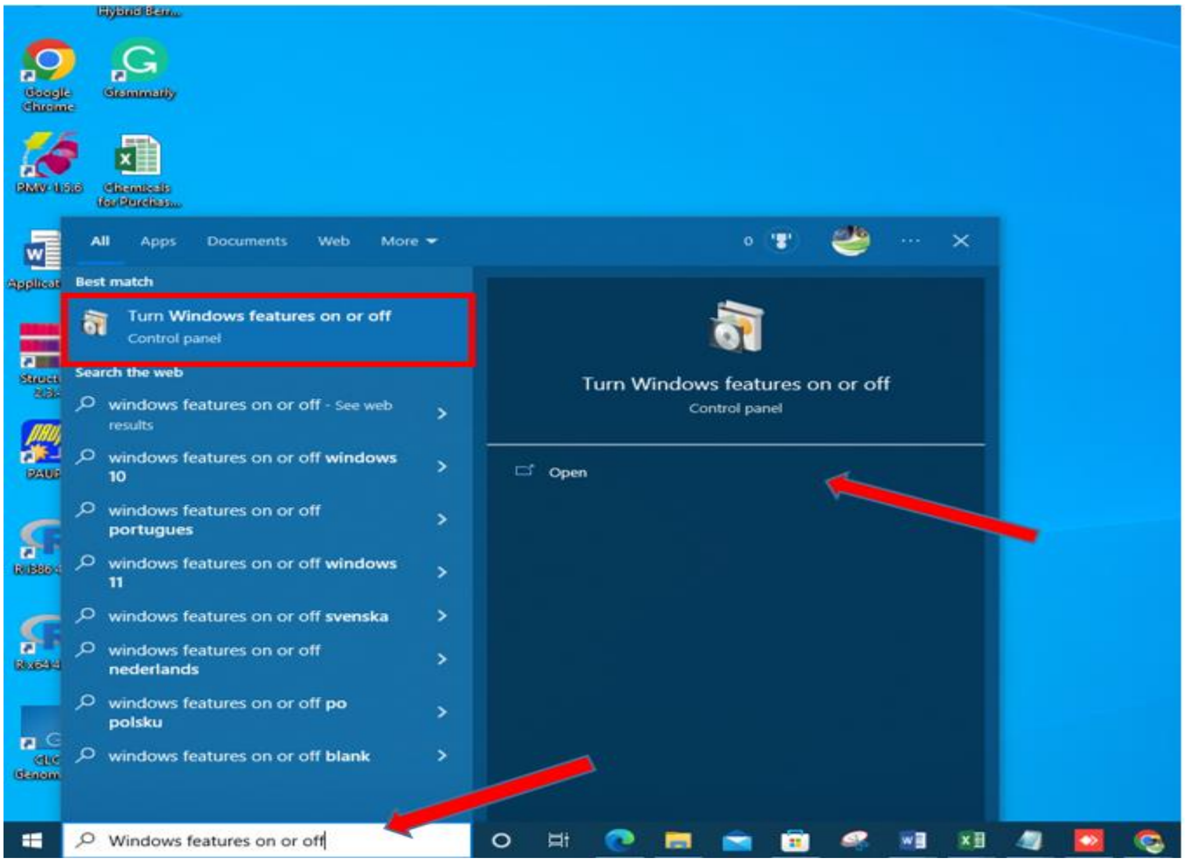
Recommended system requirements:

- 2 GHz dual-core processor or better
- 4 GB system memory
- 25 GB of free hard drive space
- Internet access is helpful
- Either a DVD drive or a USB port for the installer media

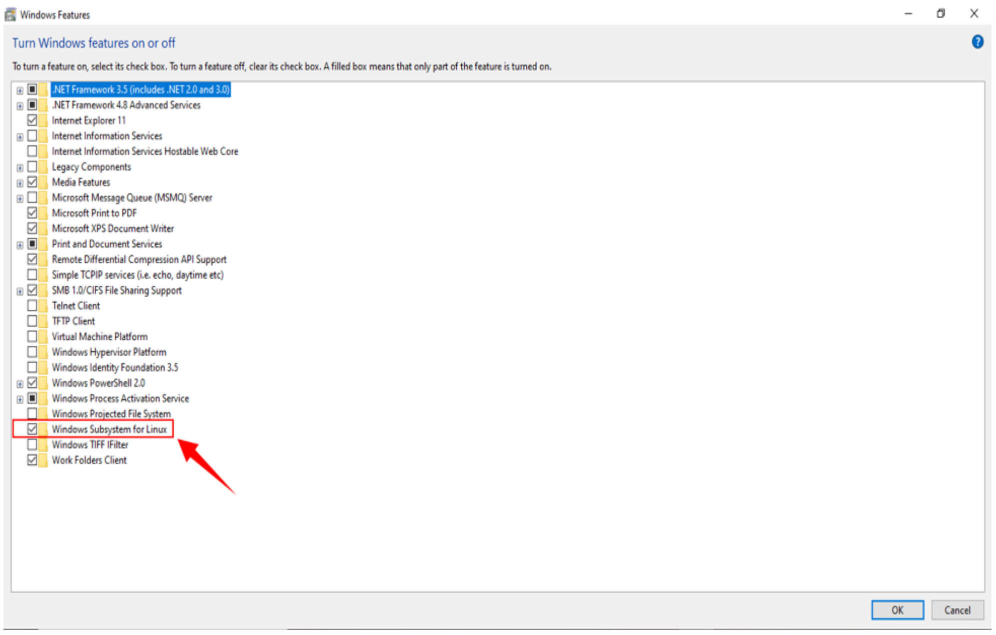
Installation Process

Enable Windows Subsystem for Linux (WSL)

➢ First, Enter "Turn Windows features on or off" in the Window search bar.


All
Apps
Documents
Web
More
Best match
Turn Windows features on or off
Control panel
Search the web
windows features on or off - See web results
windows features on or off windows 10
windows features on or off portugues
windows features on or off windows 11
windows features on or off svenska
windows features on or off nederlands
windows features on or off po polsku
windows features on or off blank
Turn Windows features on or off
Control panel
Open
Windows features on or off

➢ Locate "Windows subsystem for Linux". We need to mark this check box "Windows Subsystem for Linux". Press "OK" to install this feature.


Windows Features

Turn Windows features on or off

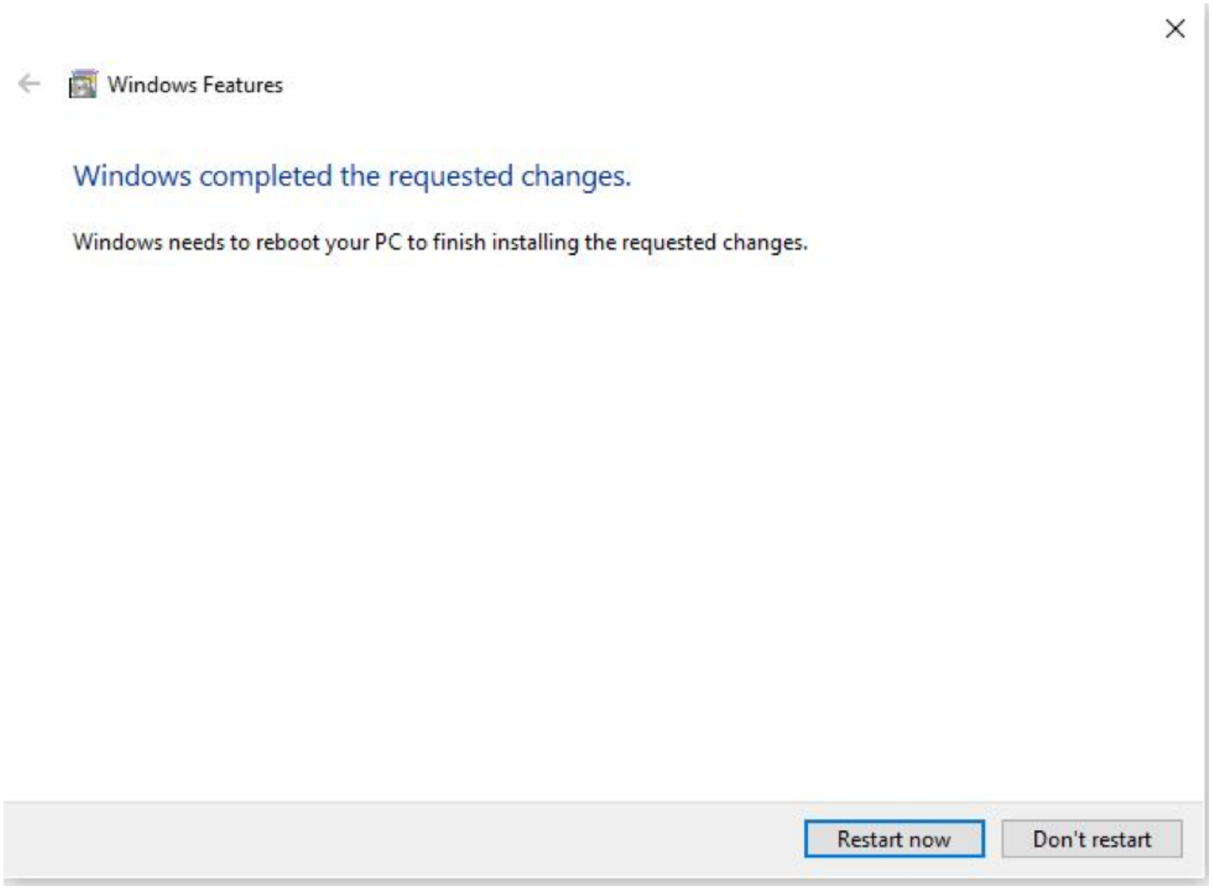
To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.

- ▣ .NET Framework 3.5 (includes .NET 2.0 and 3.0)
- ▣ .NET Framework 4.8 Advanced Services
- ☑ Internet Explorer 11
- ☐ Internet Information Services
- ☐ Internet Information Services Hostable Web Core
- ☐ Legacy Components
- ☑ Media Features
- ☐ Microsoft Message Queue (MSMQ) Server
- ☑ Microsoft Print to PDF
- ☑ Microsoft XPS Document Writer
- ▣ Print and Document Services
- ☑ Remote Differential Compression API Support
- ☐ Simple TCPIP services (i.e. echo, daytime etc)
- ☑ SMB 1.0/CIFS File Sharing Support
- ☐ Telnet Client
- ☐ TFTP Client
- ☐ Virtual Machine Platform
- ☐ Windows Hypervisor Platform
- ☐ Windows Identity Foundation 3.5
- ☑ Windows PowerShell 2.0
- ▣ Windows Process Activation Service
- ☐ Windows Projected File System
- ☑ Windows Subsystem for Linux
- ☐ Windows TIFF IFilter
- ☑ Work Folders Client

➢ It takes a couple of moments to enable the WSL.

- When WSL is enabled, we need to restart our system to finish the requested changes.

- Click “Restart now”.

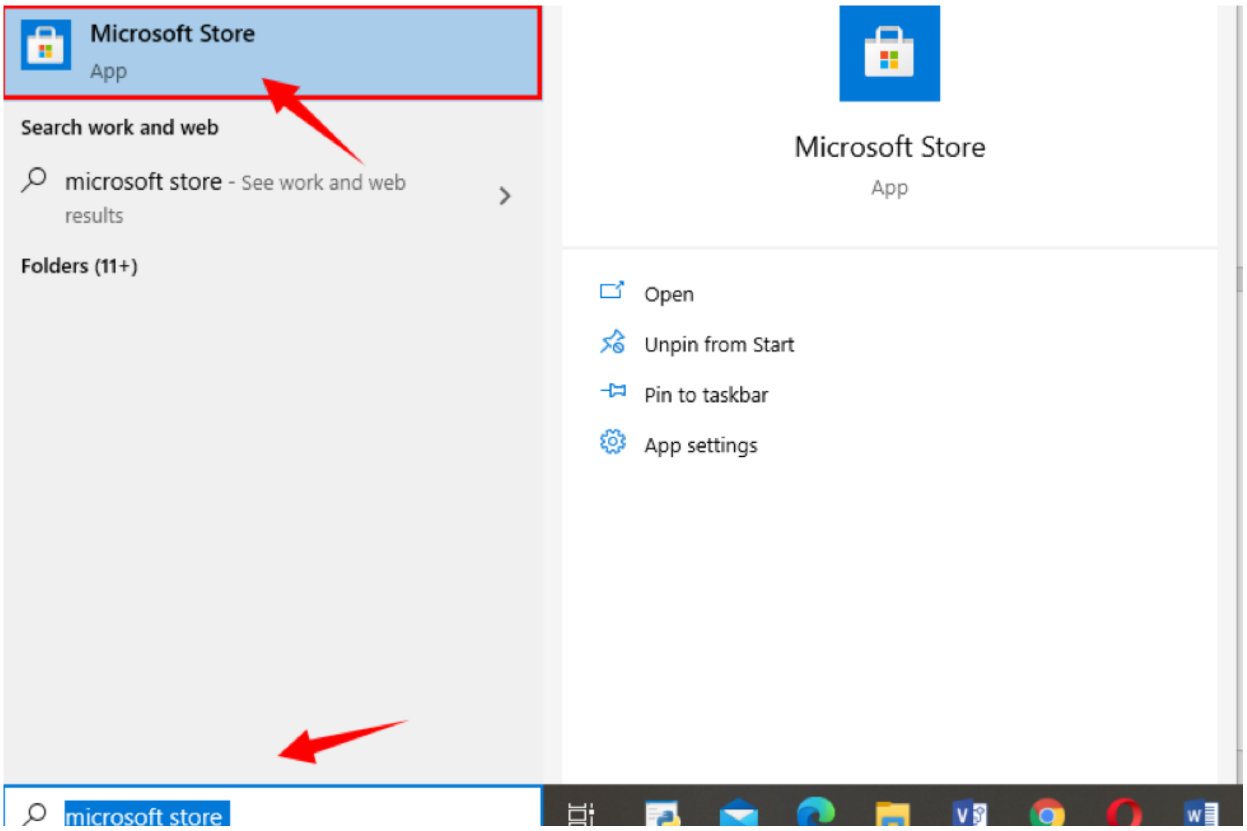

Windows Features

Windows completed the requested changes.

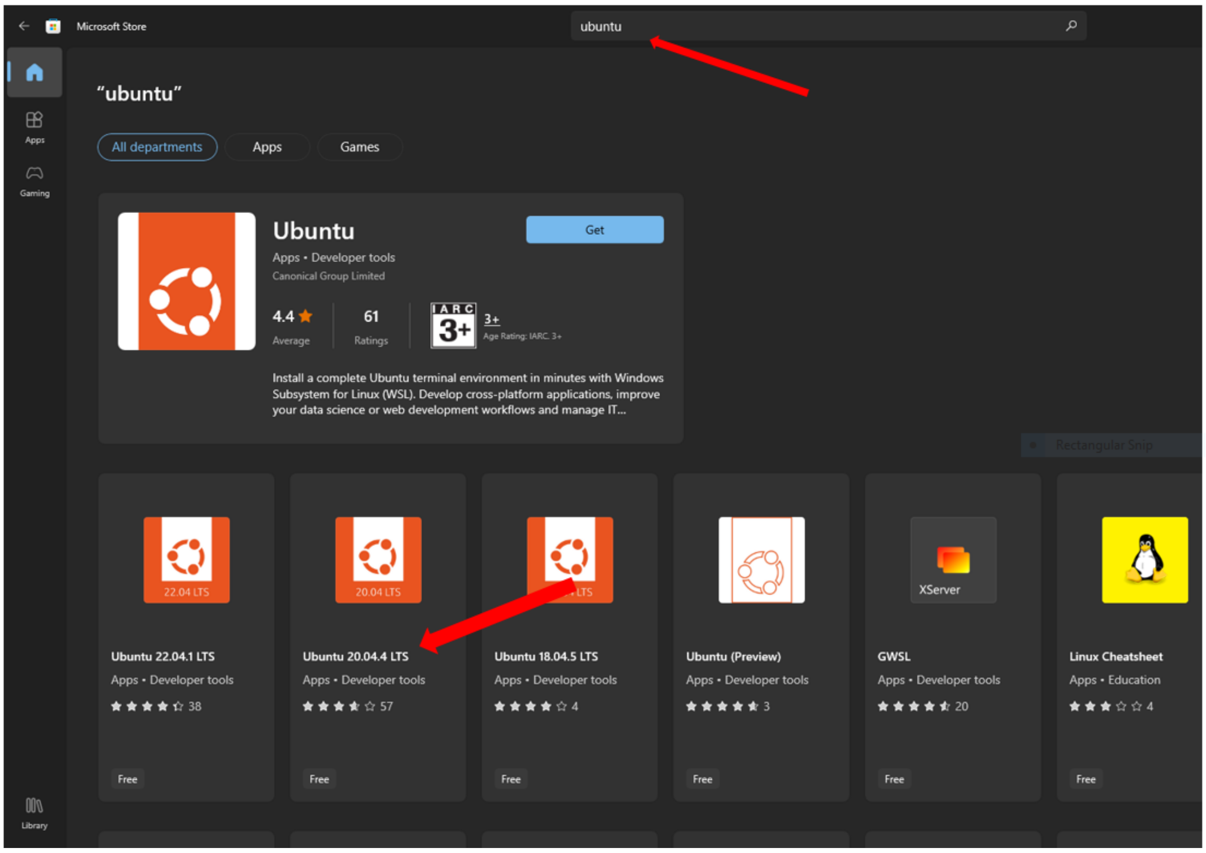
Windows needs to reboot your PC to finish installing the requested changes.

Download and Install Ubuntu 20.04 on window via Microsoft store

- Type “Microsoft Store” on the Windows Search Bar.

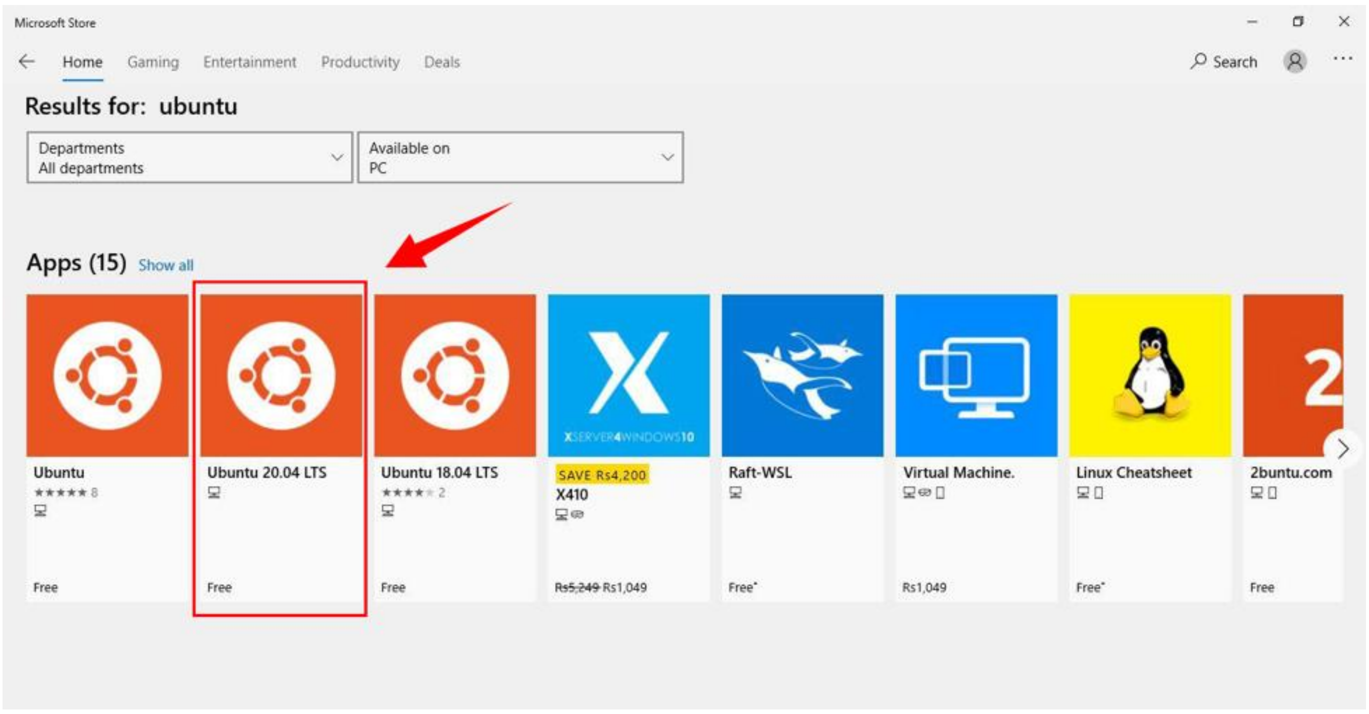


- When the Microsoft store opens, there is a search bar. Type “Ubuntu”.

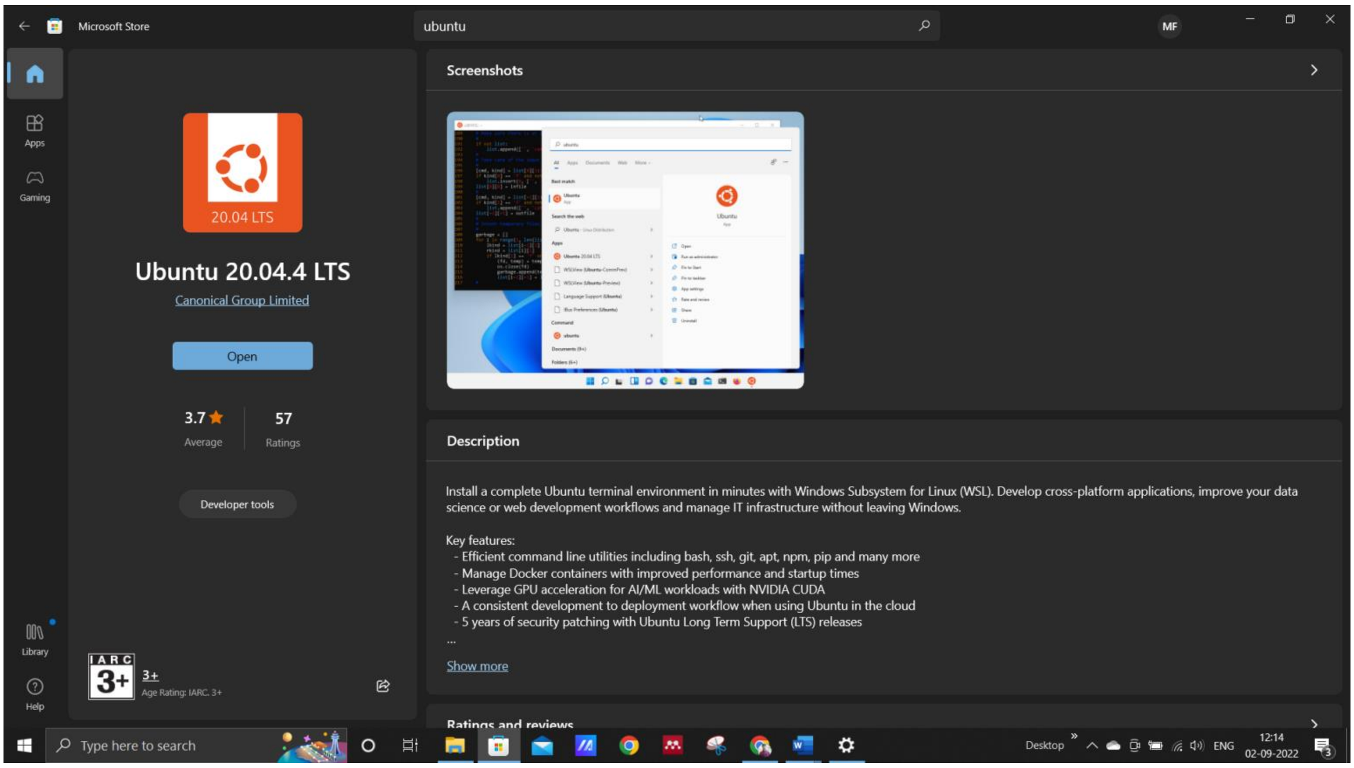


Install a complete Ubuntu terminal environment in minutes with Windows Subsystem for Linux (WSL). Develop cross-platform applications, improve your data science or web development workflows and manage IT...

- Different Ubuntu apps will be displayed. Select Ubuntu 20.04 from the given applications.



➢ Press "Get" to install the application. Downloading will start.



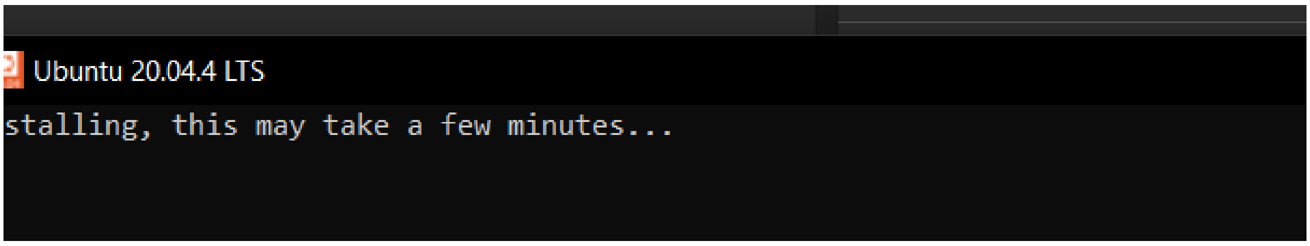
Install a complete Ubuntu terminal environment in minutes with Windows Subsystem for Linux (WSL). Develop cross-platform applications, improve your data science or web development workflows and manage IT infrastructure without leaving Windows.

Key features:
- Efficient command line utilities including bash, ssh, git, apt, npm, pip and many more
- Manage Docker containers with improved performance and startup times
- Leverage GPU acceleration for AI/ML workloads with NVIDIA CUDA
- A consistent development to deployment workflow when using Ubuntu in the cloud
- 5 years of security patching with Ubuntu Long Term Support (LTS) releases

...

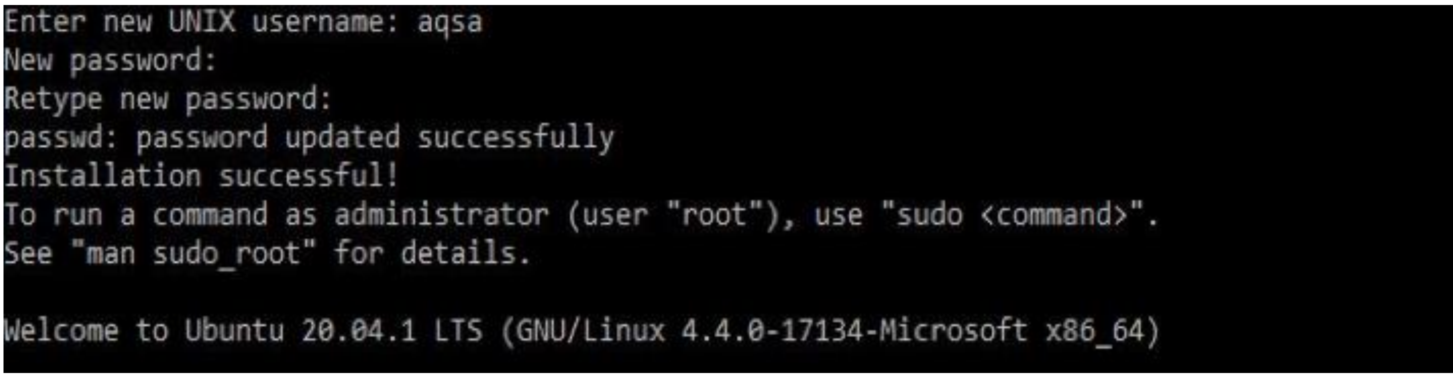
➢ Upon downloading click "Open".

- When Ubuntu is installed for the first time, the terminal window will open, which shows that Ubuntu 20.04 is being installed, and we need to hold on for a while.



```
Ubuntu 20.04.4 LTS
stalling, this may take a few minutes...
```

- Upon installation, we will be asked for a username.
- Give any specific username (Don't use uppercase).
- Press "enter".
- Enter "password" and then enter again (Password is not show in terminal).
- The message will appear, "password updated".



```
Enter new UNIX username: aqsa
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 4.4.0-17134-Microsoft x86_64)
```

- Now we can run any command on Linux prompt.

Ubuntu 20.04 terminal is ready for use on Windows 10.

Topic 4: Introduction to Linux: Hands on Practice

Merlin Lopez[1] and Fayad M A[2]
[1]Scientist (Bioinformatics), Community Agrobiodiversity Centre, MS Swaminathan Research Foundation, Wayanad, Kerala
[2]Research Scholar (Bioinformatics Cell), ICAR-Indian Institute of Spices Research, Kozhikode, Kerala

Introduction

The Linux command is a utility of the Linux operating system. All basic and advanced tasks can be done by executing commands. The commands are executed on the Linux terminal. The terminal is a command-line interface to interact with the system, which is similar to the command prompt in the Windows OS. Commands in Linux are case-sensitive.

Linux provides a powerful command-line interface compared to other operating systems such as Windows and MacOS. We can do basic work and advanced work through its terminal. We can do some basic tasks such as creating a file, deleting a file, moving a file, and more. In addition, we can also perform advanced tasks such as administrative tasks (including package installation, user management), networking tasks (ssh connection), security tasks, and many more.

Some of the basic Linux commands

- **Listingfilesanddirectories(ls)**

Whenyoufirstlogin,yourcurrentworkingdirectoryisyourhomedirectory.Yourhomedirectory has the same name as your user-name, for example, *nye1*, and it is whereyourpersonalfilesandsubdirectories aresaved.

Tofindoutwhatis inyour homedirectorytype

```
$ ls
```

Thelscommandliststhecontentsofyourcurrentworkingdirectory.

Important options

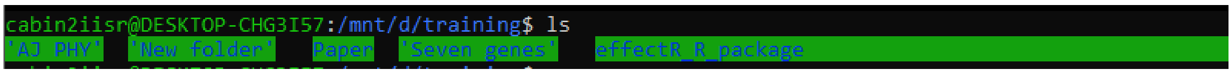
-a	list also files/directories which begin with a dot (hidden)
-l	long listing format. Displays permissions, user and group, time stamp, size, etc.

-R for directories, all sub-directories will be displayed recursively.

.. list the contents of the parent directory one level above

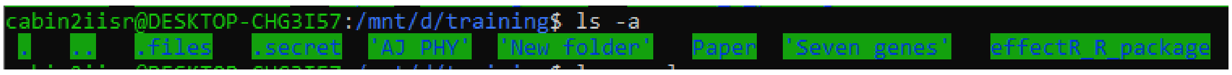
<u>Example</u>

$ ls



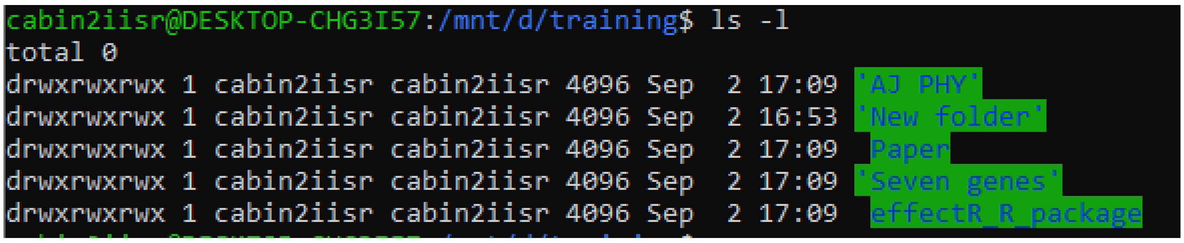
```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package
```

$ ls –a



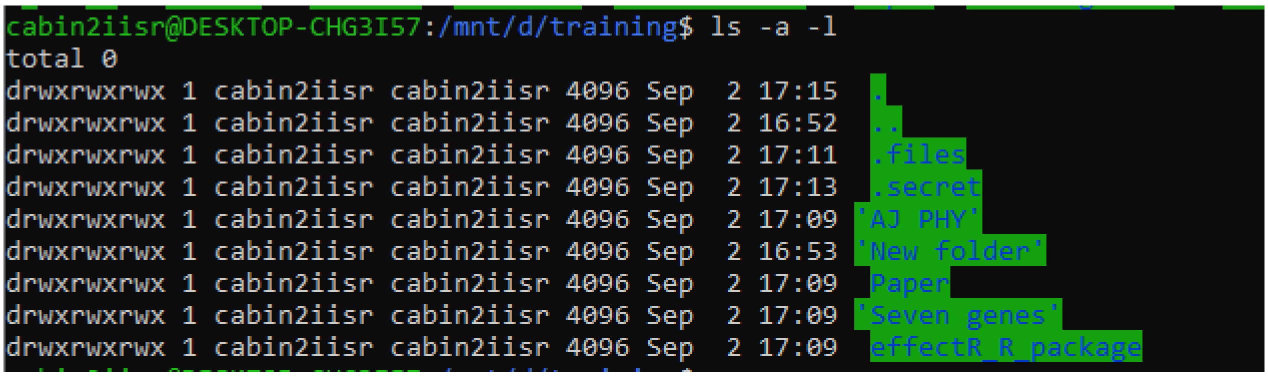
```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training$ ls -a
 .   ..   .files   .secret  'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package
```

$ ls –l



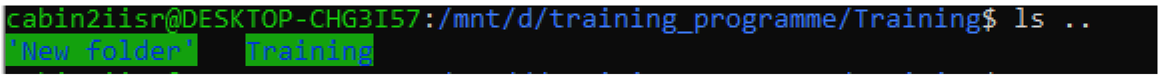
```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training$ ls -l
total 0
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09 'AJ PHY'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 16:53 'New folder'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09  Paper
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09 'Seven genes'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09  effectR_R_package
```

$ ls -a -l



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training$ ls -a -l
total 0
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:15  .
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 16:52  ..
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:11  .files
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:13  .secret
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09 'AJ PHY'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 16:53 'New folder'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09  Paper
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09 'Seven genes'
drwxrwxrwx 1 cabin2iisr cabin2iisr 4096 Sep  2 17:09  effectR_R_package
```

$ ls ..



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls ..
'New folder'   Training
```

➢ Making(mkdir)& Removing Directories (rmdir)

The command “mkdir” stands for “make directory”. It creates each directory specifed on the command line in the order given. This command can create multiple directories at once as well as set the permissions for the directories.

The “rmdir” directory is used to remove directories, but only those that are empty (i.e., contain no files or subdirectories)

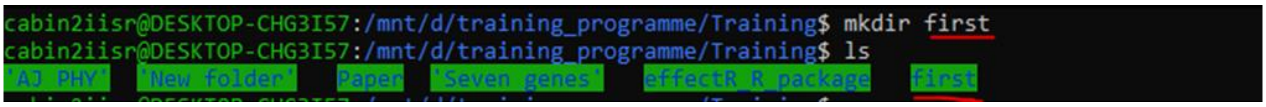
<u>Important options</u>(mkdir)

-v or –verbose:	It displays a message for every directory created.
-p:	A flag which enables the command to create parent directories as necessary. If the directories exist, no error is specified.

<u>Example</u>

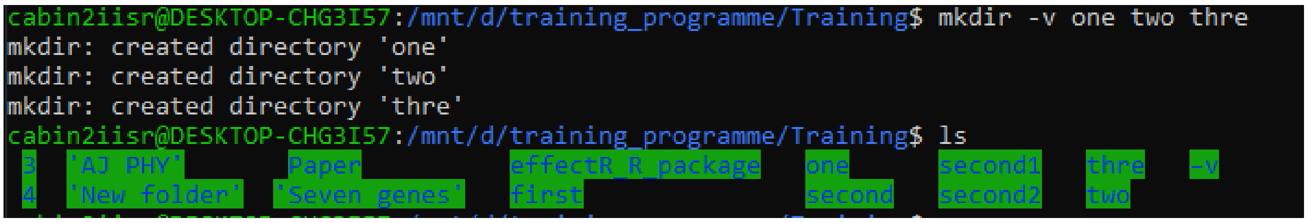
mkdir [Directory name]

“ls” command used to see the file from list



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ mkdir first
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   first
```

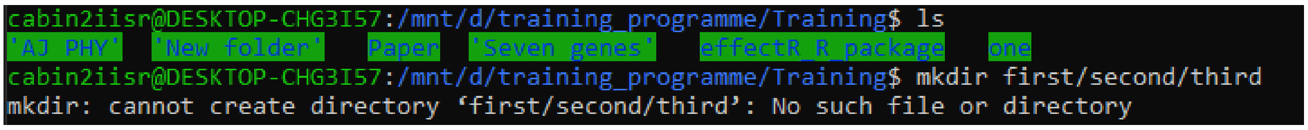
$ mkdir –v one two three



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ mkdir -v one two thre
mkdir: created directory 'one'
mkdir: created directory 'two'
mkdir: created directory 'thre'
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
 3  'AJ PHY'         Paper            effectR_R_package   one      second1   thre   -v
 4  'New folder'   'Seven genes'    first               second   second2   two
```

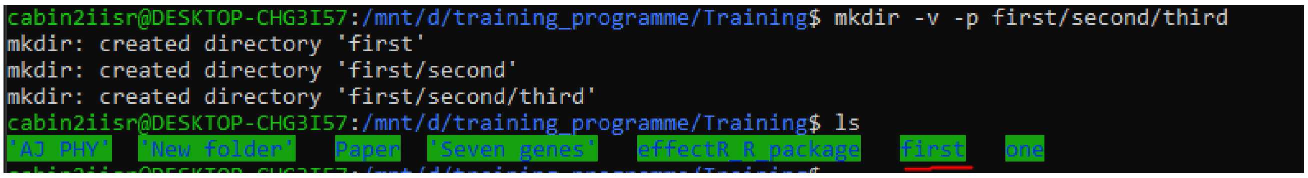
$ mkdir -p first/second/third

If the first and second directories do not exist, due to the -p option, mkdir will create these directories for us. If we do not specify the -p option, and request the creation of directories, where parent directory doesn’t exist, we will get the following output –



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   one
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ mkdir first/second/third
mkdir: cannot create directory ‘first/second/third’: No such file or directory
```

If we specify the -p option, the directories will be created, and no error will be reported. Following is the output of one such execution. We’ve also provided the -v option, so that we can see it in action.



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ mkdir -v -p first/second/third
mkdir: created directory 'first'
mkdir: created directory 'first/second'
mkdir: created directory 'first/second/third'
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   first   one
```

<u>Important options</u>(rmdir)

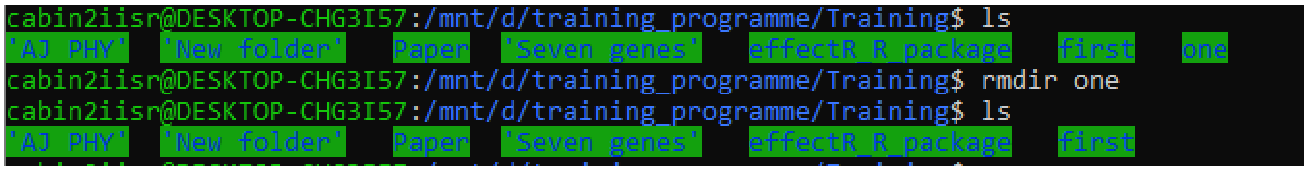
-v or –verbose: It displays a message for every directory deleted.

-p: A flag which enables the command to remove parent directories as well. If the directories exist, no error is specified.

-r: To remove non-empty directories and all the files within them

<u>Example1</u> (Removing directories)

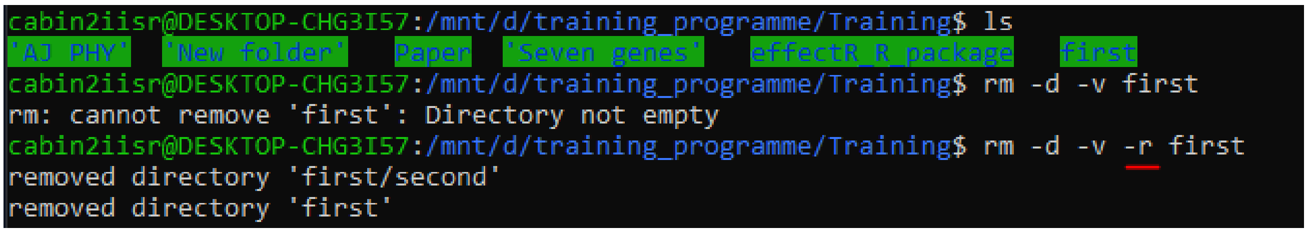
$ rmdir one



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   first   one
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ rmdir one
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   first
```

$ rm -d-v -r first

To remove non-empty directories and all the files within them, use the “rm” command with the “-r”



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   Paper  'Seven genes'   effectR_R_package   first
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ rm -d -v first
rm: cannot remove 'first': Directory not empty
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ rm -d -v -r first
removed directory 'first/second'
removed directory 'first'
```

<u>Example 2</u> (Removing files)

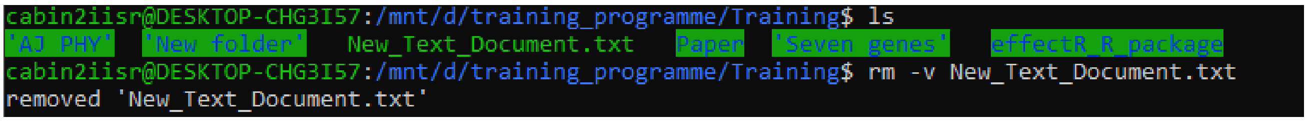
To remove (or delete) a file in Linux from the command line, use either the rm (remove) or unlink command.The unlink command allows you to remove only a single file, while with rm, you can remove multiple files at once.

<u>Be extra careful when removing files or directories, because once the file is deleted it</u>

cannot be easily recovered.

$ unlink filename

$ rm filename



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'  'New folder'   New_Text_Document.txt   Paper  'Seven genes'   effectR_R_package
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ rm -v New_Text_Document.txt
removed 'New_Text_Document.txt'
```

To delete multiple files at once, use the rm command followed by the file names separated by space. You can also use a wildcard (*) and regular expansions to match multiple files. For example, to remove all .pdf files in the current directory, use the following command: (**Caution**: All the .pdf files from the current directory removed, So don't use the following command if the directory had important .pdf files)

$ rm *.pdf

➢ cd command

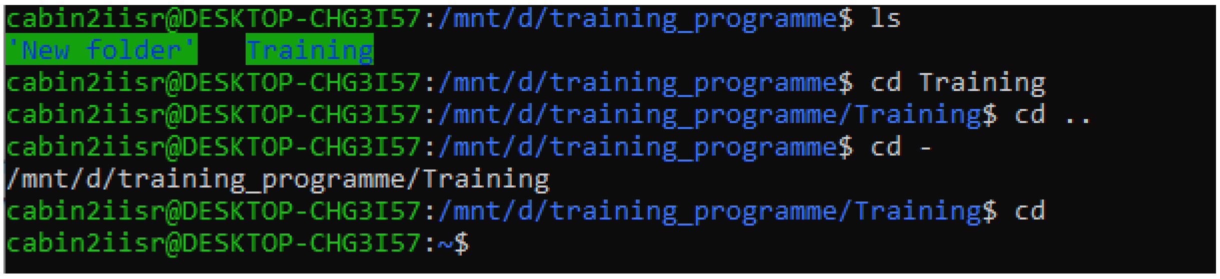
cd command in linux known as change directory command. The **cd** command will allow you to change directories. When you open a terminal you will be in your home directory. To move around the file system you will use **cd**

Important options(cd)

cd or cd ~:	To change directory to the home directory
cd ..:	To move to the parent directory of current directory, or the directory one level up from the current directory. ".." represents parent directory.
cd - :	To navigate to the previous directory (or back)

Example (cd)

cd [directory]

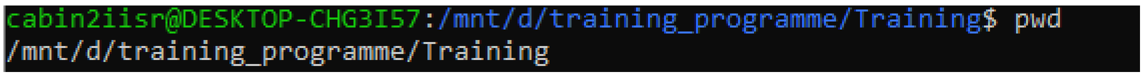


```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme$ ls
'New folder'   Training
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme$ cd Training
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cd ..
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme$ cd -
/mnt/d/training_programme/Training
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cd
cabin2iisr@DESKTOP-CHG3I57:~$
```

- **Pathnames (pwd)**

Pathnames enable you to work out where you are in relation to the whole file-system. The **pwd** command writes to standard output the full path name of your current directory (from the root directory). All directories are separated by a / (slash). The root directory is represented by the first /, and the last directory named is your current directory

$ pwd



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ pwd
/mnt/d/training_programme/Training
```

ShellShortcuts

```
Ctrl-A(jumptostartofline)Ctrl-
E(jumptoendofline)

Ctrl-K(delete(kill)everythingfromthecursoronwardsCtrl-W
(deletethepreviouswordonly)

Ctrl-
Y(pastewhateverwasjustdeleted)Ctrl-C
(kill/exit a running process)Ctrl-L
(clearthescreen)

Ctrl-R(searchforpreviouslyexecutedcommands)
```

Summary

ls	listfilesanddirectories
ls-a	listallfilesanddirectories
mkdir	makeadirectory
cd *directory*	changetonameddirectory
cd	changetohome-directory
cd~	changetohome-directory
cd ..	changetoparentdirectory
pwd	displaythepathofthecurrentdirectory

Copying(cp) and Move(mv)FilesandDirectories

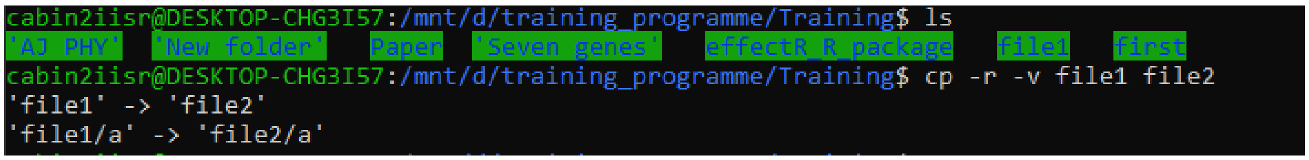
cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. **cp** command require at least two filenames in its arguments.

Important options(cp)

-r:	To copy directory
-i:	This option system first warns the user before overwriting the destination file.

Example (cp)

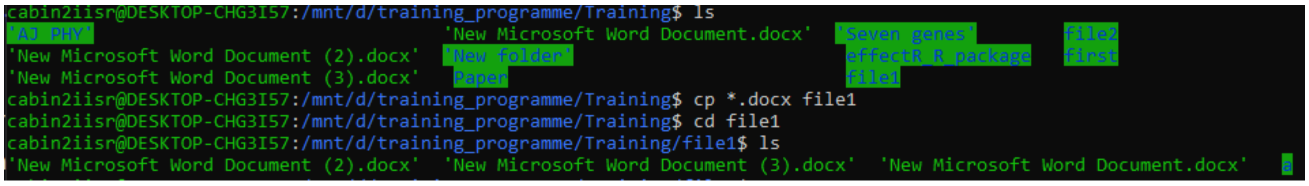
$ cp -r -v*file1file2*



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'   'New folder'   Paper   'Seven genes'   effectR_R_package   file1   first
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cp -r -v file1 file2
'file1' -> 'file2'
'file1/a' -> 'file2/a'
```

"cpfile1file2"isthecommandwhichmakesacopyof**file1**inthecurrentworkingdirectory and calls it**file2**.

$ cp *.txt file1



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'                                'New Microsoft Word Document.docx'  'Seven genes'        file2
'New Microsoft Word Document (2).docx'  'New folder'                         effectR_R_package   first
'New Microsoft Word Document (3).docx'   Paper                               file1
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cp *.docx file1
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cd file1
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/file1$ ls
'New Microsoft Word Document (2).docx'  'New Microsoft Word Document (3).docx'  'New Microsoft Word Document.docx'   a
```

The star wildcard represents anything i.e. all files and directories. Suppose we have many text document in a directory and wants to copy it another directory, it takes lots of time if we copy files 1 by 1 or command becomes too long if specify all these file names as the argument, but by using * wildcard it becomes simple.

mv command is used to move one or more files or directories from one place to another in a file system like UNIX. It has two distinct functions:

(i) It renames a file or folder.

(ii) It moves a group of files to a different directory

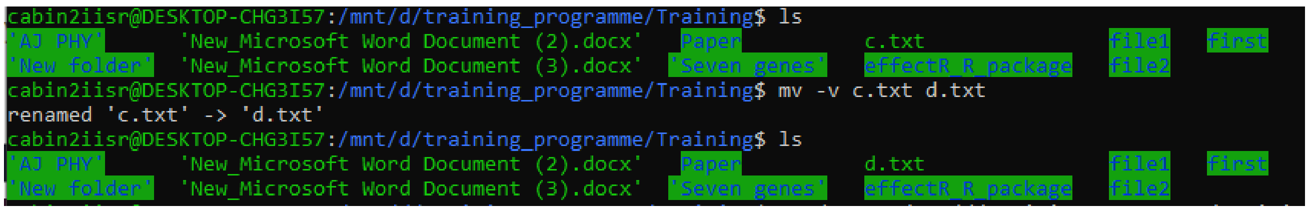
<u>Important options</u>(mv)

-r:	To copy directory
-i:	This option system first warns the user before overwriting the existing file.
-n:	It prevent an existing file from being overwritten.

<u>Example</u> (mv)

$ mv –v c.txt d.txt

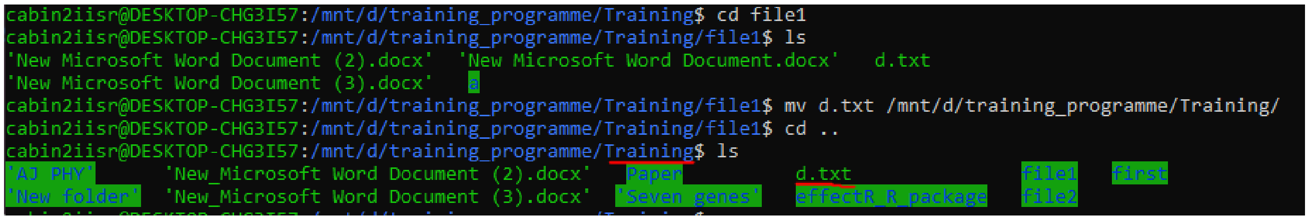
This command rename the *c.txt file* to *d.txt* file in same directory



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'      'New_Microsoft Word Document (2).docx'   Paper           c.txt              file1   first
'New folder'  'New_Microsoft Word Document (3).docx'  'Seven genes'   effectR_R_package  file2
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ mv -v c.txt d.txt
renamed 'c.txt' -> 'd.txt'
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'      'New_Microsoft Word Document (2).docx'   Paper           d.txt              file1   first
'New folder'  'New_Microsoft Word Document (3).docx'  'Seven genes'   effectR_R_package  file2
```

$ mv d.txt /mnt/d/training_programme/Training/

This command move d.txt from the location /mnt/d/training_programme/Training/file1 to the parent directory /mnt/d/training_programme/Training



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cd file1
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/file1$ ls
'New Microsoft Word Document (2).docx'  'New Microsoft Word Document.docx'   d.txt
'New Microsoft Word Document (3).docx'   a
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/file1$ mv d.txt /mnt/d/training_programme/Training/
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/file1$ cd ..
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ ls
'AJ PHY'       'New_Microsoft Word Document (2).docx'   Paper           d.txt              file1   first
'New folder'   'New_Microsoft Word Document (3).docx'  'Seven genes'    effectR_R_package  file2
```

➢ Display the contents of file on the screen

Concatenate (cat)

$ cat d.txt

Thecommand*cat* canbeusedtodisplaythecontentsof the d.txtfileonthescreen.

But ,thefileislongerthanthanthesizeofthewindow,soitscrollspastmakingit unreadable.

Less

$ less d.txt

Thecommand*less*writesthecontentsofafileontothescreenapageatatime.

Pressthespacebarifyouwanttoseeanotherpage,type*q*ifyouwanttoquitreading.Asyou can see, *less*isusedin preference to*cat*forlongfiles.

head

$ head d.txt

The head command will, by default, write the first ten lines of the input file to the standard

$ head -20 d.txt

With the -n option, we can let the head command output the first n lines instead of the default 10

tail

$ tail d.txt

The tail command will, by default, write the last ten lines of the input file to the standard

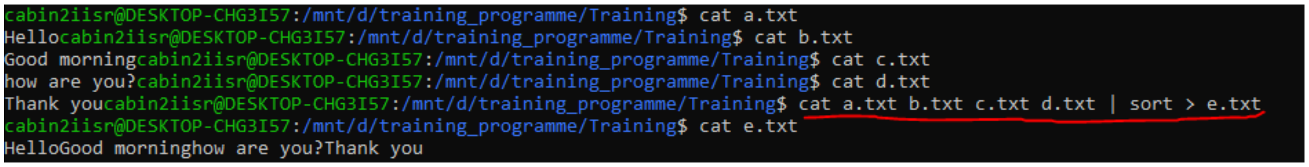
$ tail -20 d.txt

With the -n option, we can let the head command output the last n lines instead of the default 10

➢ Sorting Contents of Multiple Files in a Single File

$ cat a.txt b.txt c.txt d.txt | sort > e.txt

This will create a file e.txt and the output of the cat command is piped to sort and the result will be redirected to a newly created file.



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat a.txt
Hellocabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat b.txt
Good morningcabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat c.txt
how are you?cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat d.txt
Thank youcabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat a.txt b.txt c.txt d.txt | sort > e.txt
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ cat e.txt
HelloGood morninghow are you?Thank you
```

➢ Searchingthecontents ofa file

Simple searchingusing“less”

Using*less*,youcansearchthoughatextfilefor akeyword (pattern).For example,tosearchthrough**f.txt**forthe word 'contig',type

$ less f.txt

then,stillin*less*(i.e.don'tpress**q**toquit),typeaforwardslash(**/**)followedbythewordto searchfor, e.g.

/contig

Asyoucansee,*less*findsandhighlightsthekeyword.Type**n**tosearchforthenextoccurrenceof the word.

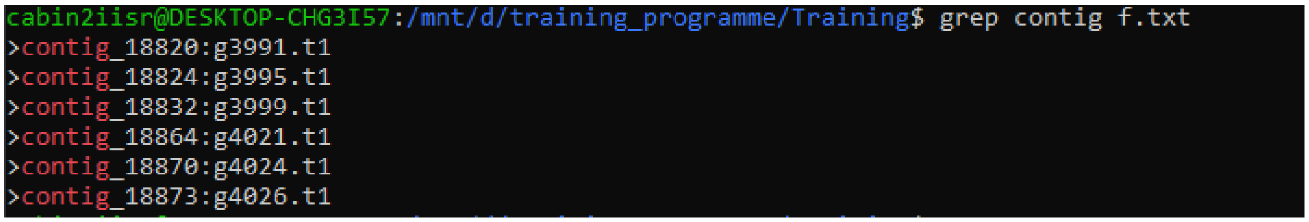
“grep”

<u>Important options</u>(grep)

-v: display those lines that do NOT match
-n: precede each matching line with the line number
-c: print only the total count of matched lines

*grep*isoneofmanystandardUNIXutilities.Itsearchesfilesforspecifiedwordsorpatterns.

$ grepcontig f.txt



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ grep contig f.txt
>contig_18820:g3991.t1
>contig_18824:g3995.t1
>contig_18832:g3999.t1
>contig_18864:g4021.t1
>contig_18870:g4024.t1
>contig_18873:g4026.t1
```

Asyoucansee,*grep*hasprintedouteachlinethatcontainsthewordcontig. Orhasit?

Trytyping

$ grepContig f.txt

The grep command is case sensitive; it distinguishes between Contig and contig. To ignore upper/lower case distinctions, use the -i option, i.e. type

wc(wordcount)

Ahandylittleutilityisthe*wc*command,shortforwordcount.Todoawordcounton **f.txt**,type

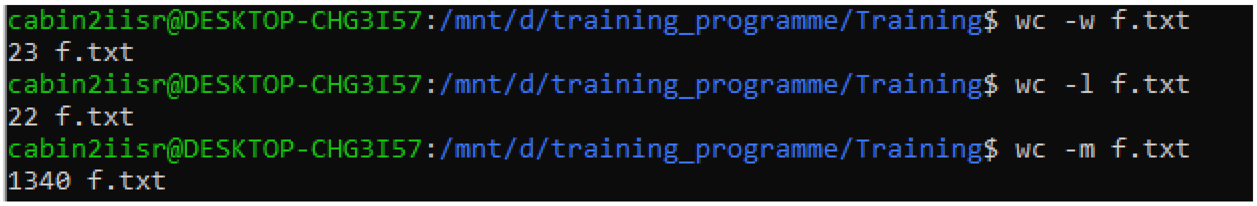
$ wc –w f.txt

Tofindouthowmanylines thefilehas,type

$ wc -l f.txt

Tofindouthowmanycharactersthefilehas,type

$ wc -m f.txt



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ wc -w f.txt
23 f.txt
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ wc -l f.txt
22 f.txt
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ wc -m f.txt
1340 f.txt
```

Gzip

gzip command compresses files. Each single file is compressed into a single file. If given a file as an argument, gzip compresses the file, adds a ".gz" suffix, and deletes the original file.

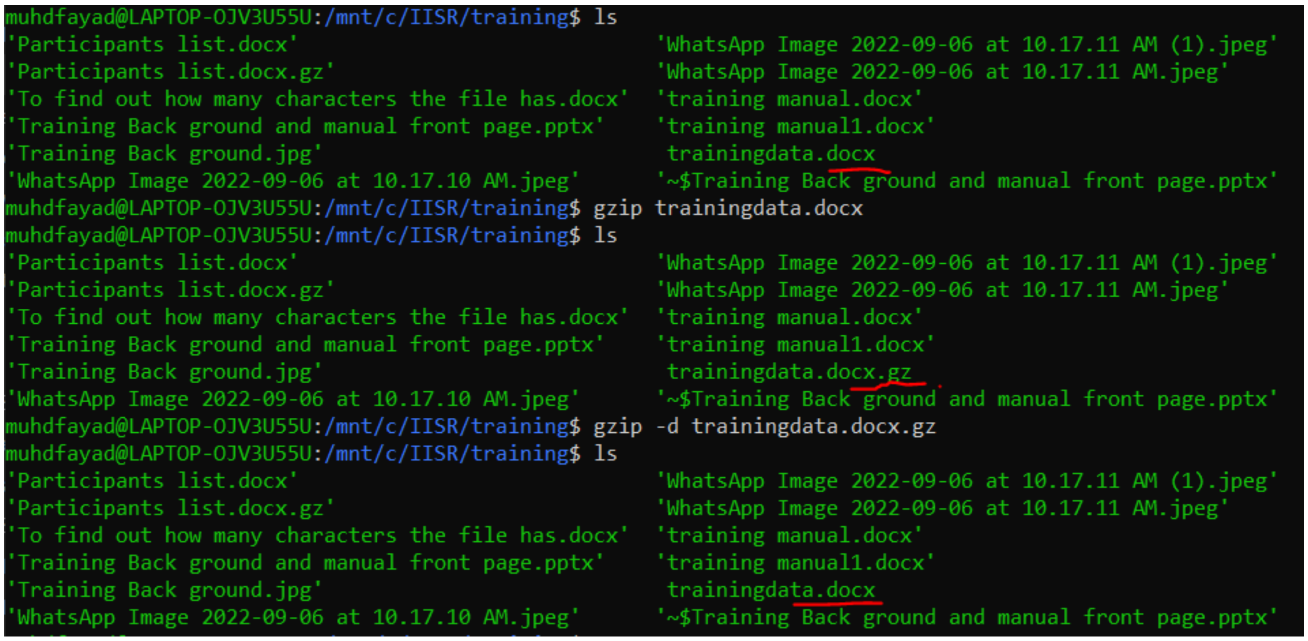
Important options(grep)

-f: This will forcefully compress a file even if there already exists a same file name.
-k: compress the file and keep the original file
-r: This will compress all the files present in the testfolder.
-[1-9]: To set the speed and compression level
-v: This option displays the name and percentage reduction for each file compressed or decompressed.
-d: This command will unzip the compressed file

$ gzip trainingdata.docx

This commands compress the trainingdata.docx file in to trainingdata.docx.gz

$ gzip -d trainingdata.docx.gz
It uncompress the file to trainingdata.docx

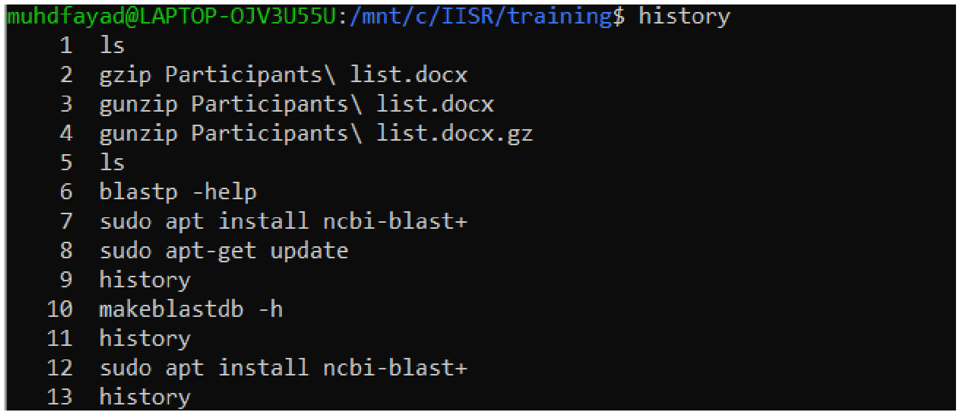


```
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ ls
'Participants list.docx'                              'WhatsApp Image 2022-09-06 at 10.17.11 AM (1).jpeg'
'Participants list.docx.gz'                           'WhatsApp Image 2022-09-06 at 10.17.11 AM.jpeg'
'To find out how many characters the file has.docx'   'training manual.docx'
'Training Back ground and manual front page.pptx'     'training manual1.docx'
'Training Back ground.jpg'                             trainingdata.docx
'WhatsApp Image 2022-09-06 at 10.17.10 AM.jpeg'       '~$Training Back ground and manual front page.pptx'
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ gzip trainingdata.docx
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ ls
'Participants list.docx'                              'WhatsApp Image 2022-09-06 at 10.17.11 AM (1).jpeg'
'Participants list.docx.gz'                           'WhatsApp Image 2022-09-06 at 10.17.11 AM.jpeg'
'To find out how many characters the file has.docx'   'training manual.docx'
'Training Back ground and manual front page.pptx'     'training manual1.docx'
'Training Back ground.jpg'                             trainingdata.docx.gz
'WhatsApp Image 2022-09-06 at 10.17.10 AM.jpeg'       '~$Training Back ground and manual front page.pptx'
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ gzip -d trainingdata.docx.gz
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ ls
'Participants list.docx'                              'WhatsApp Image 2022-09-06 at 10.17.11 AM (1).jpeg'
'Participants list.docx.gz'                           'WhatsApp Image 2022-09-06 at 10.17.11 AM.jpeg'
'To find out how many characters the file has.docx'   'training manual.docx'
'Training Back ground and manual front page.pptx'     'training manual1.docx'
'Training Back ground.jpg'                             trainingdata.docx
'WhatsApp Image 2022-09-06 at 10.17.10 AM.jpeg'       '~$Training Back ground and manual front page.pptx'
```

history

Theshellkeepsanorderedlistofallthecommandsthatyouhaveentered.Eachcomman disgiven anumberaccordingtothe orderitwasentered.

$ history



```
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ history
    1  ls
    2  gzip Participants\ list.docx
    3  gunzip Participants\ list.docx
    4  gunzip Participants\ list.docx.gz
    5  ls
    6  blastp -help
    7  sudo apt install ncbi-blast+
    8  sudo apt-get update
    9  history
   10  makeblastdb -h
   11  history
   12  sudo apt install ncbi-blast+
   13  history
```

Youcanusetheexclamationcharacter(!)torecallcommandseasily.

!!	#recalllastcommand
!-3	#recall thirdmostrecentcommand
!5	#recall5th commandinlist

```
!grep     #recall lastcommandstartingwithgrep
```

Youcanincreasethesizeofthehistorybufferbytyping

```
$ HISTSIZE=1000
```

Summary

cp *file1file2*	copyfile1 andcallitfile2
mv*file1file2*	moveorrenamefile1tofile2
rm *file*	removeafile
rmdir*directory*	removeadirectory
cat *file*	Displayorconcatenate afile
lessfile	displayafileapageata time
head*file*	displaythefirstfewlines ofafile
tail*file*	displaythe lastfewlines of afile
grep'keyword'*file*	searchafileforkeywords
wc*file*	countnumberoflines/words/charactersinfile

Standalone BLAST

The standalone BLAST server suite of programs was designed similar to the regular NCBI BLAST server and such command-line NCBI BLAST programs like "blastall", "blastpgp", "rpsblast" and "megablast". It incorporates most features, which exist in NCBI BLAST programs and should be relatively easy to use. These utilities run through DOS-like command windows and accept input through text-based command line switches. There is no graphic user interface.

The following steps discusses how to install NCBI-BLAST+
To install the NCBI-BLAST+ type

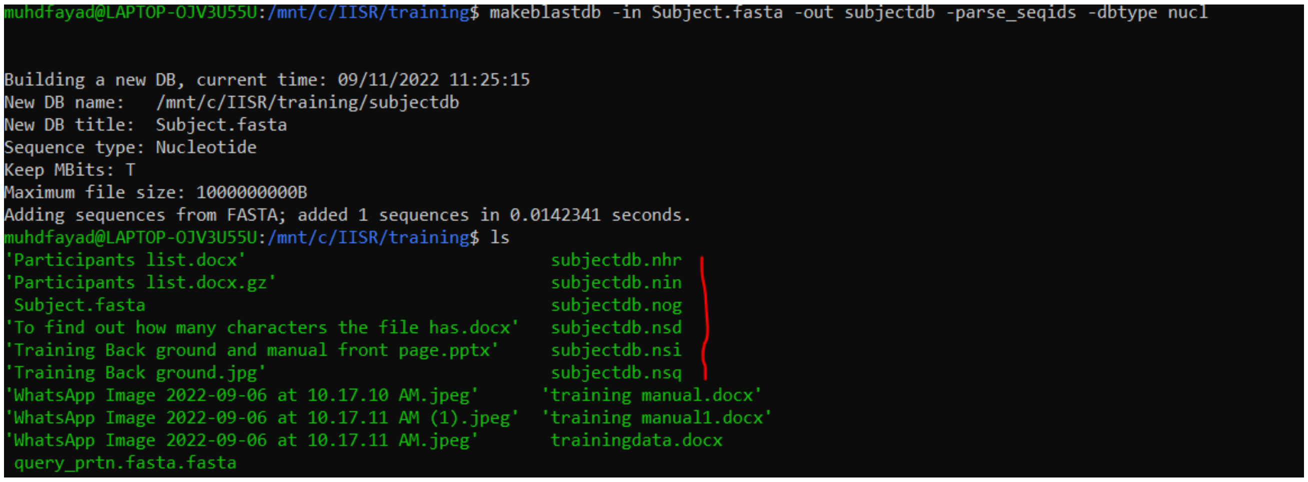
```
$ sudo apt-get -y install python ncbi-blast+
```

The programs in the BLAST+ suite can search for and against sequences in protein format and in nucleotide format. Depending on what type the query and subject sets are, different BLAST programs are used. Follow the steps to do **blastn** using Ubuntu wsl in Windows Operating System.

Creating a nucleotide database type

$ makeblastdb -in Subject.fasta -out subjectdb -parse_seqids -dbtypenucl

makeblastdb	:- Command
-inSubject.fasta	:- Input subject file
-outsubjectdb	:- Output name of the database need to create
-dbtypenucl	:- type of database need to create

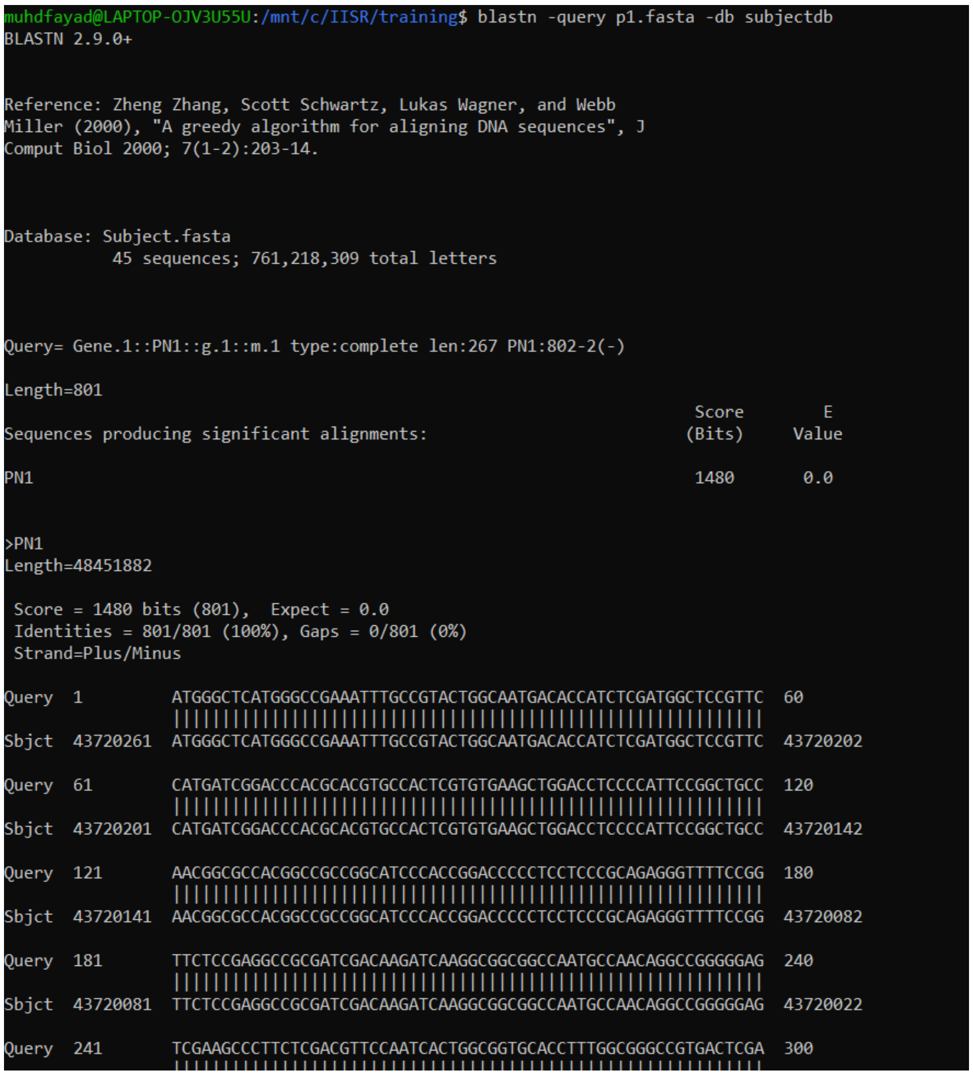


```
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ makeblastdb -in Subject.fasta -out subjectdb -parse_seqids -dbtype nucl


Building a new DB, current time: 09/11/2022 11:25:15
New DB name:   /mnt/c/IISR/training/subjectdb
New DB title:  Subject.fasta
Sequence type: Nucleotide
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 1 sequences in 0.0142341 seconds.
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ ls
'Participants list.docx'                              subjectdb.nhr
'Participants list.docx.gz'                           subjectdb.nin
 Subject.fasta                                        subjectdb.nog
'To find out how many characters the file has.docx'  subjectdb.nsd
'Training Back ground and manual front page.pptx'     subjectdb.nsi
'Training Back ground.jpg'                            subjectdb.nsq
'WhatsApp Image 2022-09-06 at 10.17.10 AM.jpeg'      'training manual.docx'
'WhatsApp Image 2022-09-06 at 10.17.11 AM (1).jpeg'  'training manual1.docx'
'WhatsApp Image 2022-09-06 at 10.17.11 AM.jpeg'       trainingdata.docx
 query_prtn.fasta.fasta
```

Blastn

To do blastn with query sequence (query.fasta) type

$ blastn -query p1.fasta -dbsubjectdb



```
muhdfayad@LAPTOP-OJV3U55U:/mnt/c/IISR/training$ blastn -query p1.fasta -db subjectdb
BLASTN 2.9.0+


Reference: Zheng Zhang, Scott Schwartz, Lukas Wagner, and Webb
Miller (2000), "A greedy algorithm for aligning DNA sequences", J
Comput Biol 2000; 7(1-2):203-14.



Database: Subject.fasta
           45 sequences; 761,218,309 total letters



Query= Gene.1::PN1::g.1::m.1 type:complete len:267 PN1:802-2(-)

Length=801
                                                                      Score        E
Sequences producing significant alignments:                          (Bits)     Value

PN1                                                                   1480       0.0


>PN1
Length=48451882

 Score = 1480 bits (801),  Expect = 0.0
 Identities = 801/801 (100%), Gaps = 0/801 (0%)
 Strand=Plus/Minus

Query  1         ATGGGCTCATGGGCCGAAATTTGCCGTACTGGCAATGACACCATCTCGATGGCTCCGTTC  60
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  43720261  ATGGGCTCATGGGCCGAAATTTGCCGTACTGGCAATGACACCATCTCGATGGCTCCGTTC  43720202

Query  61        CATGATCGGACCCACGCACGTGCCACTCGTGTGAAGCTGGACCTCCCCATTCCGGCTGCC  120
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  43720201  CATGATCGGACCCACGCACGTGCCACTCGTGTGAAGCTGGACCTCCCCATTCCGGCTGCC  43720142

Query  121       AACGGCGCCACGGCCGCCGGCATCCCACCGGACCCCCTCCTCCCGCAGAGGGTTTTCCGG  180
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  43720141  AACGGCGCCACGGCCGCCGGCATCCCACCGGACCCCCTCCTCCCGCAGAGGGTTTTCCGG  43720082

Query  181       TTCTCCGAGGCCGCGATCGACAAGATCAAGGCGGCGGCCAATGCCAACAGGCCGGGGGAG  240
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  43720081  TTCTCCGAGGCCGCGATCGACAAGATCAAGGCGGCGGCCAATGCCAACAGGCCGGGGGAG  43720022

Query  241       TCGAAGCCCTTCTCGACGTTCCAATCACTGGCGGTGCACCTTTGGCGGGCCGTGACTCGA  300
                 ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
```

$ blastn -query p1.fasta -dbsubjectdb -outfmt 7 -out result.txt

This code make a result.txt file having blast result in tabular format

(

-query <fasta file>

The name (or path) of the FASTA-formatted file to search for as query sequences.

-subject <fasta file>

The name (or path) of the FASTA-formatted file to search in as subject sequences.

-evalue<real number>

Only HSPs with E values smaller than this should be reported. For example: -evalue 0.001 or -evalue 1e-6.

-outfmt<integer>

How to format the output.

-outfmt<String>

alignment view options:

0 = Pairwise,

1 = Query-anchored showing identities,

2 = Query-anchored no identities,

3 = Flat query-anchored showing identities,

4 = Flat query-anchored no identities,

5 = BLAST XML,

6 = Tabular,

7 = Tabular with comment lines,

8 = Seqalign (Text ASN.1),

9 = Seqalign (Binary ASN.1),

10 = Comma-separated values,

11 = BLAST archive (ASN.1),

12 = Seqalign (JSON),

13 = Multiple-file BLAST JSON,

14 = Multiple-file BLAST XML2,

15 = Single-file BLAST JSON,

16 = Single-file BLAST XML2,

17 = Sequence Alignment/Map (SAM),

18 = Organism Report

)

Bioinformatics workflows

When working with high-throughput sequencing data, the raw reads you get off of the sequencer will need to pass through a number of different tools in order to generate your final desired output. The execution of this set of tools in a specified order is commonly referred to as a *workflow* or a *pipeline*.

1. Quality control - Assessing quality using FastQC

Make a new directory

```
$ mkdir -p workflow
```

Here we are using the -p option for mkdir. This option allows mkdir to create the new directory, even if one of the parent directories does not already exist. It also supresses errors if the directory already exists, without overwriting that directory.

```
$ cd ~/mnt/d/training_programme/Training/workflow
```

So we will enter into the new directory

Download the data

To download the data, run the commands below.

```
$ curl -O
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR258/004/SRR2589044/SRR2589044_1.fastq.gz
```

```
$ curl -O
ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR258/004/SRR2589044/SRR2589044_2.fastq.gz
```

Unzipping the file

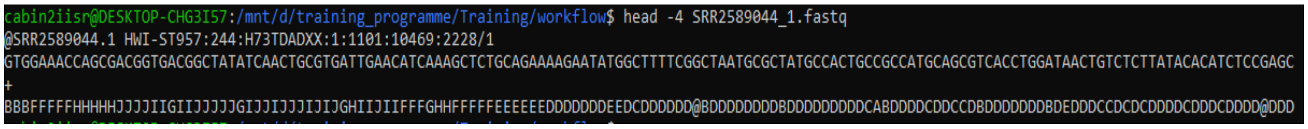
```
$ gunzipSRR2589044_1.fastq.gz
$ gunzipSRR2589044_2.fastq.gz
```

Checking the fastq file

We can view the first complete read in one of the files our dataset by using head to look at the first four lines.

```
$ head -4 SRR2589044_1.fastq
```

Although it looks complicated (and it is), we can understand the fastq format with a little decoding. Some rules about the format include



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ head -4 SRR2589044_1.fastq
@SRR2589044.1 HWI-ST957:244:H73TDADXX:1:1101:10469:2228/1
GTGGAAACCAGCGACGGTGACGGCTATATCAACTGCGTGATTGAACATCAAAGCTCTGCAGAAAAGAATATGGCTTTTCGGCTAATGCGCTATGCCACTGCCGCCATGCAGCGTCACCTGGATAACTGTCTCTTATACACATCTCCGAGC
+
BBBFFFFFHHHHHJJJJIIGIIJJJJJGIJJIJJJIJIJGHIIJIIFFFGHHFFFFFEEEEEEDDDDDDDEEDCDDDDDD@BDDDDDDDDBDDDDDDDDDCABDDDDCDDCCDBDDDDDDDBDEDDDCCDCDCDDDDCDDDCDDDD@DDD
```

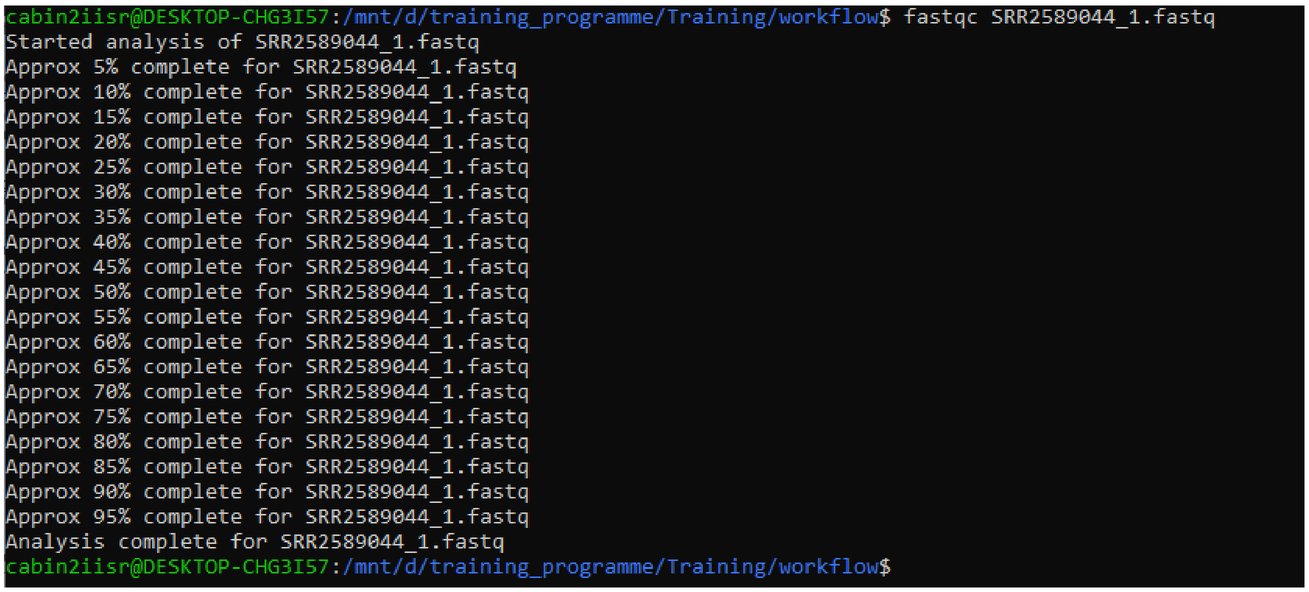
Line	Description
1	Always begins with ‘@’ and then information about the read
2	The actual DNA sequence
3	Always begins with a ‘+’ and sometimes the same info in line 1
4	Has a string of characters which represent the quality scores; must have same number of characters as line 2

Installing fastqc

```
$ sudo apt update
$ sudo apt install fastqc
```

Run fastqc

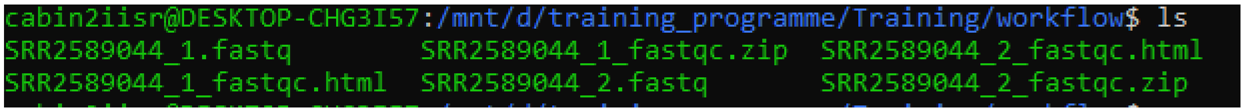
```
$ fastqcSRR2589044_1.fastq
$ fastqcSRR2589044_2.fastq
```



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ fastqc SRR2589044_1.fastq
Started analysis of SRR2589044_1.fastq
Approx 5% complete for SRR2589044_1.fastq
Approx 10% complete for SRR2589044_1.fastq
Approx 15% complete for SRR2589044_1.fastq
Approx 20% complete for SRR2589044_1.fastq
Approx 25% complete for SRR2589044_1.fastq
Approx 30% complete for SRR2589044_1.fastq
Approx 35% complete for SRR2589044_1.fastq
Approx 40% complete for SRR2589044_1.fastq
Approx 45% complete for SRR2589044_1.fastq
Approx 50% complete for SRR2589044_1.fastq
Approx 55% complete for SRR2589044_1.fastq
Approx 60% complete for SRR2589044_1.fastq
Approx 65% complete for SRR2589044_1.fastq
Approx 70% complete for SRR2589044_1.fastq
Approx 75% complete for SRR2589044_1.fastq
Approx 80% complete for SRR2589044_1.fastq
Approx 85% complete for SRR2589044_1.fastq
Approx 90% complete for SRR2589044_1.fastq
Approx 95% complete for SRR2589044_1.fastq
Analysis complete for SRR2589044_1.fastq
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$
```

It should take some time for FastQC to run FASTQ files. When the analysis completes, your prompt will return.

The FastQC program has created several new files within our directory.



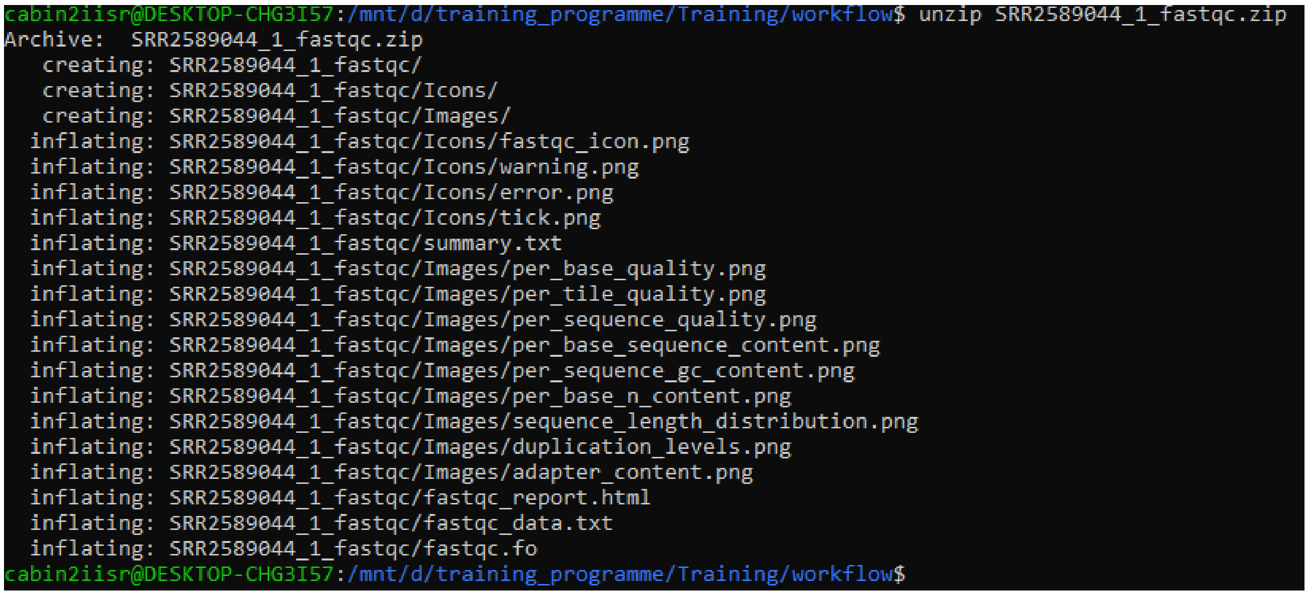
```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ ls
SRR2589044_1.fastq        SRR2589044_1_fastqc.zip  SRR2589044_2_fastqc.html
SRR2589044_1_fastqc.html  SRR2589044_2.fastq       SRR2589044_2_fastqc.zip
```

For each input FASTQ file, FastQC has created a .zip file and a.html file. The .zip file extension indicates that this is actually a compressed set of multiple output files. The .html file is a stable webpage displaying the summary report for each of our samples.

Our .zip files are compressed files. They each contain multiple different types of output files for a single input FASTQ file. To view the contents of a .zip file, we can use the program unzip to decompress these files. Let’s try

$ unzip SRR2589044_1_fastqc.zip

$ unzip SRR2589044_2_fastqc.zip



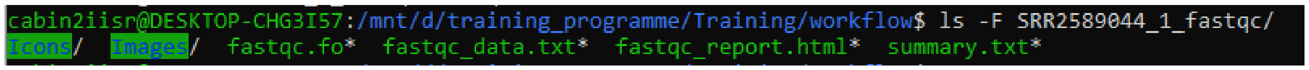
```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ unzip SRR2589044_1_fastqc.zip
Archive:  SRR2589044_1_fastqc.zip
   creating: SRR2589044_1_fastqc/
   creating: SRR2589044_1_fastqc/Icons/
   creating: SRR2589044_1_fastqc/Images/
  inflating: SRR2589044_1_fastqc/Icons/fastqc_icon.png
  inflating: SRR2589044_1_fastqc/Icons/warning.png
  inflating: SRR2589044_1_fastqc/Icons/error.png
  inflating: SRR2589044_1_fastqc/Icons/tick.png
  inflating: SRR2589044_1_fastqc/summary.txt
  inflating: SRR2589044_1_fastqc/Images/per_base_quality.png
  inflating: SRR2589044_1_fastqc/Images/per_tile_quality.png
  inflating: SRR2589044_1_fastqc/Images/per_sequence_quality.png
  inflating: SRR2589044_1_fastqc/Images/per_base_sequence_content.png
  inflating: SRR2589044_1_fastqc/Images/per_sequence_gc_content.png
  inflating: SRR2589044_1_fastqc/Images/per_base_n_content.png
  inflating: SRR2589044_1_fastqc/Images/sequence_length_distribution.png
  inflating: SRR2589044_1_fastqc/Images/duplication_levels.png
  inflating: SRR2589044_1_fastqc/Images/adapter_content.png
  inflating: SRR2589044_1_fastqc/fastqc_report.html
  inflating: SRR2589044_1_fastqc/fastqc_data.txt
  inflating: SRR2589044_1_fastqc/fastqc.fo
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$
```

The unzip program is decompressing the .zip files and creating a new directory (with subdirectories) for each of our samples, to store all of the different output that is produced by FastQC. There are a lot of files here. The one we are going to focus on is the summary.txt file

Let’s see what files are present within one of these output directories.

$ ls -F SRR2589044_1_fastqc/

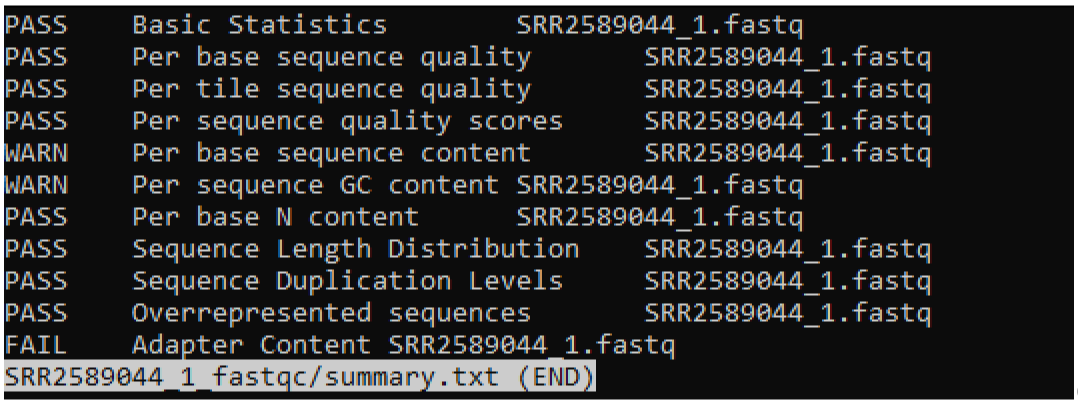
$ ls -F SRR2589044_2_fastqc/



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ ls -F SRR2589044_1_fastqc/
Icons/  Images/  fastqc.fo*  fastqc_data.txt*  fastqc_report.html*  summary.txt*
```

Use less to preview the summary.txt file for this sample.

$ lessSRR2589044_1_fastqc/summary.txt



```
PASS    Basic Statistics        SRR2589044_1.fastq
PASS    Per base sequence quality       SRR2589044_1.fastq
PASS    Per tile sequence quality       SRR2589044_1.fastq
PASS    Per sequence quality scores     SRR2589044_1.fastq
WARN    Per base sequence content       SRR2589044_1.fastq
WARN    Per sequence GC content SRR2589044_1.fastq
PASS    Per base N content      SRR2589044_1.fastq
PASS    Sequence Length Distribution    SRR2589044_1.fastq
PASS    Sequence Duplication Levels     SRR2589044_1.fastq
PASS    Overrepresented sequences       SRR2589044_1.fastq
FAIL    Adapter Content SRR2589044_1.fastq
SRR2589044_1_fastqc/summary.txt (END)
```

q

Documenting the work

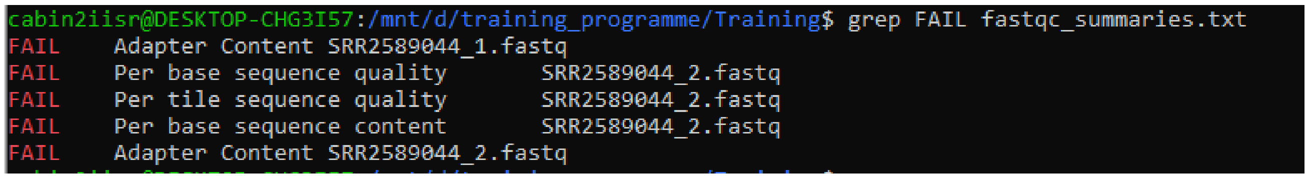
We can make a record of the results we obtained for all our samplesby concatenating all of our summary.txt files into a single file using the cat command. We will call this fastqc_summaries.txt

$ cat */summary.txt >/mnt/d/training_programme/Training/fastqc_summaries.txt

We can get the list of all failed tests using grep

$ cd/mnt/d/training_programme/Training

$ grep FAIL fastqc_summaries.txt



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training$ grep FAIL fastqc_summaries.txt
FAIL    Adapter Content SRR2589044_1.fastq
FAIL    Per base sequence quality       SRR2589044_2.fastq
FAIL    Per tile sequence quality       SRR2589044_2.fastq
FAIL    Per base sequence content       SRR2589044_2.fastq
FAIL    Adapter Content SRR2589044_2.fastq
```

2. Cutadapt

To trim a 3' adapter from the untrimmed fastq file

To install cutadapt type

$ sudo apt install cutadapt

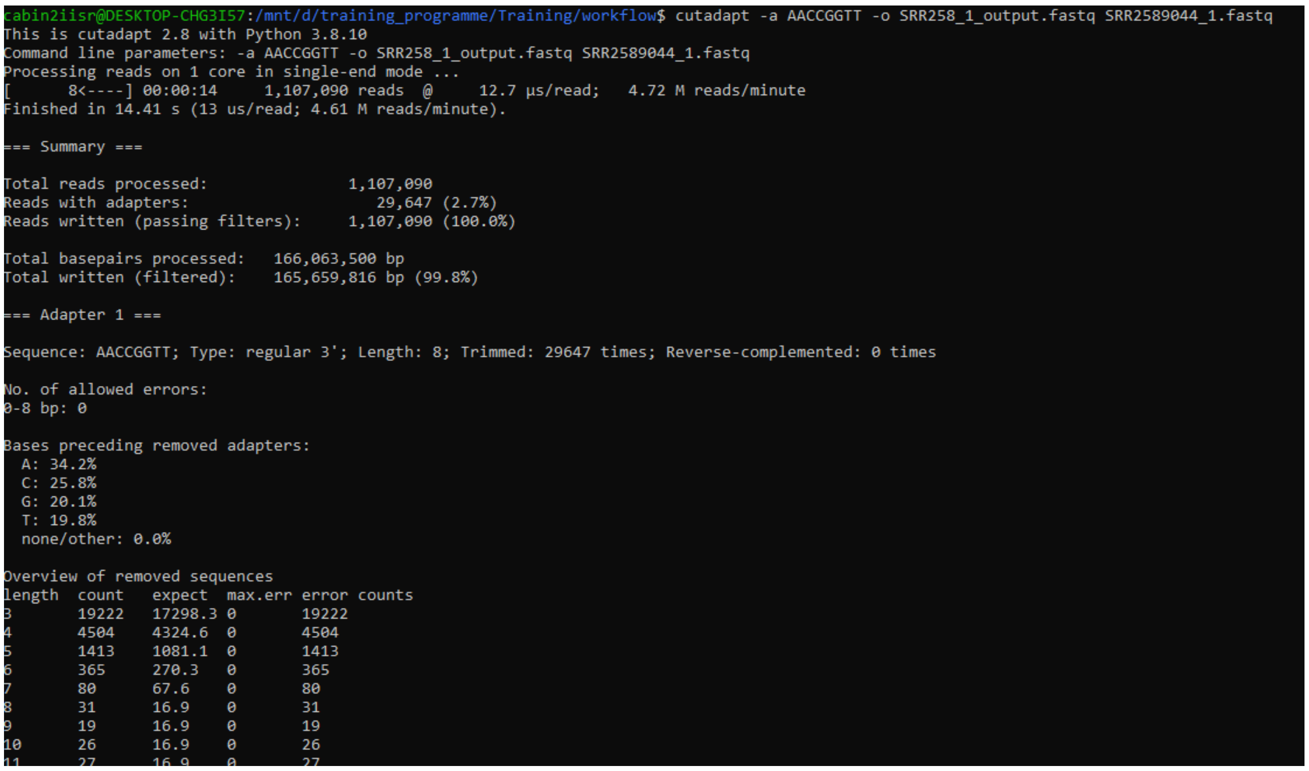
To run cutadapt, move to the file containing directory

```
$ cd /mnt/d/training_programme/Training/workflow
```

The basic command-line for Cutadapt is
cutadapt -a AACCGGTT -o output.fastqinput.fastq

The sequence of the adapter is given with the -a option. You need to replace AACCGGTT with the correct adapter sequence. Reads are read from the input file input.fastq and are written to the output file output.fastq

```
$ cutadapt -a AACCGGTT -o SRR258_1_output.fastq
SRR2589044_1.fastq
```



```
cabin2iisr@DESKTOP-CHG3I57:/mnt/d/training_programme/Training/workflow$ cutadapt -a AACCGGTT -o SRR258_1_output.fastq SRR2589044_1.fastq
This is cutadapt 2.8 with Python 3.8.10
Command line parameters: -a AACCGGTT -o SRR258_1_output.fastq SRR2589044_1.fastq
Processing reads on 1 core in single-end mode ...
[      8<----] 00:00:14     1,107,090 reads  @     12.7 µs/read;   4.72 M reads/minute
Finished in 14.41 s (13 us/read; 4.61 M reads/minute).

=== Summary ===

Total reads processed:               1,107,090
Reads with adapters:                    29,647 (2.7%)
Reads written (passing filters):     1,107,090 (100.0%)

Total basepairs processed:   166,063,500 bp
Total written (filtered):    165,659,816 bp (99.8%)

=== Adapter 1 ===

Sequence: AACCGGTT; Type: regular 3'; Length: 8; Trimmed: 29647 times; Reverse-complemented: 0 times

No. of allowed errors:
0-8 bp: 0

Bases preceding removed adapters:
  A: 34.2%
  C: 25.8%
  G: 20.1%
  T: 19.8%
  none/other: 0.0%

Overview of removed sequences
length  count   expect  max.err error counts
3       19222   17298.3 0       19222
4       4504    4324.6  0       4504
5       1413    1081.1  0       1413
6       365     270.3   0       365
7       80      67.6    0       80
8       31      16.9    0       31
9       19      16.9    0       19
10      26      16.9    0       26
11      27      16.9    0       27
```

Topic 6: Introduction to Python

Mr. Subeesh A
Scientist (Computer Applications),
ICAR- Central Institute of Agricultural Engineering, Bhopal, Madhya pradesh
Email: subeesh.a@icar.gov.in

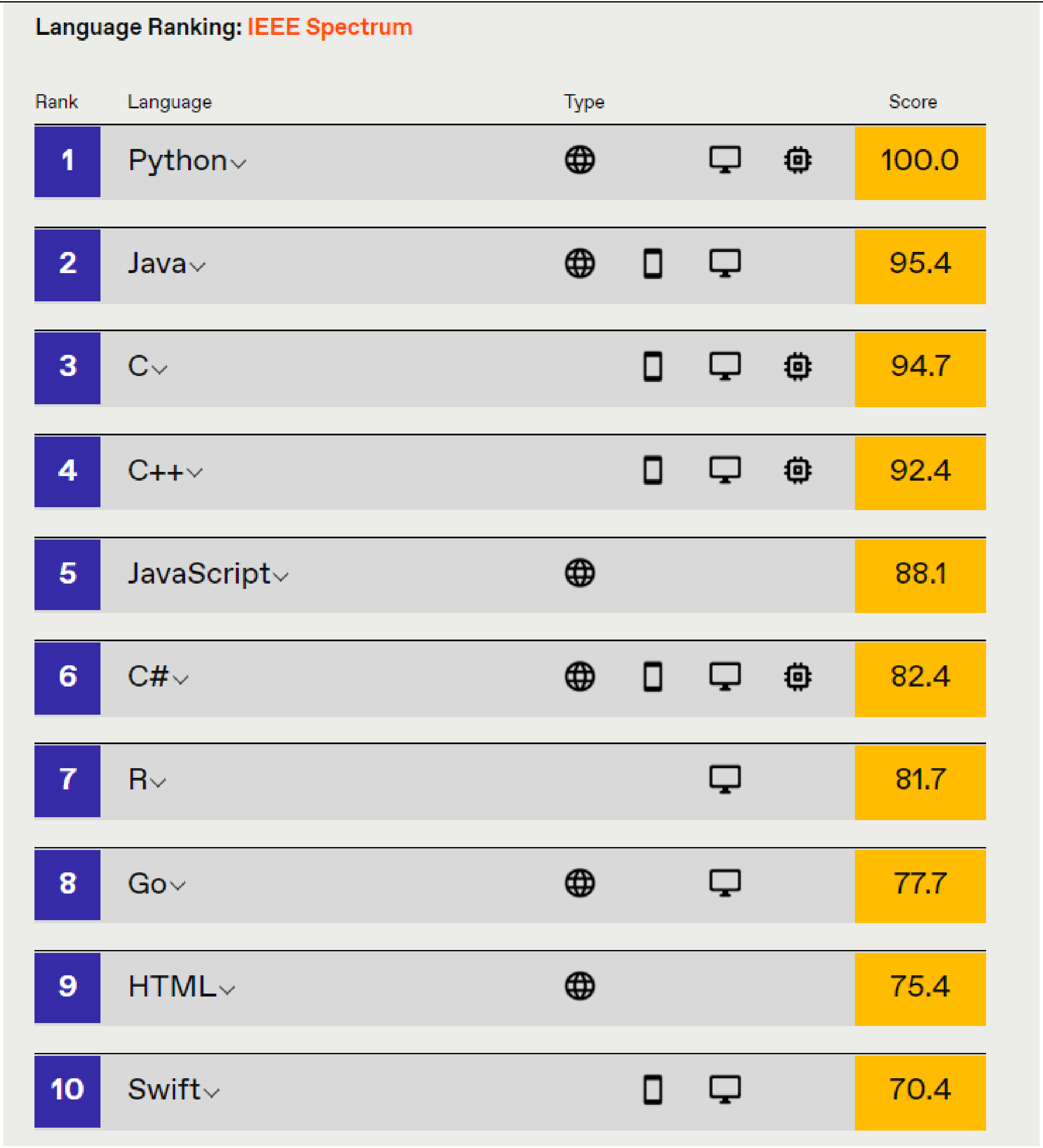
Overview

Python is a widely used high-level object oriented programming language created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It is also called general-purpose programming language as it is used in almost every domain we can think of such as:

- Web Development
- Software Development
- Game Development
- Artificial Intelligence and Machine learning
- Data Analytics, etc.

The main reasons for the wide adoption of python are very simple to understand, scalable because of which the speed of development is so fast. Python has simpler syntax similar to the English language and also the syntax allows developers to write programs with fewer lines of code than some other programming language. Since it is open-source there are many libraries available that make developers' jobs easy ultimately results in high productivity. This means that prototyping can be very quick. IEE spectrum has ranked python as #1 popular language of 2021.

The most recent major version of Python is Python 3, which we shall be using in this training manual.



Language Ranking: IEEE Spectrum

Rank	Language	Type				Score
1	Python	🌐		🖥	⚙	100.0
2	Java	🌐	📱	🖥		95.4
3	C		📱	🖥	⚙	94.7
4	C++		📱	🖥	⚙	92.4
5	JavaScript	🌐				88.1
6	C#	🌐	📱	🖥	⚙	82.4
7	R			🖥		81.7
8	Go	🌐		🖥		77.7
9	HTML	🌐				75.4
10	Swift		📱	🖥		70.4

Figure 1 : IEE spectrum ranking of languages 2021 (https://spectrum.ieee.org/top-programming-languages/)

Notes:

- Python runs on an interpreter system, meaning that code can be executed as soon as it is written.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.
- This training manual uses google Colab to execute python commands. All the codes are written in Python 3.7 version. Python programs can be written in a text editor as well. It is also possible to write Python in an Integrated Development Environment, such as Spyder, Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Beginning with Python Programming

1. Python print statements

The print() function in Python is used to print a specified message on the screen. The print command in Python prints strings or objects which are converted to a string while printing on a screen.

```
>>print ("Hello python")
```

2. Python Indentations

Indentation refers to the spaces at the beginning of a code line. The indentation in Python is very important and itindicate a block of code.

Eg:

```
if 6 > 2:
print("Six is greater than two!")
```

3. Python Comments

Comments can be used to explain a python code and it makes the code more readable.

Comments starts with a #, and Python will ignore them during the execution.

E.g:

```
#This is a comment
print("Hello, World!")
```

4. Python Variables

Variables are containers for storing data values. Python has no command for declaring a variable. A variable is created the moment you first assign a value to it.

Eg:

```
x = 6
y = "Sam"
print(x)
print(y)
```

When we assign any value to the variable, that variable is declared automatically.

The equal (=) operator is used to assign value to a variable.

E.g:

```
data = "Welcome”

print(data)
```

Assigning multiple values to multiple variables can be performed using the below code.

```
a, b, c = 5, 4.5, "Testdata"


print (a)

print (b)

print (c)
```

5. Identifiers

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language.

Examples of valid identifiers: test, a65, _num, n_9data, etc.

Examples of invalid identifiers: 1a, n%4, n 9, etc.

6. Keywords

Keywords are the reserved words in Python and we cannot use a keyword as a variable name, function name or any other identifier. They are used to define the syntax and structure of the Python language.

E.g : if, break, import, else, for, is, etc.

7. Data types

Variables can hold values, and every value has a data-type. Python is a dynamically typed language; hence we do not need to define the type of the variable while declaring it. The interpreter implicitly binds the value with its type.

Python enables us to check the type of the variable used in the program. Python provides us the **type()** function, which returns the type of the variable passed.

```
a=10

b="Hi Python"

c = 10.5

print(type(a))     # Outputs <type 'int'>

print(type(b))     # Outputs <type 'str'>

print(type(c))     # Outputs <type 'float'>
```

Some of the standard datatypes used in python are given below.

7.1. Python Numbers

Integers, floating point numbers and complex numbers fall under Python numbers category. They are defined as int, float and complex classes in Python.

```
a = 7       # Integer type
a= 2.2      # Float type
a= 1+3j  # Complex type
```

7.2. Python List

List is an ordered sequence of elements. It is one of the most used datatype in Python and is very flexible. All the items in a list do not need to be of the same type. A python list is declared with elements separated by commas are enclosed within brackets [ ].

Eg:

a = [1, 4.3,‘data’]

Slicing operator [ ] to extract an item or a range of items from a list. The index starts from 0 in Python.

7.3. Python Tuple

Tuple is an ordered sequence of items same as a list. The only difference is that tuples are immutable. Tuples once created cannot be modified and it is faster than lists.

It is defined within parentheses () where items are separated by commas.

E.g:

```
test = (5,’data, 1+5j)

print("test[1] = ", test[1])        #outputs 5

t[1] =  56                                  #Generates error
```

7.4. Python Strings

String is sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ''' or """.

Eg:

```
s = "This is a string"

s = '''A multiline
```

string'''

7.5. Python Set

Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

Eg:

a = {5,2,3,1,4}

7.6. Python Dictionary

Dictionaries are used to store data values in key:value pairs. It is a collection of changeable items and do not allow duplicates. Dictionaries are written with curly brackets, and have keys and values:

Eg:

```
Sample_dict= {
  "name": "James",
  "Rollno": "123",
  "year": 2001
}
```

Python Flow Control

7.7. if...else Statement

The if...else statement in python is used for decision making.The if statement is used to test a specific condition. If the condition is true, a block of code (if-block) will be executed.. If the condition provided in the if statement is false, then the else statement will be executed.

Eg:

```
if test expression:

    Body of if

else:

    Body of else
```

7.8. For loop

The for loop in Python is used to iterate over a sequence (list, dictionary, tuple, string) or other iterable objects.

For loop has the following syntax in python.

fori in sequence:

loop body

Eg:

```
names = ["John", "Sam", "James"]
for x in names:
  print(x)
```

7.9. While loop

With the while loop we can execute a set of statements as long as a condition is true.we need to define an indexing variable and change it in each iteration, otherwise the loop may continue forever.

```
whiletest_expression:

    Body of while
```

Eg:

```
i = 1
while i< 5:
  print(i)                           # Prints the numbers 1 to 4
  i += 1
```

8. Python Functions

A function is a block of code which only runs when it is called. Functions help in breaking the complex program into smaller chunks. Functions make the code more readable, less repetitive, reusable and highly manageable. In Python a function is defined using the def keyword. To call a function, use the function name followed by parenthesis. Information can be passed into functions as arguments and values can be returned. Arguments are specified after the function name, inside the parentheses, separated with a comma.

Eg 1: Function without arguments

```
def my_function():
  print("Hello, this is a  function")

my_function()
```

Eg. 2 :Function with arguments

```
def square( num ):

    return num**2

object_ = square(3)   # Returns square of the argument passed
```

Python for Data Analysis

1. Numpy

NumPy is an array processing package in Python that provides a high-performance multidimensional array object and tools for working with it. It is the fundamental package for scientific computing with Python.

2. Pandas

Pandas is referred as Python Data Analysis Library. It is another open source Python library for availing high-performance data structures and analysis tools. It is developed over the Numpy package. It contains DataFrame as its main data structure.With DataFrame you can store and manage data from tables by performing manipulation over rows and columns. Pandas can handle multiple data format such as excel, csv, SQL, HDFS, etc.

3. Matplotlib

Matplotlib is a python library used to create graphs and plots by using python scripts. It has a module named pyplot which can ease the plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc.

4. Scipy

Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc

5. Scikit-learn

Scikit-learn is one of the most popular python libraries for implementing machine learning algorithms. It is built on top of two basic Python libraries, viz., NumPy and SciPy. Scikit-learn supports most of the supervised and unsupervised learning algorithms.

6. Keras

Keras is one of the most powerful Python libraries which allow high-level neural networks APIs for integration.Keras was created for reducing challenges faced in complex researches allowing them to compute faster. Due to its modular nature, one can use varieties of modules from neural layers, optimizers, activation functions etc.., for developing a new model.

7. TensorFlow

TensorFlow is a very popular open-source library for high performance numerical computation developed by the Google Brain team. It is a framework that involves defining and running computations involving tensors. It can train and run deep neural networks that can be used to develop several AI applications and is widely used in the field of deep learning research and application.

8. Pytorch

Pytorch is a Python-based scientific computing package that uses the power of graphics processing unit. It specializes in tensor computations, automatic differentiation, and GPU acceleration. For those reasons, PyTorch is one of the most popular deep learning libraries, competing with both Keras and TensorFlow. The framework is built to speed up the process between research prototyping and deployment.

Topic 6: Introduction to Galaxy

Dr. Prashanth N Suravajhala
Principal scientist, School of Biotechnology,
Amrita Vishwa Vidyapeetham, Amritapuri, Kollam, Kerala
Email: prash@am.amrita.edu

Pipelines for transcriptomics

```
#Indexing already done using bowtie2, BWA and samtools: /home/prash/Data/hg38

#All scripts and commands are to be run from Expipe

#fastqc already one for all samples. Pl check the folder

#bowtie2 -x /home/ngs/Data/hg38/hg38 -1 AB_R1_cutadapt.fastq.gz -2 AB_R2_cutadapt.fastq.gz -S AB.sam

#samtools view AB.sam -o AB.bam

#samtools sort AB.bam >AB.sorted.bam

#samtools index AB.sorted.bam AB.sorted.bam.bai &

#samtools merge AB.merged.bam AB.sorted.*

samtools mpileup AB.sorted.bam > AB.mpileup.bam

varscan mpileup2snp AB.mpileup.bam > AB.mpileup.snps &

varscan mpileup2indel AB.mpileup.bam > AB.mpileup.indels

varscan filter AB.mpileup.snps >AB.mpileup.snps.filter

varscan readcounts AB.mpileup.bam >AB.mpileup.readcounts

samtools mpileup -uf /home/ngs/Data/hg38/hg38.fa AB.sorted.bam | bcftools view - >AB.raw.bcf &

#samtools calmd -Abr AB.sorted.bam /home/ngs/Data/hg38/hg38.fa > AB.baq.bam

#bcftools view AB.raw.bcf >AB.vcf
```

#Fastqc, trimming the raw reads and then checking the files must be done aprior

#/home/ngs/Tools/hisat2/./hisat2 -x /home/ngs/Data/hg38/hg38 -1 /home/test/datasets/Human/control_R1.fastq -2 /home/test/datasets/Human/control_R2.fastq -S control.sam &

#/home/ngs/Tools/hisat2/./hisat2 -x /home/ngs/Data/hg38/hg38 -1 /home/test/datasets/Human/test_R1.fastq -2 /home/test/datasets/Human/test_R2.fastq -S test.sam

#samtools view control.sam -o control.bam

#samtools view test.sam -o test.bam

#samtools sort control.bam -o control.sorted.bam

#samtools sort test.bam -o test.sorted.bam

#mkdir control

#mv control.* control/

#mkdir test

#mv test.* test/

#cd control

# running the cufflinks for the control in the control folder

#/home/ngs/Tools/cufflinks/./cufflinks control.sorted.bam &

#cd ..

#cd test

# running the cufflinks for the test in the test folder

#/home/ngs/Tools/cufflinks/./cufflinks test.sorted.bam &

```
#cd ..

#mkdir control_test

#cd control_test

# running the cuffdiff for the control transcripts and comparing it with the test

/home/ngs/Tools/cufflinks/./cuffdiff                    ../control/transcripts.gtf
../control/control.sorted.bam ../test/test.sorted.bam

bcftools filter -i 'MIN(INFO/DP)>20' AB.raw.bcf > AB_output_20.vcf &
```

Published workflows for exome analysis

https://usegalaxy.org/u/jeremy/w/exome-analysis