भाकृ अनुप
ICAR

ONLINE WORKSHOP ON

R for BIOLOGISTS

6 - 8 October 2021

Workshop manual

Lecture notes

iisr

Organized by

Bioinformatics Centre

ICAR-Indian Institute of Spices Research
Marikunnu, Kozhikode, Kerala

ONLINE WORKSHOP

ON

R FOR BIOLOGISTS

October 6-8, 2021

WORKSHOP MANUAL

(Lecture Notes)

Organized by

Bioinformatics Centre

ICAR- Indian Institute of Spices Research

Kozhikode, Kerala.

Published by

Dr. J Rema
Director, ICAR- Indian Institute of Spices Research

Organized by:

Bioinformatics Centre
ICAR- Indian Institute of Spices Research

Co-ordinators:

Ms. Sona Charles
Dr. Divya P Syamaladevi

Compiled & Edited by:

Ms. Sona Charles, Scientist, ICAR-IISR
Dr. Divya P Syamaladevi, Sr. Scientist, ICAR-IISR
Dr. Santhosh J Eapen, Principal Scientist & Head (Crop Protection) ICAR-IISR

ONLINE WORKSHOP ON R FOR BIOLOGISTS
(October 6-8, 2021)

Program Schedule

Day 1: 06-10-2021		
10:00 AM	Welcome Address	Dr. Divya P Syamaladevi, Senior Scientist (Agrl. Biotechnology), ICAR-Indian Institute of Spices Research
10:10 AM	Introductory Remarks and Release of Training Manual	Dr. J Rema, Director, ICAR-Indian Institute of Spices Research
10:20 AM	Felicitations	Dr. Santhosh J Eapen, Head, Crop Protection, 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-evaluation and photo session		
10:45 AM	Biology: From Digital to Synthetic (Inaugural Lecture)	Dr. Santhosh J Eapen
11:45 AM	Introduction to R (Lecture)	Ms. Sona Charles
12:30 PM	Setting up the computer (Hands-on)	Dr. Divya P Syamaladevi
2:00 PM	Introduction to R (Hands- on)	Ms. Sona Charles
3:30 PM	Introduction to R coding: control statements (Lecture and Hands-on)	Dr. Divya P Syamaladevi
Day 2: 07-10-2021		
10:00 AM	Basics of Statistics	Dr Sreekumar J, Principal Scientist (Agricultural Statistics) ICAR-Central Tuber Crops Research Institute
2:00 PM	Statistics using R (Hands-on)	
Day 3: 08.10.2021		
10:00 AM	Introduction to data visualization	Ms. Sona Charles
11:00 AM	Basics of ggplot (Hands-on)	Ms. Sona Charles
2:00 PM	Case studies in Life Science (Hands-on)	Ms. Sona Charles
Post-evaluation		
Concluding Session		
4:00 PM	Welcome	Dr. Divya P Syamaladevi
4.05 PM	Feedback	Participants
4:10 PM	Concluding Remarks	Dr. Santhosh J Eapen
4:20 PM	Vote of Thanks	Ms. Sona Charles

CONTENTS

1. Biology from digital to synthetic

Dr. Santhosh J Eapen
ICAR-Indian Institute of Spices Research
Email: Santhosh.Eapen@icar.gov.in

The field of Bioinformatics emerged in 1980s as a sub-discipline of biology and computer science concerned with the acquisition, storage, analysis, and dissemination of biological data, most often DNA and amino acid. Advancements in sequencing technologies mainly Next Generation Sequencing has resulted in huge escalation in the number of genomes from few to many thousands during the past 25 years (Figure 1). Such cumulative Genome Sequencing and the corresponding explosion of DNA sequencing data has necessitated computer databases that feature rapid assimilation, usable formats and algorithm software programs for efficient management of biological data. A fundamental activity in Bioinformatics is sequence analysis of DNA and proteins with the help of various programs and databases. Bioinformatics uses computer programs for a variety of applications, including determining gene and protein functions, establishing evolutionary relationships, and predicting the three-dimensional shapes of proteins.

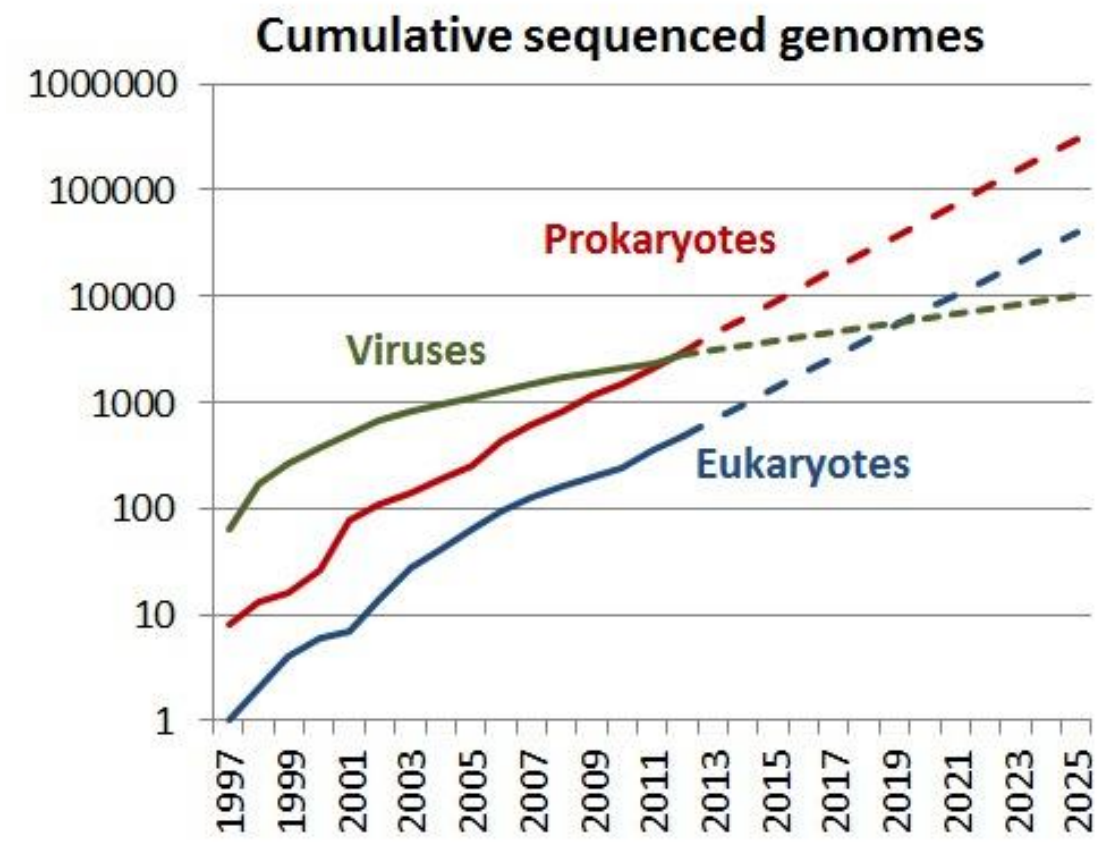


Figure 1

Because of the diverse nature of emerging data, no single comprehensive database exists for accessing all this information. However, a growing number of databases that contain helpful information for clinicians and researchers are available. Information provided by most of these databases is free of charge to academics, although some sites require subscription and industrial users pay a license fee for particular sites. Examples range from sites providing comprehensive descriptions of clinical disorders, listing disease susceptibility genetic mutations and polymorphisms, to those enabling a search for disease genes given a DNA sequence.

Recent technological advances allow for high throughput profiling of biological systems in a cost-efficient manner. The low cost of data generation is leading us to the "big data" era. The availability of big data provides unprecedented opportunities but also raises new challenges for data mining and analysis.

Applications of Bioinformatics

- Bioinformatics plays a vital role in the areas of structural genomics, functional genomics, and nutritional genomics.
- It covers emerging scientific research and the exploration of proteomes from the overall level of intracellular protein composition (protein profiles), protein structure, protein-protein interaction, and unique activity patterns (e.g. post-translational modifications).
- Bioinformatics is used for transcriptome analysis where mRNA expression levels can be determined.
- Bioinformatics is used to identify and structurally modify a natural product, to design a compound with the desired properties and to assess its therapeutic effects, theoretically.
- Cheminformatics analysis includes analyses such as similarity searching, clustering, QSAR modeling, virtual screening, etc.
- Bioinformatics is playing an increasingly important role in almost all aspects of drug discovery and drug development.
- Bioinformatics tools are very effective in prediction, analysis and interpretation of clinical and preclinical findings.

Anyone, from clinicians to molecular biologists, with access to the internet and relevant websites can now freely discover the composition of biological molecules such as nucleic acids and proteins by using basic bioinformatic tools. This does not imply that handling and analysis of raw genomic data can easily be carried out by all. Bioinformatics is an evolving discipline, and expert bioinformaticians now use complex software programs for retrieving, sorting out, analyzing, predicting, and storing DNA and protein sequence data.

Skillsets required in Bioinformatics

In bioinformatics and data analysis among the various skill sets required, knowledge in statistics and programming are the most important ones. Bioinformatics programming skills are becoming a necessity across many facets of biology and medicine, owed in part to the continuing explosion of biological data aggregation and the complexity and scale of questions now being addressed through modern bioinformatics. Although many are now receiving formal training in bioinformatics through various university degree and certificate programs, this training is often focused strongly on bioinformatics methodology, leaving many important and practical aspects of bioinformatics to self-education and experience.

2. Introduction to R

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

R is a programming language and software environment for statistical analysis, graphics representation and reporting. R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand. A large group of individuals has contributed to R by sending code and bug reports. Since mid-1997 there has been a core group (the "R Core Team") who can modify the R source code archive. R is currently developed by the R Development Core Team. R is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems like Linux, Windows and Mac. This programming language was named **R**, based on the first letter of first name of the two R authors.

Features of R

- R is open source, so it’s free.
- R is cross-plateform compatible, so it can be installed on Windows, MAC OSX and Linux.
- R provides a wide variety of statistical techniques and graphical capabilities.
- R provides the possibility to make a reproducible research by embedding script and results in a single file.
- R has a vast community both in academia and in business.
- R is highly extensible and it has thousands of well-documented extensions (named R packages) for a very broad range of applications in the financial sector, health care.
- It’s easy to create R packages for solving particular problems.

Installing R

R can be downloaded and installed on Windows, MAC OSX and Linux platforms from the Comprehensive R Archive Network (CRAN) webpage (http://cran.r-project.org/).

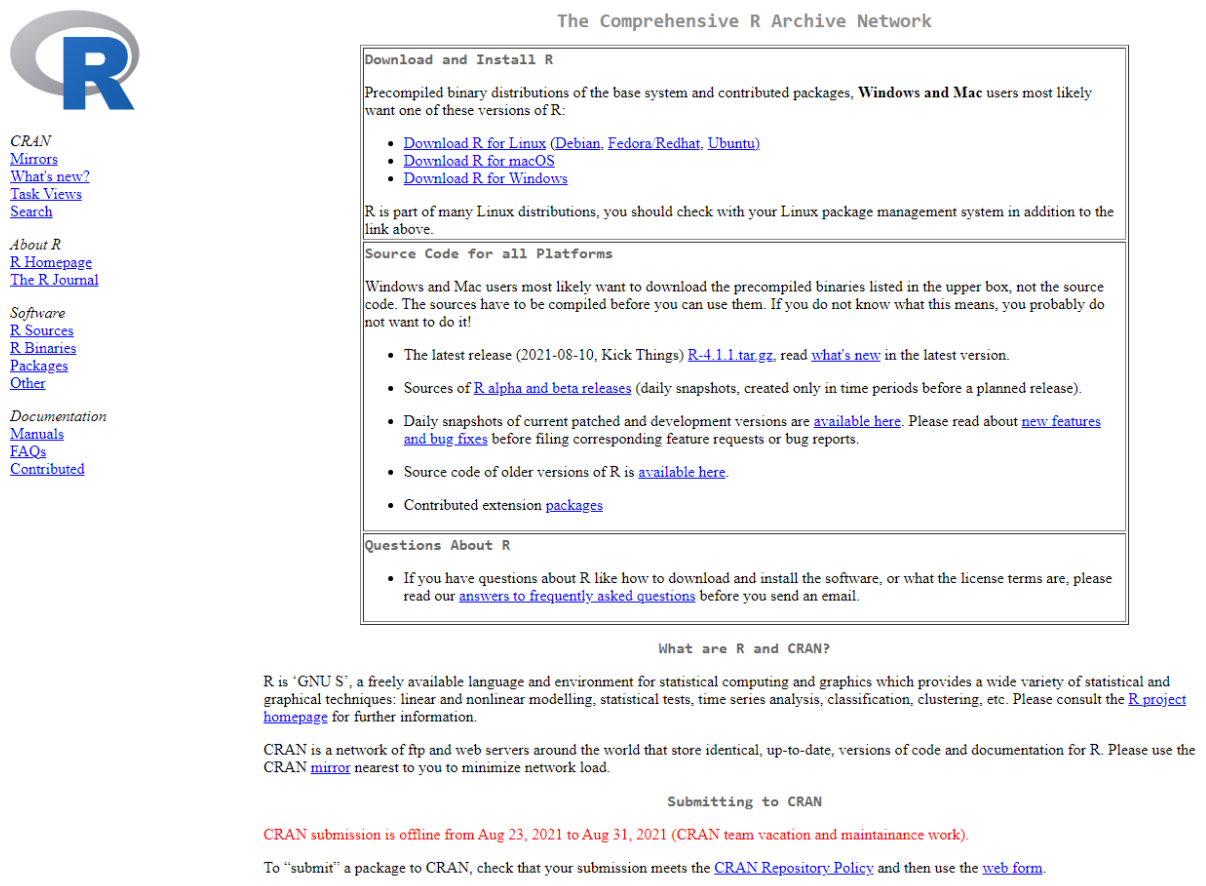
Download R based on your operating system. Latest R version is R 4.1.1 (Kick Things) released on 10-08-2021.

Windows:

Download the latest version of R, for Windows. Double-click on the file you just downloaded to install R with default parameters.

Linux:

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



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
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 (2021-08-10, Kick Things) R-4.1.1.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?

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the R project homepage for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

Submitting to CRAN

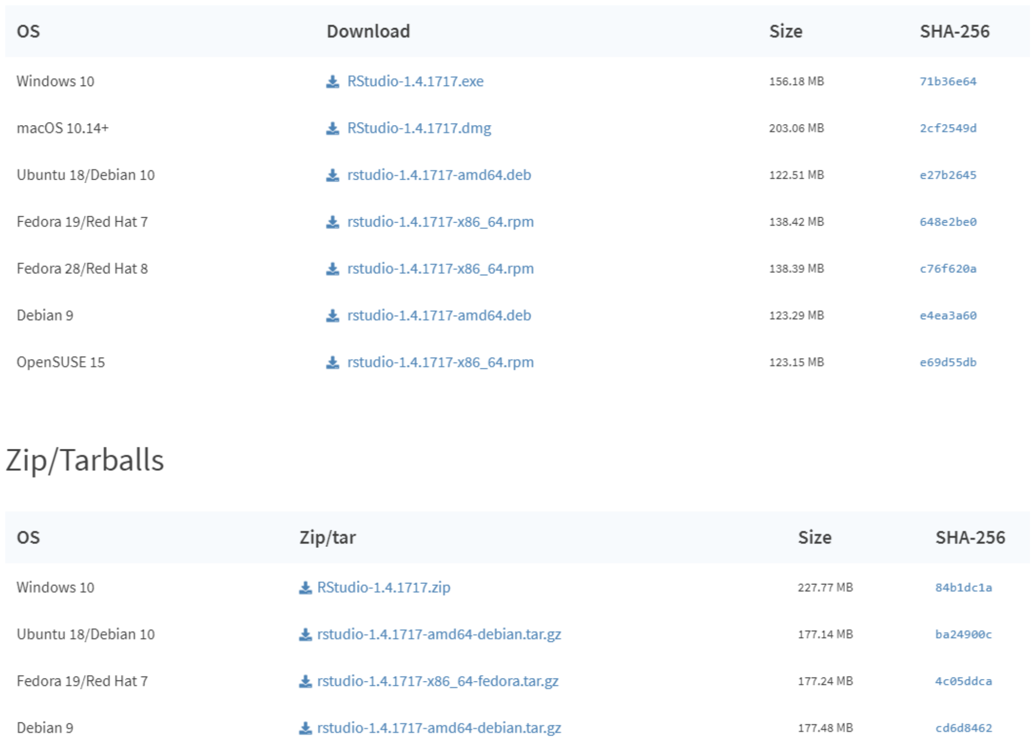
CRAN submission is offline from Aug 23, 2021 to Aug 31, 2021 (CRAN team vacation and maintainance work).

To "submit" a package to CRAN, check that your submission meets the CRAN Repository Policy and then use the web form.

Installing RStudio

RStudio for the desired operating system can be downloaded from:

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



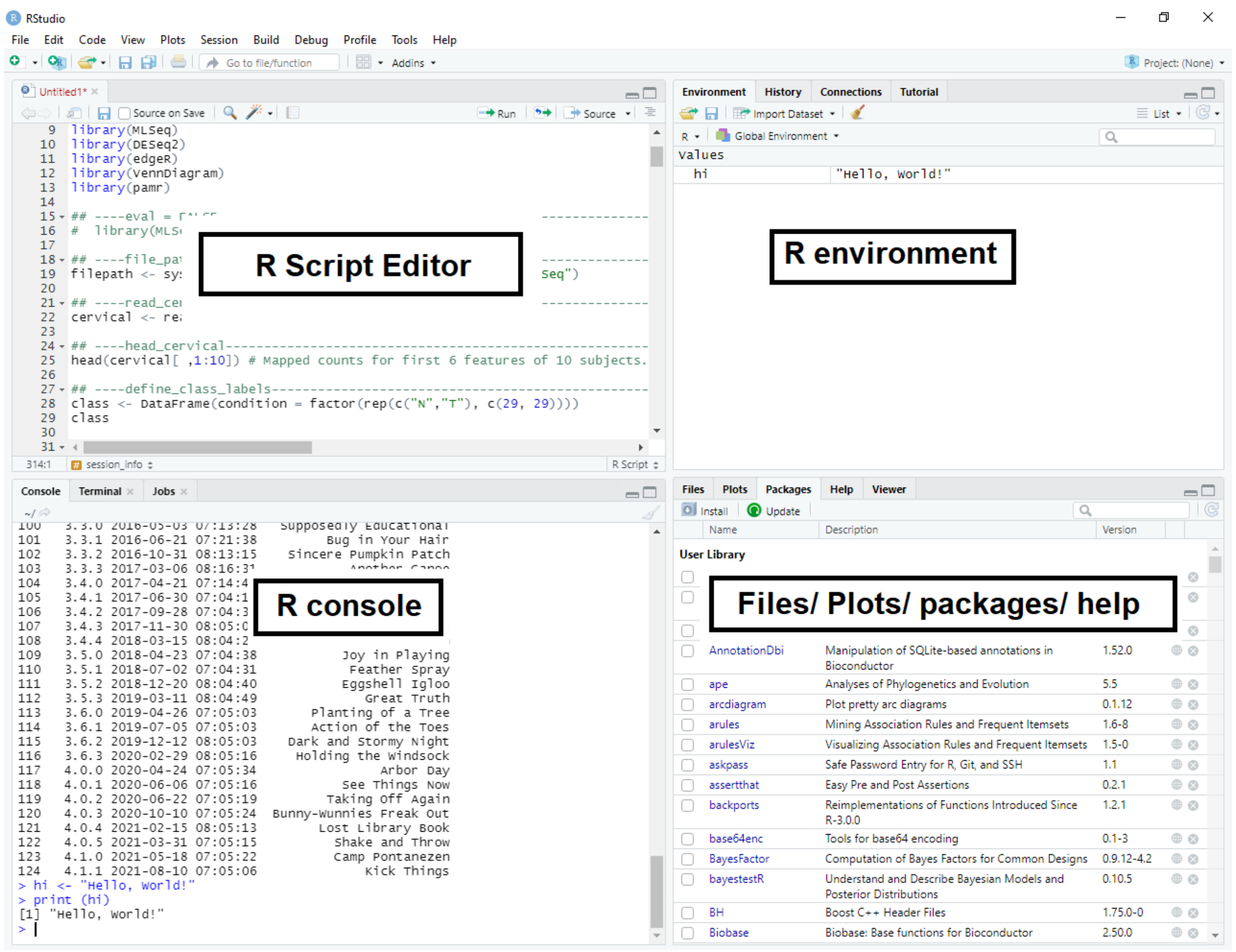
OS	Download	Size	SHA-256
Windows 10	RStudio-1.4.1717.exe	156.18 MB	71b36e64
macOS 10.14+	RStudio-1.4.1717.dmg	203.06 MB	2cf2549d
Ubuntu 18/Debian 10	rstudio-1.4.1717-amd64.deb	122.51 MB	e27b2645
Fedora 19/Red Hat 7	rstudio-1.4.1717-x86_64.rpm	138.42 MB	648e2be0
Fedora 28/Red Hat 8	rstudio-1.4.1717-x86_64.rpm	138.39 MB	c76f620a
Debian 9	rstudio-1.4.1717-amd64.deb	123.29 MB	e4ea3a60
OpenSUSE 15	rstudio-1.4.1717-x86_64.rpm	123.15 MB	e69d55db

Zip/Tarballs

OS	Zip/tar	Size	SHA-256
Windows 10	RStudio-1.4.1717.zip	227.77 MB	84b1dc1a
Ubuntu 18/Debian 10	rstudio-1.4.1717-amd64-debian.tar.gz	177.14 MB	ba24900c
Fedora 19/Red Hat 7	rstudio-1.4.1717-x86_64-fedora.tar.gz	177.24 MB	4c05ddca
Debian 9	rstudio-1.4.1717-amd64-debian.tar.gz	177.48 MB	cd6d8462

Comprehensive R Archive Network

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. The CRAN mirror India server is hosted by National Institute of Science Education and Research (NISER), Bhubaneswar.



```
library(MLSeq)
library(DESeq2)
library(edgeR)
library(VennDiagram)
library(pamr)

## ----eval = FALSE                                          --------------
#  library(MLS

## ----file_pat                                              --------------
filepath <- sy                                               Seq")

## ----read_cer                                              --------------
cervical <- re

## ----head_cervical------------------------------------------------------
head(cervical[ ,1:10]) # Mapped counts for first 6 features of 10 subjects.

## ----define_class_labels------------------------------------------------
class <- DataFrame(condition = factor(rep(c("N","T"), c(29, 29))))
class
```

```
Values
  hi                "Hello, World!"
```

```
100   3.3.0 2016-05-03 07:13:28   Supposedly Educational
101   3.3.1 2016-06-21 07:21:38        Bug in Your Hair
102   3.3.2 2016-10-31 08:13:15   Sincere Pumpkin Patch
103   3.3.3 2017-03-06 08:16:31           Another Canoe
104   3.4.0 2017-04-21 07:14:4
105   3.4.1 2017-06-30 07:04:1
106   3.4.2 2017-09-28 07:04:3
107   3.4.3 2017-11-30 08:05:0
108   3.4.4 2018-03-15 08:04:2
109   3.5.0 2018-04-23 07:04:38          Joy in Playing
110   3.5.1 2018-07-02 07:04:31           Feather Spray
111   3.5.2 2018-12-20 08:04:40          Eggshell Igloo
112   3.5.3 2019-03-11 08:04:49             Great Truth
113   3.6.0 2019-04-26 07:05:03       Planting of a Tree
114   3.6.1 2019-07-05 07:05:03       Action of the Toes
115   3.6.2 2019-12-12 08:05:03     Dark and Stormy Night
116   3.6.3 2020-02-29 08:05:16     Holding the Windsock
117   4.0.0 2020-04-24 07:05:34                Arbor Day
118   4.0.1 2020-06-06 07:05:16           See Things Now
119   4.0.2 2020-06-22 07:05:19        Taking Off Again
120   4.0.3 2020-10-10 07:05:24  Bunny-Wunnies Freak Out
121   4.0.4 2021-02-15 08:05:13       Lost Library Book
122   4.0.5 2021-03-31 07:05:15         Shake and Throw
123   4.1.0 2021-05-18 07:05:22         Camp Pontanezen
124   4.1.1 2021-08-10 07:05:06             Kick Things
> hi <- "Hello, World!"
> print (hi)
[1] "Hello, World!"
> 
```

Name	Description	Version
AnnotationDbi	Manipulation of SQLite-based annotations in Bioconductor	1.52.0
ape	Analyses of Phylogenetics and Evolution	5.5
arcdiagram	Plot pretty arc diagrams	0.1.12
arules	Mining Association Rules and Frequent Itemsets	1.6-8
arulesViz	Visualizing Association Rules and Frequent Itemsets	1.5-0
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.2.1
base64enc	Tools for base64 encoding	0.1-3
BayesFactor	Computation of Bayes Factors for Common Designs	0.9.12-4.2
bayestestR	Understand and Describe Bayesian Models and Posterior Distributions	0.10.5
BH	Boost C++ Header Files	1.75.0-0
Biobase	Biobase: Base functions for Bioconductor	2.50.0

Parts of RStudio

When you open RStudio, you will notice a lot of different windows, each with some panes.

The script editor pane

The R console pane

The R console is where you give R commands and is the lower left window in RStudio. It is the only part of RStudio that is actually R itself.

The R environment pane

The Environment pane in the top right window lists the variables and functions present in the current R session.

Files/ Plots/ Packages/ Help pane

The default tab in the lower right window is a basic file browser. You can open, delete, and rename files there. The graphs and charts developed are displayed in the Plots tab.

The Plots tab is where all of the charts and plots you generate will appear. In this tab you may configure, zoom, and inspect your graphs. The Plots tab will contain all of your graphs that have been run in your current Console section. To toggle between graphs, select the arrows and go to the graphic you wish to see. You additionally may export the graphics created in the Plots tab by selecting **Export** and deciding how you want your graphic saved.

The Packages tab allows you to install additional packages into your RStudio. Packages allow RStudio to perform specific functions. With your RStudio installation, many packages are included. When the box is checked, the package is loaded into RStudio. When the box is not checked, any command typed requiring that function will not work until you select the proper box. A brief description of what each package does is listed next to the packages' name.

The Help tab allows you to search the Help directly from your RStudio window. If you wish to search a direct term or concept from your Console, simply enter a question mark before the command name and the Help window will automatically open.

Versions of R

The R version numbers and dates are extracted from the main R SVN repository at https://svn.r-project.org/R/.

install.packages("rversions")	Install rversions
r_release()	Current R release
r_oldrel()	Previous release

r_versions()	All R versions and release date

Working Directory

The working directory is just a file path on your computer that sets the default location of any files you read into R, or save out of R.

getwd()	To get the present working directory
setwd()	To set the working directory

Installing and loading packages

R packages are a collection of R functions, complied code and sample data. They are stored under a directory called "library" in the R environment.

install.packages("name_of_package")	To install a package
installed.packages()	To check what packages are installed on your computer
library("name_of_packages")	Load a package

Vignettes

Vignettes are documents where the authors show some functionalities of their package in a more detailed way.

vignette(package = " name_of_package ")	To check what vignettes are included with the package

Keyboard shortcuts in R

Shortcut	**Function**
Ctrl-Z/Shift-Z	Undo/Redo
Ctrl-Enter	Execute the current line or code selection in the Source pane
Ctrl-Alt-R	Execute all the R code in the currently open file in the Source pane
Ctrl-Left/Right	Navigate code quickly, word by word
Home/End	Navigate to the beginning/end of the current line
Alt-Shift-Up/Down	Duplicate the current line up or down
Ctrl-D	Delete the current line

Datatypes in R

Numeric: Numbers that have a decimal value or are a fraction in nature have a data type as numeric.

Integer: Numbers that do not contain decimal values have a data type as an integer.

Character: As the name suggests, it can be a letter or a combination of letters enclosed by quotes is considered as a character data type by R. It can be alphabets or numbers.

Logical: A variable that can have a value of True and False like a boolean is called a logical variable.

Factor: They are a data type that is used to refer to a qualitative relationship like colors, good & bad, treatment vs normal, etc.

Data Structures in R

Vector

A vector is a sequence of data elements of the same basic type. Members in a vector are officially called components.

Matrix

A matrix is a two-dimensional rectangular data set and thus it can be created using vector input to the matrix function.

Array

In an array, data is stored in the form of matrices, row, and as well as in columns. We can use the matrix level, row index, and column index to access the matrix elements.

Difference between arrays and matrices

Arrays	**Matrices**
Arrays can contain greater than or equal to 1 dimensions.	Matrices contains 2 dimensions in a table like structure.
Array is a homogeneous data structure.	Matrix is also a homogeneous data structure.
It is a singular vector arranged into the specified dimensions.	It comprises of multiple equal length vectors stacked together in a table.
array() function can be used to create matrix by specifying the third dimension to be 1.	**matrix()** function however can be used to create at most 2-dimensional array.
Arrays are superset of matrices.	Matrices are a subset, special case of array where dimensions is two.

Limited set of collection-based operations.	Wide range of collection operations possible.
Mostly, intended for storage of data.	Mostly, matrices are intended for data transformation.

Lists

Lists are the objects which contain elements of different types – like strings, numbers, vectors and another list inside them. A list can also contain a matrix or a function as its elements. In other words, a list is a generic vector containing other objects.

Data Frames

It generally refers to tabular data: a data structure representing the cases (rows), each of which consists of numbers of observation or measurement (columns).A data frame is used for storing data tables. It is a list of vectors of equal length.

Characteristics of a Data Frame:

- The column names should be non-empty.
- The row names should be unique.
- The data stored in a data frame can be of numeric, factor or character type.
- Each column should contain the same number of data items.
- Datasets imported in R are stored as data frames by default.

Datasets for R

R comes with several built-in data sets, which are generally used as demo data for playing with R functions.

data(package = "datasets")	All datasets in Datasets package

library(help = "datasets")	Details about the datatsets

List of datasets

AirPassengers	Monthly Airline Passenger Numbers 1949-1960
BJsales	Sales Data with Leading Indicator
BOD	Biochemical Oxygen Demand
CO2	Carbon Dioxide Uptake in Grass Plants
ChickWeight	Weight versus age of chicks on different diets
DNase	Elisa assay of DNase
EuStockMarkets	Daily Closing Prices of Major European Stock Indices, 1991-1998
Formaldehyde	Determination of Formaldehyde
HairEyeColor	Hair and Eye Color of Statistics Students
Harman23.cor	Harman Example 2.3
Harman74.cor	Harman Example 7.4
Indometh	Pharmacokinetics of Indomethacin
InsectSprays	Effectiveness of Insect Sprays
JohnsonJohnson	Quarterly Earnings per Johnson & Johnson Share
LakeHuron	Level of Lake Huron 1875-1972
LifeCycleSavings	Intercountry Life-Cycle Savings Data
Loblolly	Growth of Loblolly pine trees
Nile	Flow of the River Nile

Orange	Growth of Orange Trees
OrchardSprays	Potency of Orchard Sprays
PlantGrowth	Results from an Experiment on Plant Growth
Puromycin	Reaction Velocity of an Enzymatic Reaction
Theoph	Pharmacokinetics of Theophylline
Titanic	Survival of passengers on the Titanic
ToothGrowth	The Effect of Vitamin C on Tooth Growth in Guinea Pigs
UCBAdmissions	Student Admissions at UC Berkeley
UKDriverDeaths	Road Casualties in Great Britain 1969-84
UKLungDeaths	Monthly Deaths from Lung Diseases in the UK
UKgas	UK Quarterly Gas Consumption
USAccDeaths	Accidental Deaths in the US 1973-1978
USArrests	Violent Crime Rates by US State
USJudgeRatings	Lawyers' Ratings of State Judges in the US Superior Court
USPersonalExpenditure	Personal Expenditure Data
VADeaths	Death Rates in Virginia (1940)
WWWusage	Internet Usage per Minute
WorldPhones	The World's Telephones
ability.cov	Ability and Intelligence Tests
airmiles	Passenger Miles on Commercial US Airlines, 1937-1960
airquality	New York Air Quality Measurements

anscombe	Anscombe's Quartet of 'Identical' Simple Linear Regressions
attenu	The Joyner-Boore Attenuation Data
attitude	The Chatterjee-Price Attitude Data
austres	Quarterly Time Series of the Number of Australian Residents
beavers	Body Temperature Series of Two Beavers
cars	Speed and Stopping Distances of Cars
chickwts	Chicken Weights by Feed Type
co2	Mauna Loa Atmospheric CO2 Concentration
crimtab	Student's 3000 Criminals Data
datasets-package	The R Datasets Package
discoveries	Yearly Numbers of Important Discoveries
esoph	Smoking, Alcohol and (O)esophageal Cancer
euro	Conversion Rates of Euro Currencies
eurodist	Distances Between European Cities and Between US Cities
faithful	Old Faithful Geyser Data
freeny	Freeny's Revenue Data
infert	Infertility after Spontaneous and Induced Abortion
iris	Edgar Anderson's Iris Data
islands	Areas of the World's Major Landmasses
lh	Luteinizing Hormone in Blood Samples
longley	Longley's Economic Regression Data
lynx	Annual Canadian Lynx trappings 1821-1934

morley	Michelson Speed of Light Data
mtcars	Motor Trend Car Road Tests
nhtemp	Average Yearly Temperatures in New Haven
nottem	Average Monthly Temperatures at Nottingham, 1920-1939 Classical N, P, K Factorial Experiment
occupationalStatus	Occupational Status of Fathers and their Sons
precip	Annual Precipitation in US Cities
presidents	Quarterly Approval Ratings of US Presidents
pressure	Vapor Pressure of Mercury as a Function of Temperature
quakes	Locations of Earthquakes off Fiji
randu	Random Numbers from Congruential Generator RANDU
rivers	Lengths of Major North American Rivers
rock	Measurements on Petroleum Rock Samples
sleep	Student's Sleep Data
stackloss	Brownlee's Stack Loss Plant Data
state	US State Facts and Figures
sunspot.month	Monthly Sunspot Data, from 1749 to "Present"
sunspot.year	Yearly Sunspot Data, 1700-1988
sunspots	Monthly Sunspot Numbers, 1749-1983
swiss	Swiss Fertility and Socioeconomic Indicators (1888) Data
treering	Yearly Treering Data, -6000-1979
trees	Diameter, Height and Volume for Black Cherry Trees

uspop	Populations Recorded by the US Census
volcano	Topographic Information on Auckland's MaungaWhau Volcano
warpbreaks	The Number of Breaks in Yarn during Weaving
women	Average Heights and Weights for American Women

File input

read.csv()	Reads a CSV file into the memory.
read.delim()	Used to read in delimited text files
read.table()	Loads data from a file into a tabular data set (table) in memory.

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.

3. Introduction to R coding: Control Statements

Divya P Syamaladevi
Senior Scientist (Agricultural Biotechnology), ICAR-ISR, Calicut
Email: P.Syamaladevi@icar.gov.in

Today, all leading organizations of research and development as well as marketing firms across the world are heavily depending on data science in decision making. Towards this, data are being obtained from biological and molecular experiments or from internet and their business. These data need to be analysed to make key decisions and draw meaningful conclusions pertaining to experiments and there by remain competitive in business or academic research.

The information collected can be erroneous or become outdated and may require cleaning or updates. The cleaning process involves either removing or updating of incomplete data, removing duplicates, imputing missing values, format improperly formatted data, and so on. According to various studies, Data Cleaning constitutes 57% to 60% of the weight in a Data Science project. Data Science further involves the processing of large sets of data with statistical methods to extract trends, patterns, or other relevant information. In short, data science encapsulates anything related to obtaining insights, trends, or any other valuable information from data. The foundations of these tasks originate from the fields of statistics, programming, and visualization. In short, a successful data scientist should have in-depth knowledge in the following aspects

Math and Statistics: From modeling to experimental design, encountering something math-related is inevitable, as data almost always requires quantitative analysis

Programming and Database: Knowing how to navigate program data hierarchies, or bigdata, and query certain datasets alongside knowing how to code algorithms and develop models is invaluable to a data scientist.

Communication and visualization: To make their work viable for all audiences, data scientists must be able to weave a coherent and impactful story through visuals and facts to convey the importance of their work. This is usually completed with certain programming languages or data visualization software, such as R

Why would a Biologist turning in to data scientist want to know coding during data analysis?

As Data Science Managers or Biologists, will you be expected to code?

Probably you may not need to code but are definitely required to structure the logic of analysis. Coding plays a significant role in data science, making appearances in almost every step of the process. It's important to remember that this process is not always linear; data scientists tend to ping-pong back and forth between different steps depending on the nature of the question in hand.

In Biology and Molecular Biology data including genomic transcriptomic sequences and proteomic metabolomic and metagenomics related data would require analysis before arriving at meaningful conclusions. In general Biologists make use of many availble statistical models. However one would be interested in developing newer models or wish to fit modified models to their data to derive better or improved conclusions. That is where a Biologists knowledge of any one (or more) programming language(s) becomes very critical.

You may not be required to write many lines of code, but certainly, coding will be involved in analyzing data and making predictive models using pre-made algorithm / libraries. When you see a snippet of code you should be able to understand what it does to your input data. You don't have to do deep coding for each and every task, there are modules for doing different tasks. But when you handle big data and have to do the same thing repetitively you can definitely think of having a little loop which would do your repetitive task and you can sit back and enjoy!!

R is a programming language and software environment commonly used in statistical computing, data analytics and scientific research. It is one of the most popular languages used by statisticians, data analysts, researchers and marketers to retrieve, clean, analyze, visualize and present data. Due to its expressive syntax and easy-to-use interface, it has grown in popularity in recent years.

• R is a programming language and software environment for statistical analysis, graphics representation and reporting. R was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R DevelopmentCore Team.

• The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions

Why learn R Programming

Here are some advantages of R programming and why learning it might be a good idea for you:

- R is a platform-independent programming language. This means that whichever operating system you use, your R program will work just fine.
- R is very easy to learn and understand. If you have a good understanding of statistics, R programming will make your tasks easier.
- R libraries provide one of the best and most insightful data visualizations.
- R programming is one of the most popular programming languages for data science and machine learning.
- R can easily be integrated with various other programming languages such as C and C++.
- R is a free language; anyone can download and use it without having to purchase a license. It is also open-source.
- The demand for R is growing at a very fast rate and it is currently a trend in the industry.
- R has a huge community of users and extensive community support to help you with the learning process.

A lot of programmers choose R over Python these days due to the following reasons:

- Even novices can start doing data analysis quickly on R, which was designed specifically keeping statisticians in mind.

- R is better suited, as compared to Python, when it comes to statistical learning. R programming has exceptional libraries for exploring and experimenting with data.
- With amazing graphics, R is perfect for data visualization.

Definition of some related common terms

Computer Programming: Computer programming is the process of designing and building an executable computer program to accomplish a specific computing result or to perform a specific task

A programming language is a formal language comprising a set of strings that produce various kinds of machine code output. Programming languages are one kind of computer language, and are used in computer programming to implement algorithms. Most programming languages consist of instructions for computers.

Syntax is the structure of statements in a computer language. Eg: syntax of a language, the arrangement of words and phrases to create well-formed sentences in a language

Code In computer programming, computer code refers to the set of instructions, or a system of rules, written in a particular programming language

Algorithm is a finite sequence of well-defined, computer-implementable instructions, typically to solve a class of specific problems or to perform a computation.

Control statements are expressions used to control the execution and flow of the program based on the conditions provided in the statements. These structures are used to make a decision after assessing the variable.

4. Basics of Statistics

J. Sreekumar, Principal Scientist (Agricultural Statistics),
ICAR-Central Tuber Crops Research Institute, Thiruvananthapuram
Email: Sreekumar.J@icar.gov.in

Levels of variables

This scheme was developed by Stevens and published in 1946.
A categorical variable, also called a nominal variableA ordinal variable, is one where the order matters but not the difference between values. An interval variable is a measurement where the difference between two values is meaningful. A ratio variable has all the properties of an interval variable, and also has a clear definition of 0.0.

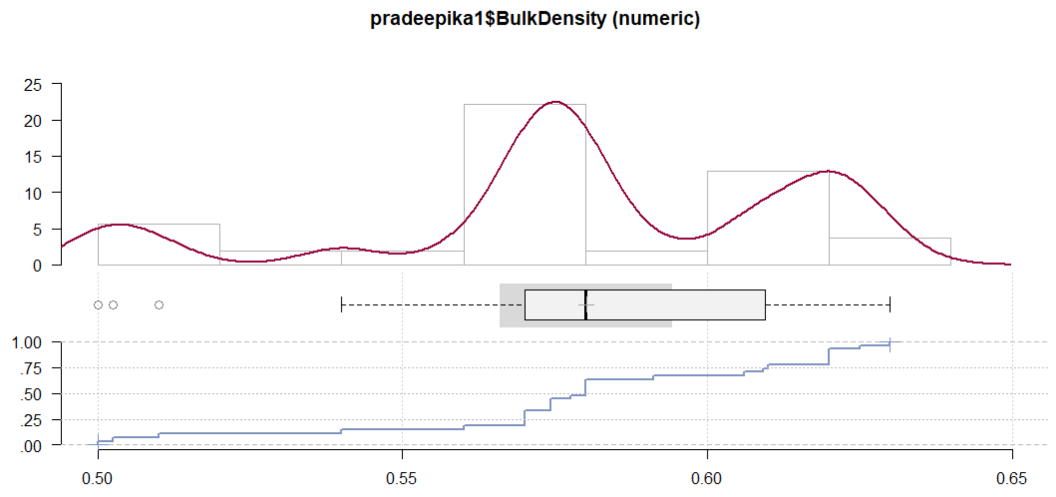
Descriptive Statistics
Types of descriptive statistics

There are 3 main types of descriptive statistics:

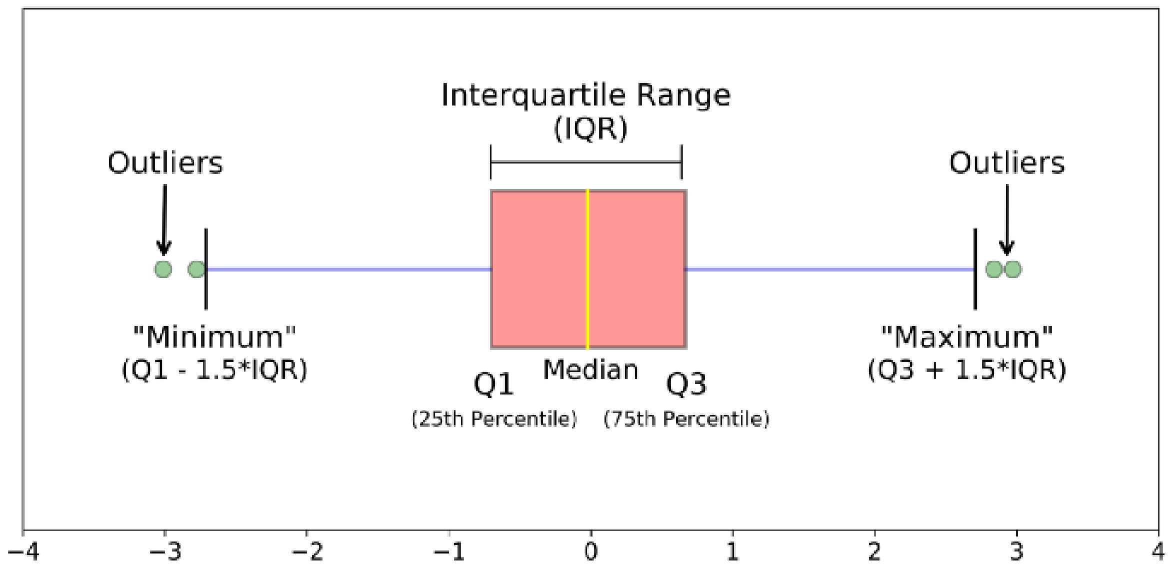
- The distribution concerns the frequency of each value.
- The central tendency concerns the averages of the values.
- The variability or dispersion concerns how spread out the values are.

You can apply these to assess only one variable at a time, in univariate analysis, or to compare two or more, in bivariate and multivariate analysis.

In descriptive statistics no prediction will be there understand the data of current samples (Mean, Median, Mode, IQR, Range).*Descriptive statistics* will answer the question to client *what happened?* by means of Exploratory Data Analysis.After Completing EDA in prediction with the help of model it will answer the question to client *what will happen?* is called *Inferential statistics*.
Measures of Central Tendency are Mean, Median, Mode
Measures of Dispersion are Variance, standard deviation, Range, Quartile Deviation etc



Box plot

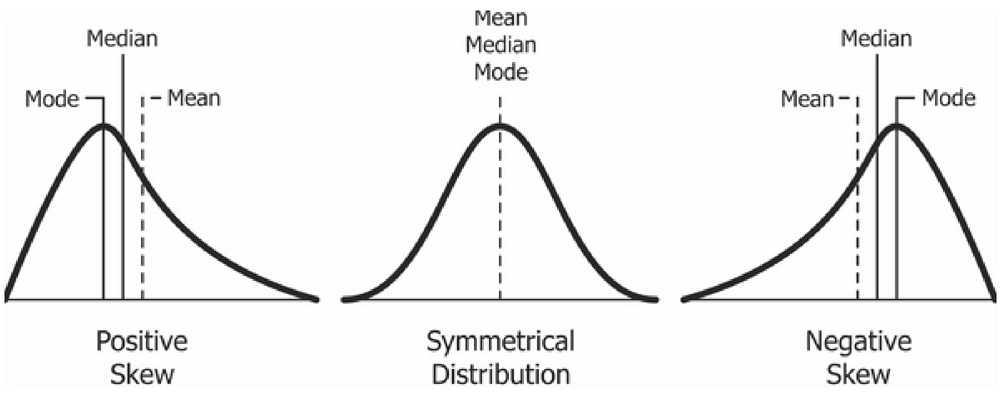


The **mean,** is the most commonly used method for finding the average.

To find the mean, simply add up all response values and divide the sum by the total number of responses. The total number of responses or observations is called N.

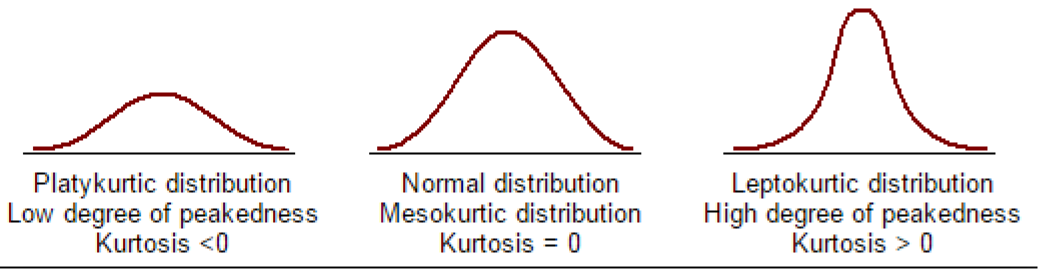
Standard deviation

The standard deviation (s) is the average amount of variability in your dataset. It tells you, on average, how far each score lies from the mean. The larger the standard deviation, the more variable the data set is.



Skewness: Measure of symmetric and asymmetric.
Mean=Median : symmetry ,Mean>Median : Right-skewed , Mean < Median : Left-skewed.
kurtosis: Measure of peakedness.

- Mesokurtic — This is the case when the kurtosis is zero, similar to the normal distributions.
- Leptokurtic — This is when the tail of the distribution is heavy (outlier present) and kurtosis is higher than that of the normal distribution.
- Platykurtic — This is when the tail of the distribution is light(no outlier) and kurtosis is lesser than that of the normal distribution.



Visualization and Tests of Normality

Histogram

Histograms are exploratory plot because they show densities of data and can help provide distributional information.

The R function **shapiro.test**() can be used to perform the Shapiro-Wilk test of normality for one variable (univariate):

Testing of Hypothesis

Writing statistical hypotheses

The goal of research is often to investigate a relationship between variables within a population. You start with a prediction and use statistical analysis to test that prediction.

A statistical hypothesis is a formal way of writing a prediction about a population. Every research prediction is rephrased into null and alternative hypotheses that can be tested using sample data.

While the null hypothesis always predicts no effect or no relationship between variables, the alternative hypothesis states your research prediction of an effect or relationship.

What exactly is a *p*-value?

The *p*-**value**, or probability value, tells you how likely it is that your data could have occurred under the null hypothesis. It does this by calculating the likelihood of your **test statistic**, which is the number calculated by a statistical test using your data.

The *p*-value tells you how often you would expect to see a test statistic as extreme or more extreme than the one calculated by your statistical test if the null hypothesis of that test was true. The *p*-value gets smaller as the test statistic calculated from your data gets further away from the range of test statistics predicted by the null hypothesis.

The *p*-value is a proportion: if your *p*-value is 0.05, that means that 5% of the time you would see a test statistic at least as extreme as the one you found if the null hypothesis was true.

Statistical significance is another way of saying that the *p*-value of a statistical test is small enough to reject the null hypothesis of the test.

How small is small enough? The most common threshold is $p < 0.05$; that is, when you would expect to find a test statistic as extreme as the one calculated by your test only 5% of the time. But the threshold depends on your field of study – some fields prefer thresholds of 0.01, or even 0.001.

The threshold value for determining statistical significance is also known as the alpha value.

Reporting the results

In our comparison of mouse diet A and mouse diet B, we found that the lifespan on diet A (mean = 2.1 years; sd = 0.12) was significantly shorter than the lifespan on diet B (mean = 2.6 years; sd = 0.1), with an average difference of 6 months (t(80) = -12.75; $p < 0.01$).

Errors in inference

For the most part, the statistical tests we use are based on probability, and our data could always be the result of chance. Considering the coin flipping example, if we did flip a coin 100 times and came up with 95 heads, we would be compelled to conclude that the coin was not fair. But 95 heads could happen with a fair coin strictly by chance.

The following table summarizes these errors.

	Reality	
Decision of Test	Null is true	Null is false
Reject null hypothesis	Type I error (prob. = alpha)	Correctly reject null (prob. = 1 – beta)
Retain null hypothesis	Correctly retain null (prob. = 1 – alpha)	Type II error (prob. = beta)

We can, therefore, make two kinds of errors in testing the null hypothesis:

- A Type I error occurs when the null hypothesis really is true but based on our decision rule, we reject the null hypothesis. In this case, our result is a false positive; we think there is an effect (unfair coin, association between variables, difference among groups) when really there isn't. The probability of making this kind error is alpha, the same alpha we used in our decision rule.
- A Type II error occurs when the null hypothesis is really false but based on our decision rule, we fail to reject the null hypothesis. In this case, our result is a false

negative; we have failed to find an effect that really does exist. The probability of making this kind of error is called beta.

t-test

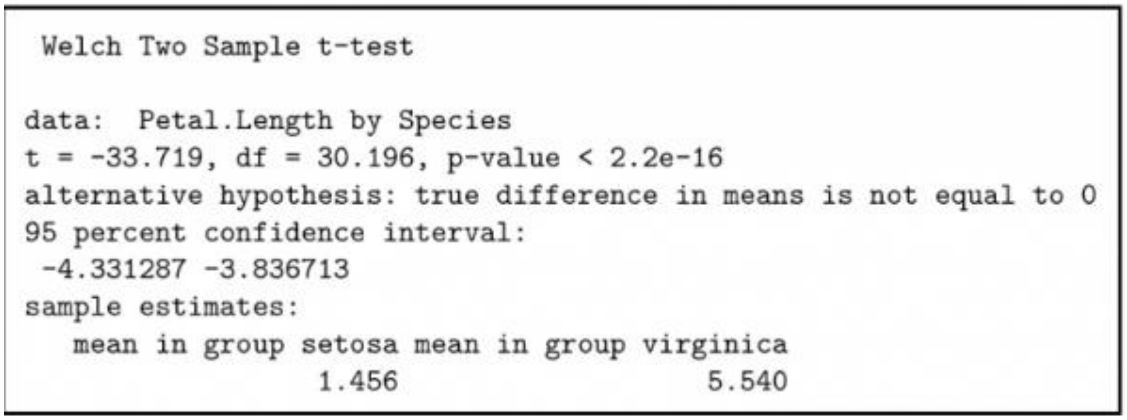
The t-test is a parametric test of difference, meaning that it makes the same assumptions about your data as other parametric tests. The t-test assumes dataare independent and are (approximately) normally distributed and data have a similar amount of variance within each group being compared (a.k.a. homogeneity of variance)If your data do not fit these assumptions, you can try a nonparametric alternative to the t-test, such as the Wilcoxon Signed-Rank test for data with unequal variances.

One-sample, two-sample, or paired t-test?

- If the groups come from a single population (e.g. measuring before and after an experimental treatment), perform a paired t-test.
- If the groups come from two different populations (e.g. two different species, or people from two separate cities), perform a two-sample t-test (a.k.a. independent t-test).
- If there is one group being compared against a standard value (e.g. comparing the acidity of a liquid to a neutral pH of 7), perform a one-sample t-test.

One-tailed or two-tailed t-test?

- If you only care whether the two populations are different from one another, perform a two-tailed t-test.
- If you want to know whether one population mean is greater than or less than the other, perform a one-tailed t-test.



```
 Welch Two Sample t-test

data:  Petal.Length by Species
t = -33.719, df = 30.196, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -4.331287 -3.836713
sample estimates:
   mean in group setosa mean in group virginica
                  1.456                   5.540
```

The output provides:

- An explanation of what is being compared, called data in the output table.
- The t-value: -33.719. Note that it's negative; this is fine! In most cases, we only care about the absolute value of the difference, or the distance from 0. It doesn't matter which direction.
- The degrees of freedom: 30.196. Degrees of freedom is related to your sample size, and shows how many 'free' data points are available in your test for making comparisons. The greater the degrees of freedom, the better your statistical test will work.
- The p-value: 2.2e-16 (i.e. 2.2 with 15 zeros in front). This describes the probability that you would see a t-value as large as this one by chance.
- A statement of the alternate hypothesis (Ha). In this test, the Ha is that the difference is not 0.
- The 95% confidence interval. This is the range of numbers within which the true difference in means will be 95% of the time. This can be changed from 95% if you want a larger or smaller interval, but 95% is very commonly used.
- The mean petal length for each group.

Correlation: estimate and test linear relationship between two variables

Covariance and correlation:

Covariance measures how change in one variable is associated with other variable. It gives the together spread of data.For example: if we want to compare weight of the person is related to age or height we can’t able to compare because it is in units kg/yrs and kg/cm so we need to break the unit so comes concept called correlation it scales the unit and break the units and gives linear association between 2 numeric variables. Now we get weight-age is 0.7 and weight-height is 0.88 so we can say weight is highly dependent on height variable.

For Population

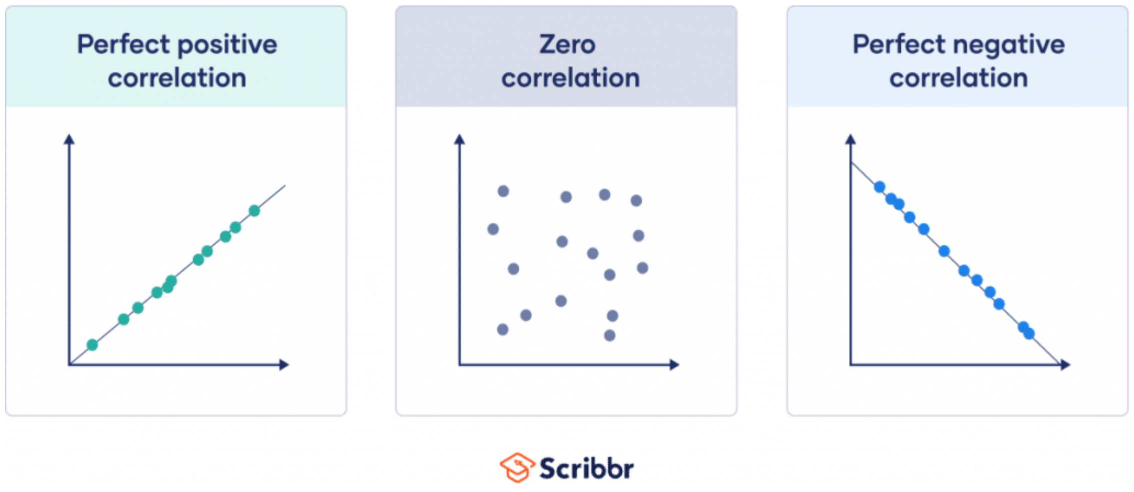
$$\text{Cov}(x,y) = \frac{\Sigma\,(x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

For Sample

$$\text{Cov}(x,y) = \frac{\Sigma\,(x_i - \bar{x}) * (y_i - \bar{y})}{(N - 1)}$$

A correlation coefficient is a number between -1 and 1 that tells you the strength and direction of a relationship between variables.In other words, it reflects how similar the measurements of two or more variables are across a dataset.

Correlation coefficient value	Correlation type	Meaning
1	Perfect positive correlation	When one variable changes, the other variables change in the same direction.
0	Zero correlation	There is no relationship between the variables.
-1	Perfect negative correlation	When one variable changes, the other variables change in the opposite direction.



Regression

Assumptions of simple Linear regression

Simple linear regression is a parametric test, meaning that it makes certain assumptions about the data. These assumptions are:

1. Homogeneity of variance (homoscedasticity): the size of the error in our prediction doesn't change significantly across the values of the independent variable.
2. Independence of observations: the observations in the dataset were collected using statistically valid sampling methods, and there are no hidden relationships among observations.
3. Normality: The data follows a normal distribution.

Linear regression makes one additional assumption:

The relationship between the independent and dependent variable is linear: the line of best fit through the data points is a straight line (rather than a curve or some sort of grouping factor).

If your data do not meet the assumptions of homoscedasticity or normality, you may be able to use a nonparametric test instead, such as the Spearman rank test.

Simple linear regression formula

The formula for a simple linear regression is:

$$y = \beta_0 + \beta_1 X + \varepsilon$$

y is the predicted value of the dependent variable (**y**) for any given value of the independent variable (**x**).

B0 is the **intercept**, the predicted value of **y** when the **x** is 0.

B1 is the regression coefficient – how much we expect **y** to change as **x** increases.

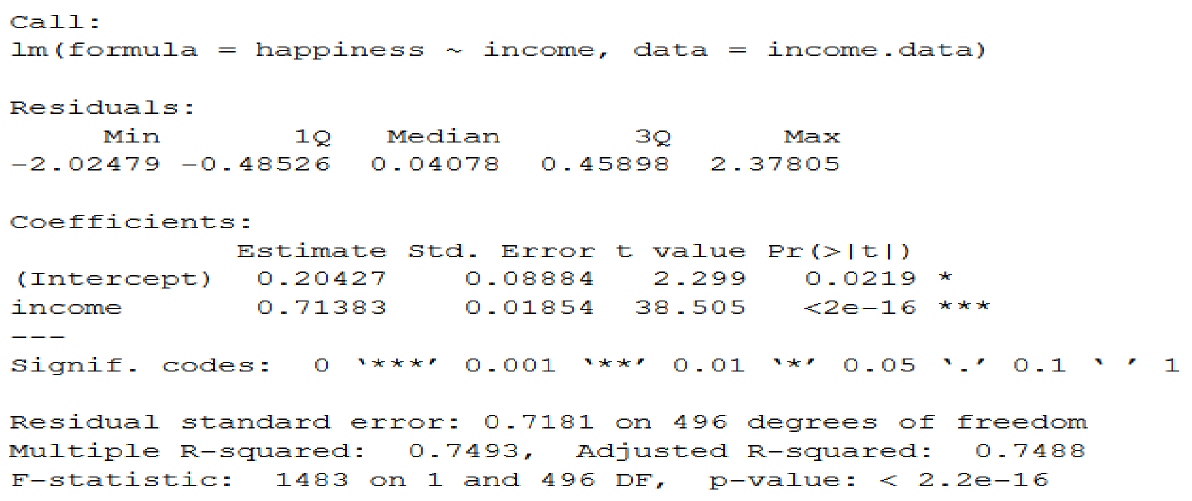
x is the independent variable (the variable we expect is influencing **y**).

e is the **error** of the estimate, or how much variation there is in our estimate of the regression coefficient.

Linear regression finds the line of best fit line through your data by searching for the regression coefficient (B1) that minimizes the total error (e) of the model.

While you can perform a linear regression by hand, this is a tedious process, so most people use statistical programs to help them quickly analyze the data.

Output of regression



```
Call:
lm(formula = happiness ~ income, data = income.data)

Residuals:
     Min       1Q   Median       3Q      Max 
-2.02479 -0.48526  0.04078  0.45898  2.37805 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.20427    0.08884   2.299   0.0219 *  
income       0.71383    0.01854  38.505   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7181 on 496 degrees of freedom
Multiple R-squared:  0.7493,	Adjusted R-squared:  0.7488 
F-statistic:  1483 on 1 and 496 DF,  p-value: < 2.2e-16
```

Presenting the result

We found a significant relationship ($p < 0.001$) between income and happiness (R2 = 0.71 ± 0.018), with a 0.71-unit increase in reported happiness for every $10,000 increase in income.

# ANOVA (ANALYSIS OF VARIANCE): Comparison of multiple mean values

ANOVA is a statistical test for estimating how a quantitative dependent variable changes according to the levels of one or more categorical independent variables. ANOVA tests whether there is a difference in means of the groups at each level of the independent variable.

The null hypothesis (H_0) of the ANOVA is no difference in means, and the alternate hypothesis (H_a) is that the means are different from one another.

Assumptions of ANOVA

The assumptions of the ANOVA test are the same as the general assumptions for any parametric test:

1. Independence of observations: the data were collected using statistically valid methods, and there are no hidden relationships among observations. If your data fail to meet this assumption because you have a confounding variable that you need to control for statistically, use an ANOVA with blocking variables.
2. Normally distributed response variable: The values of the dependent variable follow a normal distribution.
3. Homogeneity of variance: The variation within each group being compared is similar for every group. If the variances are different among the groups, then ANOVA probably isn't the right fit for the data.

```
model <-aov (yield ~ fertilizer, data = field)
```

The summary of an ANOVA test (in R) looks like this:

```
           Df Sum Sq Mean Sq F value Pr(>F)
fertilizer   2   6.07  3.0340   7.863  7e-04 ***
Residuals   93  35.89  0.3859
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1
```

The ANOVA output provides an estimate of how much variation in the dependent variable that can be explained by the independent variable.

- The first column lists the independent variable along with the model residuals (aka the model error).
- The Df column displays the degrees of freedom for the independent variable (calculated by taking the number of levels within the variable and subtracting 1), and the degrees of freedom for the residuals (calculated by taking the total number of observations minus 1, then subtracting the number of levels in each of the independent variables).
- The Sum Sq column displays the sum of squares (a.k.a. the total variation) between the group means and the overall mean explained by that variable. The sum of squares for the fertilizer variable is 6.07, while the sum of squares of the residuals is 35.89.
- The Mean Sq column is the mean of the sum of squares, which is calculated by dividing the sum of squares by the degrees of freedom.
- The F-value column is the test statistic from the F test: the mean square of each independent variable divided by the mean square of the residuals. The larger the F value, the more likely it is that the variation associated with the independent variable is real and not due to chance.
- The Pr(>F) column is the p-value of the F-statistic. This shows how likely it is that the F-value calculated from the test would have occurred if the null hypothesis of no difference among group means were true.

Because the p-value of the independent variable, fertilizer, is significant ($p < 0.05$), it is likely that fertilizer type does have a significant effect on average crop yield.

PCA and Clustering

Principal Component Analysis (PCA) is a popular technique for deriving a set of low dimensional features from a large set of variables. However, another popular application of PCA is visualizing higher dimensional data. In this tutorial, we will go over the basics of PCA and apply it to cluster a data set consisting of houses with different features.
Principal Component Analysis (PCA) is an unsupervised machine learning technique that attempts to derive a set of low-dimensional set of features from a much larger set while still preserving as much variance as possible. Perhaps the two main applications of PCA are

1. Variable selection
2. Visualizing High-Dimensional Data

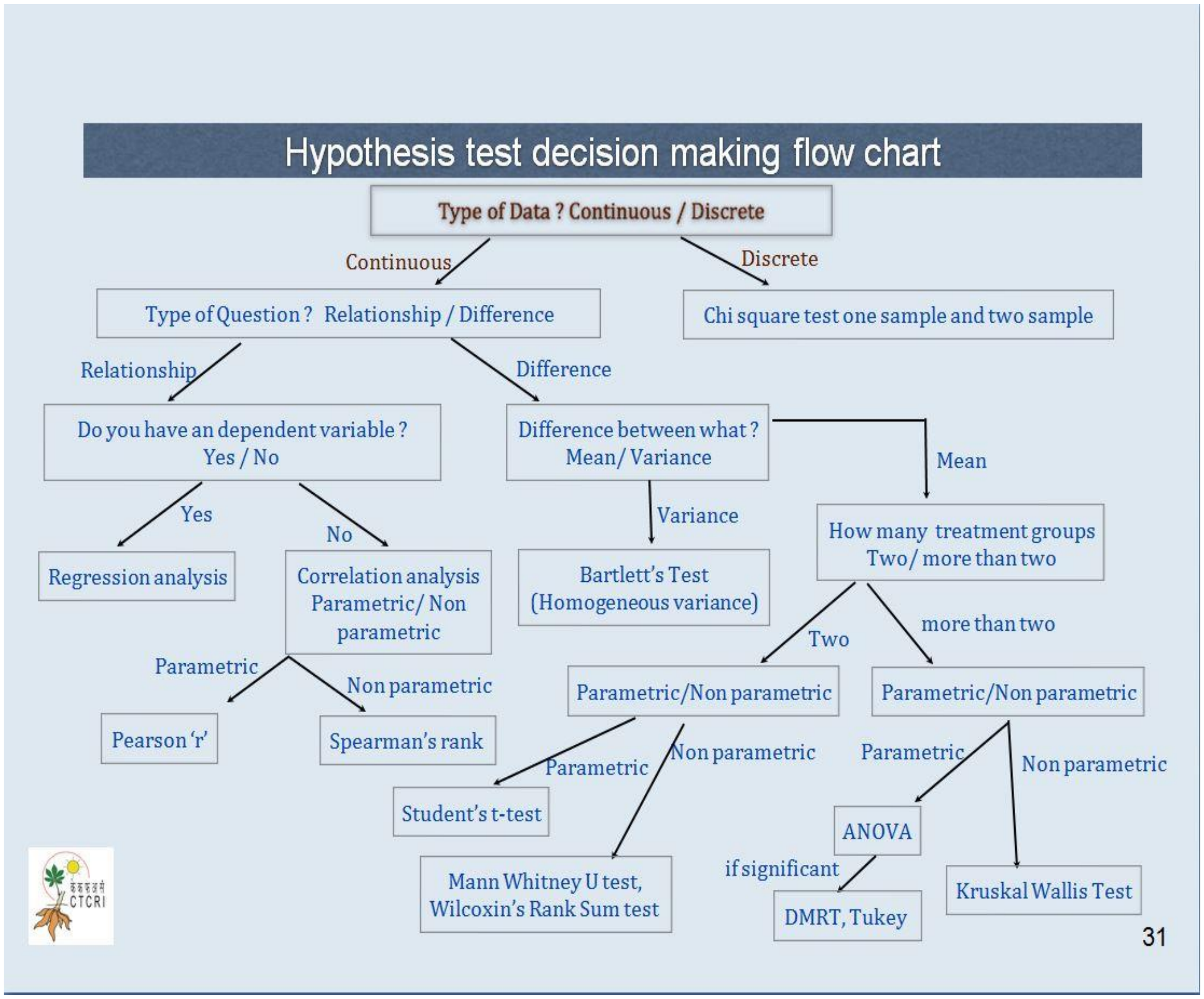
Principal Component Analysis or PCA is a method of reducing the dimensions of the given dataset while still retaining most of its variance. Wikipedia defines it as, "PCA is defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on."

A simple algorithm for PCA is as follows

- Normalize the Data - This is often necessary if the features in your feature set are measured in different units. A common way of normalizing the features in your data set is to convert them to z-scores.
- Calculate the covariance matrix.
- Compute the eigen values and vectors.
- Re-orient the data.
- Plot the data (PC1 against PC2)

Clustering is an unsupervised machine learning algorithm and it recognizes patterns without specific labels and clusters the data according to the features. In our case, we will see if a clustering algorithm (k-means) can find a pattern between different images of the apparel in f-MNIST without the labels (y).

K-means clustering works by assigning a number of centroids based on the number of clusters given. Each data point is assigned to the cluster whose centroid is nearest to it. The algorithm aims to minimize the squared Euclidean distances between the observation and the centroid of cluster to which it belongs.


Hypothesis test decision making flow chart
Type of Data ? Continuous / Discrete
Continuous
Discrete
Type of Question ? Relationship / Difference
Chi square test one sample and two sample
Relationship
Difference
Do you have an dependent variable ? Yes / No
Difference between what ? Mean/ Variance
Mean
Yes
No
Variance
Regression analysis
Correlation analysis Parametric/ Non parametric
Bartlett's Test (Homogeneous variance)
How many treatment groups Two/ more than two
Two
more than two
Parametric
Non parametric
Parametric/Non parametric
Parametric/Non parametric
Pearson 'r'
Spearman's rank
Parametric
Non parametric
Parametric
Non parametric
Student's t-test
ANOVA
if significant
Mann Whitney U test, Wilcoxin's Rank Sum test
DMRT, Tukey
Kruskal Wallis Test
CTCRI
31

5. Introduction to data visualization

Sona Charles, Scientist (Ag. Bioinformatics)
ICAR-IISR, Calicut
Email: sona.charles@icar.gov.in

Why data visualization is important?

The volume of data used in research and technological development is massive and continues to grow. It becomes harder and harder for a user to grab a key message from this universe of data. That's where data visualization comes in: summarizing and presenting large data in simple and easy-to-understand visualizations to give readers insightful information.

There are many advanced visualizations (e.g., networks, 3D-models and map overlays) used for specialized purposes such as 3D medical imaging, agriculturalyield monitoring etc. But regardless of the complexity of a visualization, its purpose is to help readers see a pattern or trend in the data being analyzed, rather than having them read tedious descriptions. A good visualization summarizes information and organizes in a way that enables the reader to focus on the points that are relevant to the key message being conveyed.

First you must import your data into R. This typically means that you take data stored in a file, database, or web application programming interface (API), and load it into a data frame in R. If you can’t get your data into R, you can’t do data science on it!

Once you’ve imported your data, it is a good idea to tidy it. Tidying your data means storing it in a consistent form that matches the semantics of the dataset with the way it is stored. In brief, when your data is tidy, each column is a variable, and each row is an observation. Tidy data is important because the consistent structure lets you focus your struggle on questions about the data, not fighting to get the data into the right form for different functions.

Once you have tidy data, a common first step is to transform it. Transformation includes narrowing in on observations of interest (like all people in one city, or all data from the last year), creating new variables that are functions of existing variables (like computing speed from distance and time), and calculating a set of summary statistics (like counts or means). Together, tidying and transforming are called wrangling, because getting your data in a form that's natural to work with often feels like a fight!

Once you have tidy data with the variables you need, there are two main engines of knowledge generation: visualisation and modelling. These have complementary strengths and weaknesses so any real analysis will iterate between them many times.

Visualisation is a fundamentally human activity. A good visualisation will show you things that you did not expect, or raise new questions about the data. A good visualisation might also hint that you're asking the wrong question, or you need to collect different data. Visualisations can surprise you, but don't scale particularly well because they require a human to interpret them.

Models are complementary tools to visualisation. Once you have made your questions sufficiently precise, you can use a model to answer them. Models are a fundamentally mathematical or computational tool, so they generally scale well. Even when they don't, it's usually cheaper to buy more computers than it is to buy more brains! But every model makes assumptions, and by its very nature a model cannot question its own assumptions. That means a model cannot fundamentally surprise you.

The last step of data science is communication, an absolutely critical part of any data analysis project. It doesn't matter how well your models and visualisation have led you to understand the data unless you can also communicate your results to others.

Surrounding all these tools is programming. Programming is a cross-cutting tool that you use in every part of the project. You don't need to be an expert programmer to be a data scientist, but learning more about programming pays off because becoming a better programmer allows you to automate common tasks, and solve new problems with greater ease.

Exploratory data visualization is perhaps the greatest strength of R. One can quickly go from idea to data to plot with a unique balance of flexibility and ease. Many other approaches are available for creating plots in R. In fact, the plotting capabilities that come with a basic installation of R are already quite powerful. There are also other packages for creating graphics such as grid and lattice. We chose to use ggplot2 because it breaks plots into components in a way that permits beginners to create relatively complex and aesthetically pleasing plots using syntax that is intuitive and comparatively easy to remember.

One reason ggplot2 is generally more intuitive for beginners is that it uses a grammar of graphics, the gg in ggplot2. This is analogous to the way learning grammar can help a beginner construct hundreds of different sentences by learning just a handful of verbs, nouns and adjectives without having to memorize each specific sentence. Similarly, by learning a handful of ggplot2 building blocks and its grammar, you will be able to create hundreds of different plots.

Another reason ggplot2 is easy for beginners is that its default behavior is carefully chosen to satisfy the great majority of cases and is visually pleasing. As a result, it is possible to create informative and elegant graphs with relatively simple and readable code.

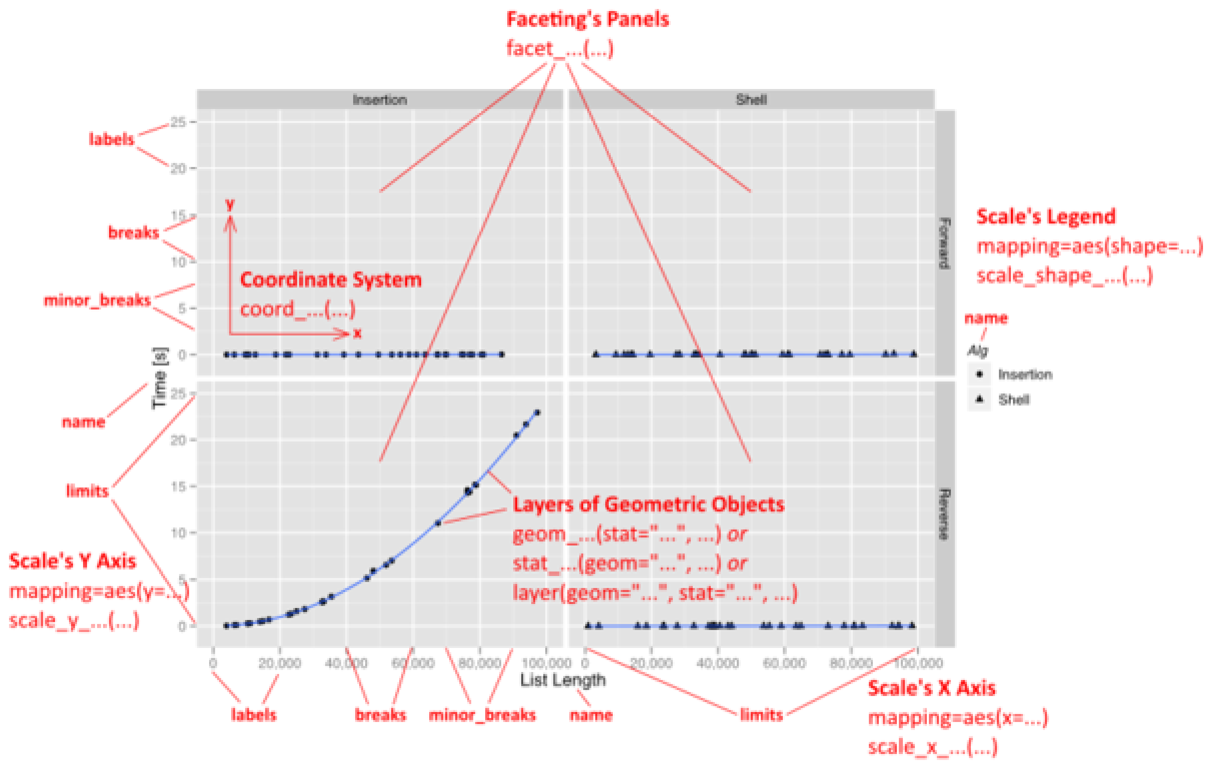
One limitation is that ggplot2 is designed to work exclusively with data tables in tidy format (where rows are observations and columns are variables). However, a substantial percentage of datasets that beginners work with are in, or can be converted into, this format. An advantage of this approach is that, assuming that our data is tidy,ggplot2 simplifies plotting code and the learning of grammar for a variety of plots.

What is the grammar of graphics?

Wilkinson created the grammar of graphics to describe the fundamental features that underlie all statistical graphics. The grammar of graphics is an answer to the question of what is a statistical graphic?ggplot22 builds on Wilkinson's grammar by focussing on the primacy of layers and adapting it for use in R. In brief, the grammar tells us that a graphic maps the data to the aesthetic attributes (colour, shape, size) of geometric objects (points, lines, bars). The plot may also include statistical transformations of the data and information about the plot's coordinate system. Facetting can be used to plot for different subsets of the data. The combination of these independent components are what make up a graphic.

Anatomy of a ggplot

All plots are composed of the data, the information you want to visualise, and a mapping, the description of how the data's variables are mapped to aesthetic attributes. There are five mapping components



A **layer** is a collection of geometric elements and statistical transformations. Geometric elements, geoms for short, represent what you actually see in the plot: points, lines, polygons, etc. Statistical transformations, stats for short, summarise the data: for example, binning and counting observations to create a histogram, or fitting a linear model.

Scales map values in the data space to values in the aesthetic space. This includes the use of colour, shape or size. Scales also draw the legend and axes, which make it possible to read the original data values from the plot (an inverse mapping).

A **coord**, or coordinate system, describes how data coordinates are mapped to the plane of the graphic. It also provides axes and gridlines to help read the graph. We normally use the Cartesian coordinate system, but a number of others are available, including polar coordinates and map projections.

A **facet** specifies how to break up and display subsets of data as small multiples. This is also known as conditioning or latticing/trellising.

A **theme** controls the finer points of display, like the font size and background colour. While the defaults in ggplot2 have been chosen with care, you may need to consult other references to create an attractive plot.

SCATTERPLOTS

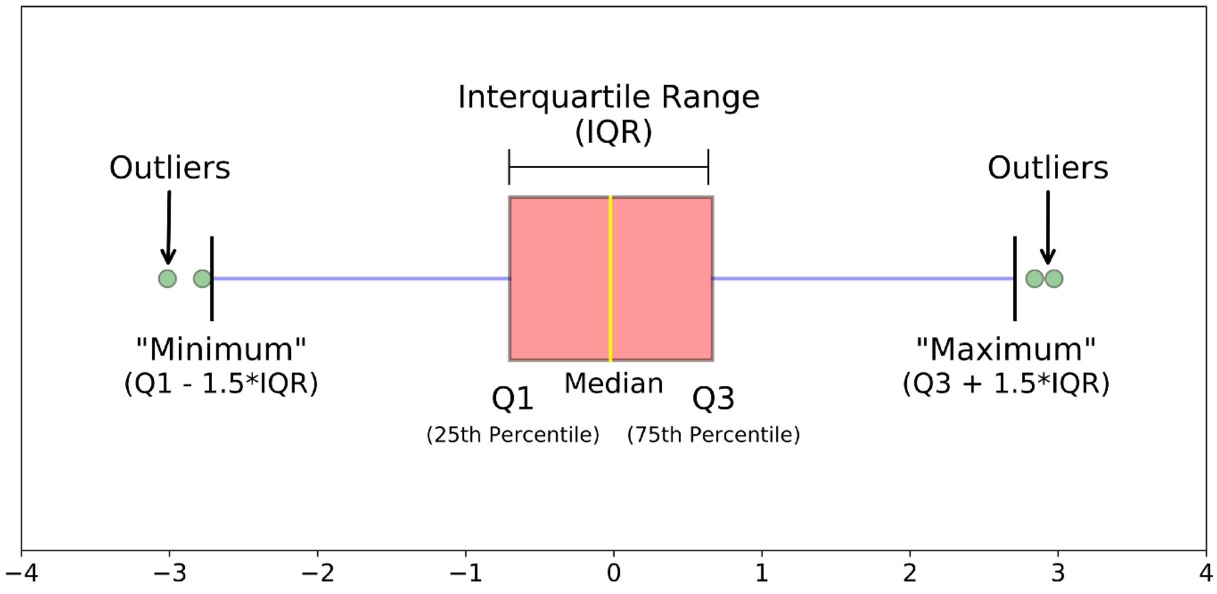
A scatter plot is a chart type that is normally used to observe and visually display the relationship between variables. The values of the variables are represented by dots. Scatter plots are also known as scattergrams, scatter graphs, or scatter charts.

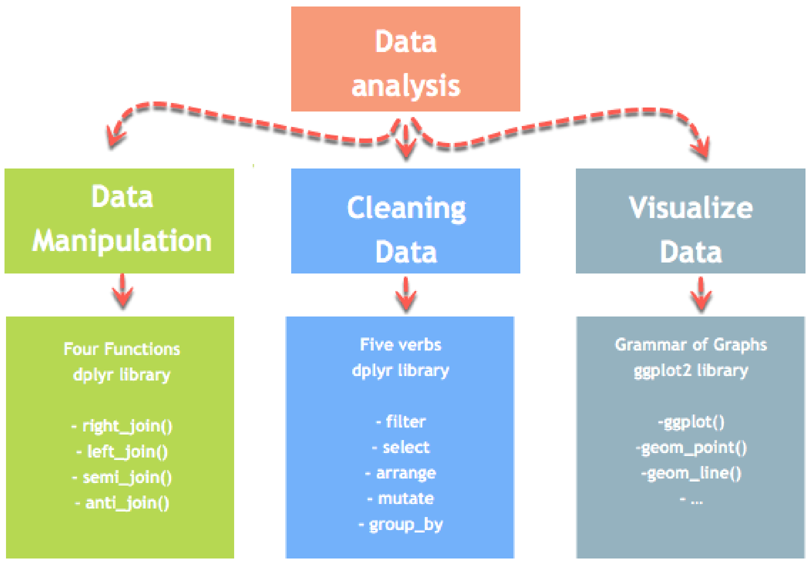
Boxplots are a standardized way of displaying the distribution of data based on a five number summary median (Q2/50th Percentile): the middle value of the dataset.

- first quartile (Q1/25th Percentile): the middle number between the smallest number (not the "minimum") and the median of the dataset.
- third quartile (Q3/75th Percentile): the middle value between the median and the highest value (not the "maximum") of the dataset.
- interquartile range (IQR): 25th to the 75th percentile.
- whiskers (shown in blue)
- outliers (shown as green circles)

"maximum": Q3 + 1.5*IQR

"minimum": Q1 -1.5*IQR





The package dplyr provide easy tools for the most common data manipulation tasks. It is built to work directly with data frames.

We're going to learn some of the most common dplyr functions: select(), filter(), mutate(), arrange(), and summarize().

DOT PLOT

Dot Plot is a graph for displaying the distribution of quantitative variable where each dot represents a value.

VIOLIN PLOT

A violin plot depicts distributions of numeric data for one or more groups using density curves. The width of each curve corresponds with the approximate frequency of data points in each region.

HISTOGRAM

A histogram contains rectangular area to display the statistical information which is proportional to the frequency of a variable and its width in successive numerical intervals.

DENSITY PLOT

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.

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.

PIE CHART

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

RIDGELINE PLOT

A Ridgeline plot (sometimes called Joyplot) shows the distribution of a numeric value for several groups. Distribution can be represented using histograms or density plots, all aligned to the same horizontal scale and presented with a slight overlap.

6. Case studies in Life Science

Sona Charles, Scientist (Ag. Bioinformatics)
ICAR-IISR, Calicut
Email: sona.charles@icar.gov.in

VOLCANO PLOT

Volcano plots are commonly used to display the results of RNA-seq or other omics experiments. A volcano plot is a type of scatterplot that shows statistical significance (P value) versus magnitude of change (fold change). It enables quick visual identification of genes with large fold changes that are also statistically significant. These may be the most biologically significant genes. In a volcano plot, the most upregulated genes are towards the right, the most downregulated genes are towards the left, and the most statistically significant genes are towards the top.

To generate a volcano plot of RNA-seq results, we need a file of differentially expressed results, a sample of which is provided in this workshop.

CIRCOS PLOT/ IDIOGRAM

Circular layout is very useful to represent complicated information. First, it elegantly represents information with long axes or a large amount of categories; second, it intuitively shows data with multiple tracks focusing on the same object; third, it easily demonstrates relations between elements. It provides an efficient way to arrange information on the circle and it is beautiful.

Circular visualization is popular in Genomics and related omics fields. It is efficient in revealing associations in high dimensional genomic data. In genomic plots, categories are usually chromosomes and data on x axes are genomic positions, but it can also be any kind of general genomic categories.

To make is easy for Genomics analysis, circlize package particularly provides functions which focus on genomic plots. Genomic data is usually stored as a table where the first three columns define the genomic regions and following columns are values associated with the corresponding regions. Each genomic region is composed by three elements: genomic category (in most case, it is the chromosome), start position on the genomic category and the end position. Such data structure is known as BED format and is broadly used in genomic research.

HEATMAP

Heatmap is a powerful tool for the visual display of microarray data or data from next-generation sequencing studies such as microbiome analysis. Heatmap is a graphical representation of data that uses a system of color-coding in representing different values contained in a matrix. As early as the 19th century, heatmaps were used in statistical analysis and progressed in 2008 as a useful tool for almost every field such as engineering, medicine, and even in research. Heatmap is also user-friendly, more importantly to those who are not accustomed to reading large quantities of data since it is more visually accessible than traditional data formats. Heatmap is considered a useful tool because it can provide a comprehensive overview as its data visualization tools are easy to understand and are often self-explanatory. It is a lot different from a table or chart which both need to be interpreted or studied to be understood.

DENDROGRAM

A dendrogram (or tree diagram) is a network structure. It is constituted of a root node that gives birth to several nodes connected by edges or branches. The last nodes of the hierarchy are called leaves. Many options are available to build one with R.