R Statistics Essential Training

Trainer: Dr. Ravi Kumar Tiwari

TINFOTECH

Website:www.tertiarycourses.com.sg
Email: enquiry@tertiaryinfotech.com

Agenda

Module 1. Getting Started

- What is R
- Install R and RStudio
- Explore RStudio Interface
- Variables

Module 2. Data Types

- Numbers
- Text
- Vector
- Matrix
- Array
- Data Frame
- Factor
- List

Agenda

Module 3. Packages & Data Sets

- Packages
- Data Sets

Module 4. Data Input/Output

- Read data from file
- Read data from web
- Write data to file

Module 5. Charts

- Scatter Plot
- Boxplot
- Bar chart
- Pie chart
- Histogram

Agenda

Module 6. Control Structures

- Conditional
- Loop
- Break & Next
- Operators

Module 7: Function

- Function Syntax
- Function Example
- Function With Default Arguments

Agenda

Module 8. Statistical Applications of R

- Basic Statistics
- Correlation
- Linear Regression
- Multiple Regression
- 2 Sample T-Test
- 1 Sample T-Test
- ANOVA
- Clustering

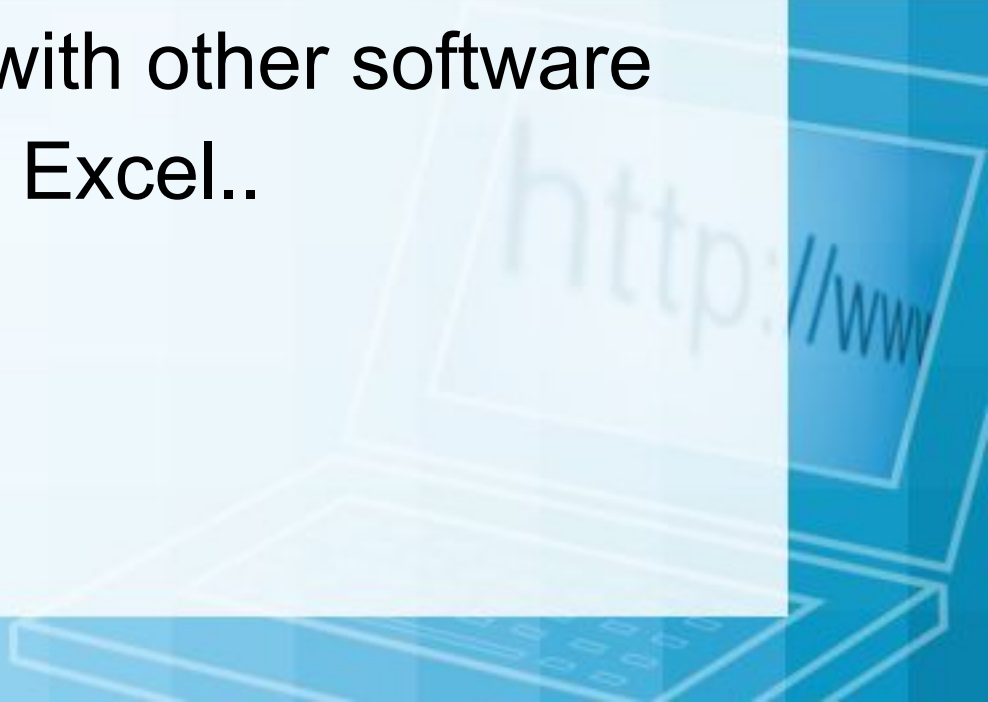
Module 1
Getting Started

What is R?

- R was developed by Ross Ihaka and Robert Gentleman
- R is a language and environment for statistical computing and graphics
- R was developed based on S

Why R?

- R is open source and free!
- R is vector-based
- R is a programming language meant for statistics
- R can be integrated with other software such as SAS, SPSS, Excel..



R Development History

Install R

http://cran.stat.nus.edu.sg/

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

The Comprehensive R Archive Network

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
- Download R for (Mac) OS X
- 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 (2015-12-10, Wooden Christmas-Tree) R-3.2.3.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

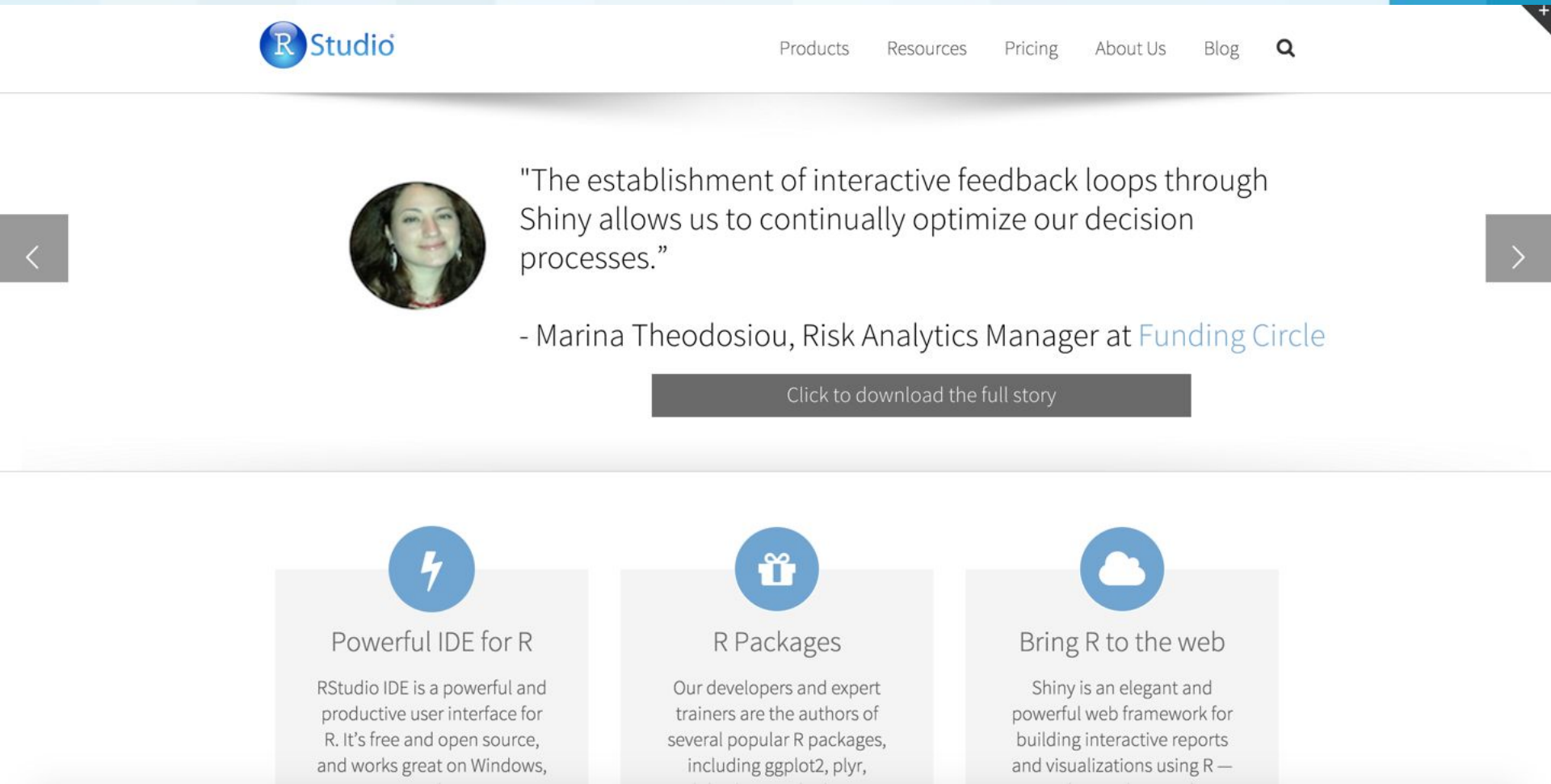
- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

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.

Install R Studio IDE

https://www.rstudio.com/



Explore RStudio Interface

RStudio File Edit Code View Plots Session Build Debug Tools Window Help — 38% Mon 11 Jan 10:16 PM Alfred Ang

RStudio

Untitled2* | Untitled3* | Ex02_03.R*

Source on Save | Run | Source

```
1  data(lynx)  # Annual Canadian Lynx trappings 1821-1934
2  hist(lynx)
3
4  # Modify histogram
5  h <- hist(lynx,  # Save histogram as object
6            breaks = 13,  # "Suggests" 11 bins
7  #          breaks = seq(0, 7000, by = 100),
8  #          breaks = c(0, [illegible], 7000),
9  #          freq = FALSE,
10           col = "thistle", # Use color() [626]
11           main = "Histogram of Annual Canadian Lynx Trappings\n1821-1934",
12           xlab = "Number of Lynx Trapped")
13
14 rm(list = ls())  # Clean up
```

SCRIPT

1:1 (Top Level) — R Script

Environment | History

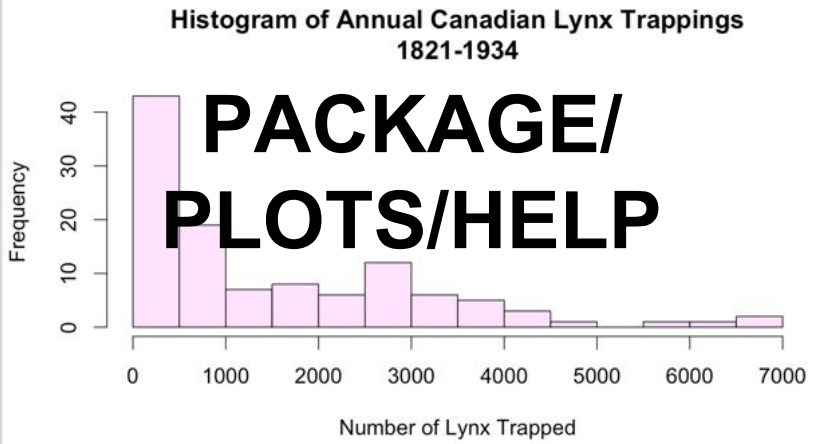
Import Dataset | List

Global Environment

WORKSPACE

Files | Plots | Packages | Help | Viewer

Zoom | Export | Publish



PACKAGE/
PLOTS/HELP

Console ~/

```
> rm(list = ls())  # Clean up
> lynx
Time Series:
Start = 1821
End = 1934
Frequency = 1
  [1]  269  321  585  871 1475 2821 3928 5943 4950 2577  523   98  184  279  409 2285 2685 3409 1824  409
 [21]  151   45   68  213  546 1033 2129 2536  957  361  377  225  360  731 1638 2725 2871 2119  684  299
 [41]  236  245  552 1623 3311 6721 4254  687  255  473  358  784 1594 1676 2251 1426  756  299  201  229
 [61]  469  736 2042 2811 4431 2511  389   73   39   49   59  188  377 1292 4031 3495  587  105  153  387
 [81]  758 1307 3465 6991 6313 3794 1836  345  382  808 1388 2713 3800 3091 2985 3790  674   81   80  108
[101]  229  399 1132 2432 3574 2935 1537  529  485  662 1000 1590 2657 3396
> ?lynx
> #
```

CONSOLE

Comments and Help

Single line comment

#

Help

?....

help(...)

Variables

a <- 1 (Most common)

a = 1

1 -> a

List all variables

ls()

Clean Up the Variables

```
rm(x)
rm(a, b)
rm(list = ls())
```

Module 2
Data Types

Data Types

- Numbers
- Text
- Vector
- Matrix
- Array
- Data Frame
- Factor
- List

Numbers

Working With Numbers

2^4

abs(-3.2)

round(3.4), round(5.24, digits = 1), round(5.24, 1)

sqrt(4)

cos(pi/2), factorial(3)

3/0

Inf/Inf

NaN : Not a Number

Text

Splitting Text

a <- "Today is a good day"

Splitting & Joining Text

```
a <- "Today is a good day"
strsplit(a," ")
strsplit(a,"is ")

a<-"angch"
b<-"tertiaryinfotech.com"
paste(a,b,sep="@")
```

Sorting Text

```
v <- c("Red","Blue","yellow","violet")
sort(v)
sort(v, decreasing=TRUE)
```

Vector

Create Vector Using : Operator

```
a <- 0:10
a <- 5:13
a <- 10:-4
```

```
class(a)    - check the type of a
str(a)      - check the structure of a
```

Create Vector Using c Operator

```
c(1,2,4)
c(1,7:9)
c('red','green',"yellow")
c(1:5, 10.5, "red")
```

Create Vector Using seq

```
seq(10)
seq(3,20)
seq(3,20,3)
seq(1, 9, by = 2)
seq(from=4.5, to=2.5, by=-0.2)
seq(0, 1, length.out = 11)
```

Application of seq

```
x<-seq(0,4*pi,length.out = 200);
y <- sin(x);
plot(x,y)
```

Creating Vectors Using rep -1

```
rep(3,10)
rep(1:3, 3)
rep(c(1,2,3),3)
rep(seq(3),3)
rep(seq(3),length.out=5)
rep(seq(3),len=5)
rep(1:4, each = 2)
rep(1:4, c(2,1,2,1))
```

Creating Vectors Using rep -2

```
rep(1:4, each = 2, len = 4)
rep(1:4, each = 2, len = 10)
rep(1:4, each = 2, times = 3)
```

Accessing Vector Elements

a[5]

a[5:8]

a[5:3]

a[-2] - take away the 2nd element

a[-1:-4]

a[c(5, 6, 7, 8)]

Tail and Head

tail(a)

tail(a,n=3)

tail(a,3)

head(a)

head(a,n=3)

head(a,3)

Logical Indexing

c(1,2,3) = c(3,2,1)

1 %in% c(3,4,5)

c(1,2) %in% c(1,2,3,4)

Challenge: Logical Indexing

a <-c(2,3,-1,3,5,2,-3,1)

find the sum of the positive elements of a

Hint: a>0

Dropping NA values

```
a<-c(3,-2,4,NA,-1,8,-4,9,NA, 11,3,8,NA);
a[!is.na(a)]
```

NA: Missing data

Manipulating Vectors

```
x <- c(1,2,1)
y <- c(3,2,4)
v <- x + y
v <- x - y
```

Vector Arithmetic

a <-c(2,3,4,2,5,6)

mean(a)

median(a)

sum(a), prod(a)

min(a). max(a)

cummin(a), cummax(a)

cumsum(a), cumprod(a)

diff(a)

Sorting Vector Elements

```
v <- c(3,8,4,5,0,11, -9, 304)
sort(v)
sort(v, decreasing=TRUE)
```

Sum Vector Elements

```
v <- c(3,6,2,NA,1)
sum(v)
sum(v, na.rm=TRUE)
sum(a[!is.na(a)])
```

Structure of Vectors

```
v <- c(3,8,4,5,0,11, -9, 304)
v <- c("Red","Blue","yellow","violet")


str(v)
```

Matrix

Matrix

Matrices are the R objects in which the elements are arranged in a two-dimensional rectangular layout.

Syntax:

matrix(data, nrow, ncol, byrow = FALSE, dimnames)

Creating Matrix

Usage: matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)

```
matrix(1:12, ncol=4)
matrix(1:12, nrow=4)
matrix(c(3, 9, -1, 4, 2, 6), nrow=2)
matrix(1:12, ncol=4,byrow=TRUE)
matrix(1:12, nrow=4,byrow=TRUE)
```

Convert Vector to Matrix

```
a <- c(1,1,1,1)

dim(a)<-c(2,2)
```

Combining Vectors to Matrix

```
a1 <- c(1,1,1,1)
a2 <- c(2,2,2,2)
rbind(a1,a2)
cbind(a1,a2)
```

Accessing Matrix Elements

M[1,3] - row 1, col 3

M[2,] - row 2

M[,3] - col 3

M[1:2,3:4] - row 1 to 2, col 3 to 4

Challenge: Matrix Elements

a <- matrix(1:20,ncol=4). Extract the elements in red

	[,1]	[,2]	[,3]	[,4]
[1,]	1	6	11	16
[2,]	2	7	12	17
[3,]	3	8	13	18
[4,]	4	9	14	19
[5,]	5	10	15	20

Time: 5 mins

Manipulating Matrix Elements

```
M1 <- matrix(c(3, 9, -1, 4, 2, 6), nrow=2)
M2 <- matrix(c(5, 2, 0, 9, 3, 4), nrow=2)
M3 <- M1 + M2
M4 <- M1 + M2
```

Row and Column Names

```
matrix(c(3, 9, -1, 4, 2, 6), nrow=2)
rownames(M)<-c("row1","row1")
colnames(M)<-c("col1","col2","col3")

M["row2",]
M[,"col3"]
```

Matrix Arithmetic

rowSums(M) - sum of each row

colSums(M) - sum of each col

colMeans(M) - mean of each col

t(M1) - transpose

solve(M1) - invert

* - element-wise multiple

%*% - dot product/matrix muplitcation

Array

What is Array?

Arrays are the R data objects which can store data in more than two dimensions

Creating Array

```
a <- array(c(11:14,21:24,31:34),dim=c(2,2,3))
```

Accessing Array Elements

a[,,1] - 1st matrix

a[1,,1] - 1st row, 1st matrix

Data Frame

Data Frame

A data frame is a table or a two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column.

- 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 same number of data items.

Creating Data Frame

```
a <- data.frame(
        gender = c("Male", "Male","Female"),
        height = c(152, 171.5, 165),
        weight = c(81,93, 78),
        age =c(42,38,26),
        row.names=c('Ally','Belinda','Alfred')
        )
```

Creating DF from Vectors

```
gender  <- c('Female','Female','Male')
height  <- c(162,169,170)
weight  <- c(40,50,60)
age     <- c(21,22,23)
name    <- c('Ally','Belinda','Alfred')

a <- data.frame(gender, height, weight, age,
row.names= name)
```

Columns & Rows Names

rownames(x)

colnames(x) or names(x)

Rows & Column Numbers

nrow(x) : Number of Rows

ncol(x) : Number of Columns

Subset Operators

$: Select a single component from the data

[[: Select single component by name or position

[: Select multiple components

Accessing Single Component

Using $ operator

x$gender

x$height

Using [[operators]]

x[["gender"]]

x[[1]]

Accessing Multiple Components

Using [operator

a[1]

a["gender"]

a[-2]

a[1:2,]

a[c(1,4)]

a[,2]

a[c(2,3),c(1,2)]

Filter Data using subset

```
subset(a, select = c("gender", "age"))
subset(a, subset = height > 163, select = c("gender", "height", "age"))
```

Add Column to Data Frame

a$name =c('Ally','Belinda','Jane')

Add Rows to Data Frame

```
b <- data.frame(
        gender = c("Female", "Male","Female"),
        height = c(155, 171.5, 155),
        weight = c(71,93, 68),
        age =c(42,38,26),
        row.names =c('SC','Alfred','TC')
        )

c <- rbind(a,b)
```

Summary of Data Frame

summary(a)

Factor

Factor

Factors are discrete data such as colors, countries, digits

```
a <- factor(c("east","east","west","south"))
a <- factor(rep(letters[1:3],each=4))
```

List

List

Lists are the R objects which contain elements of different types like − numbers, strings, vectors and another list inside it.

Unnamed List

```
a  <-list(1:10, matrix(1:4,ncol = 2),"r" )
```

Named List

List groups different objects

```
a  <-list('A'=1:10, 'B'=matrix(1:4,ncol = 2),'C'="r"
)
```

Accessing List Element

Using [[ ]] operator

a[[1]]

a[['A']]

Using [ ] operator

a[1]

a['A']

Using $ operator

a$A

Modifying List Element

list1[[1]] = c("d","e","f")

list1$'letters' - c("aa","bb","cc")

Merging Lists

```
list1 <- list(1,2,3)
list2 <- list("Sun","Mon","Tue")


merged.list <- c(list1,list2)
```

Converting List to Vector

```
v1 <- unlist(list1)
```

Date

Create a Date

```
xd <-as.Date("2016-03-13")
xd <-as.Date("5 Aug 2016",format="%d %b %Y")
```

%Y : Year with century
%y : Year without century
%m : Month in decimal
%B : Full month name
%b : Abbreviated month name
%d : Day in decimal

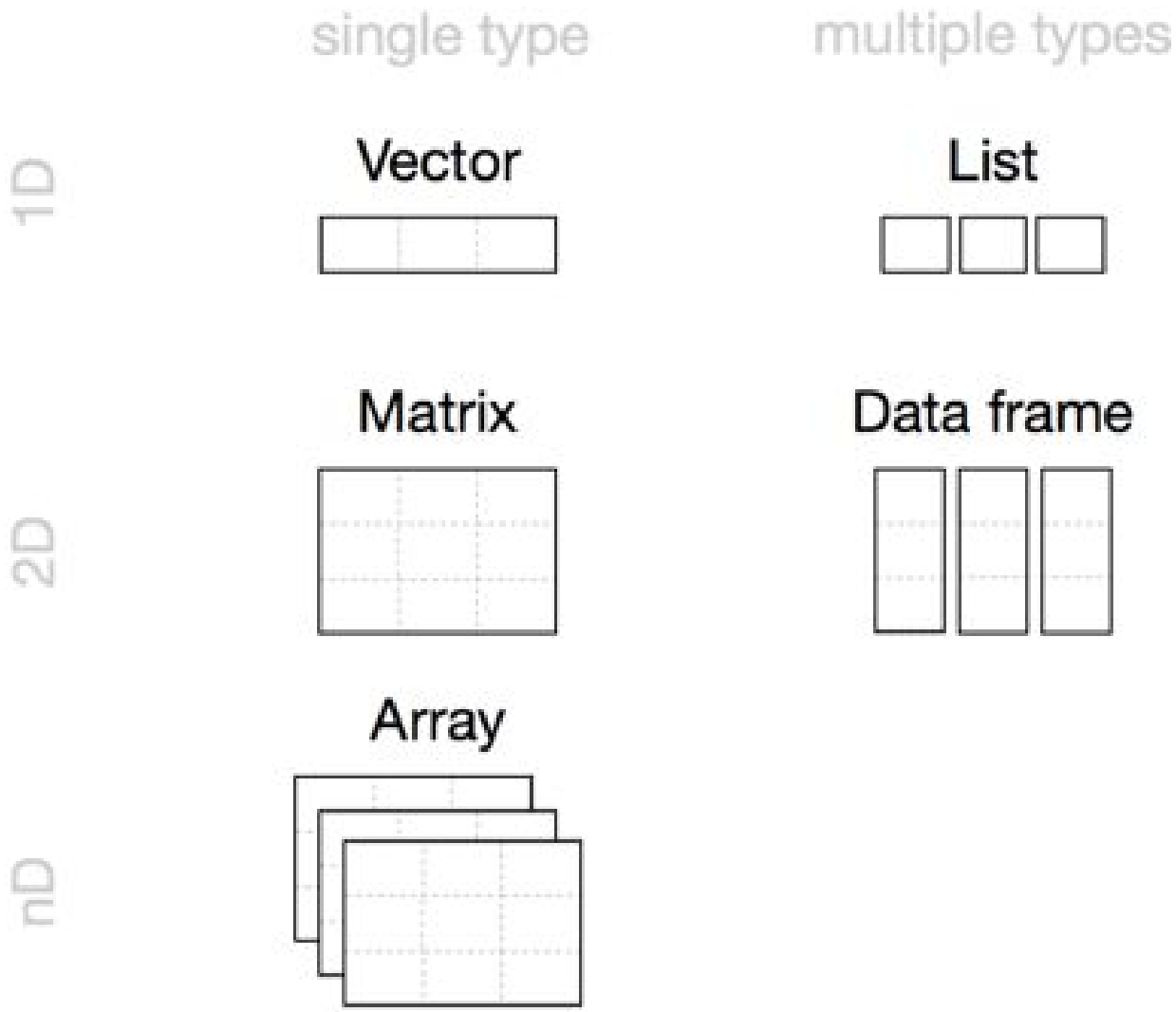
Working with Date

weekdays(xd)

months(xd)

xd+7

Data Types Summary



Module 3
Packages & Data Sets

Packages

R Packages

https://cran.r-project.org/web/views/

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
Finance	Empirical Finance
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data
Survival	Survival Analysis
TimeSeries	Time Series Analysis

Install and Load a Package

1. Install package

```
install.packages("ggplot2")
```

2. Load a package

```
library("ggplot2")
```

Update Packages

update.packages()

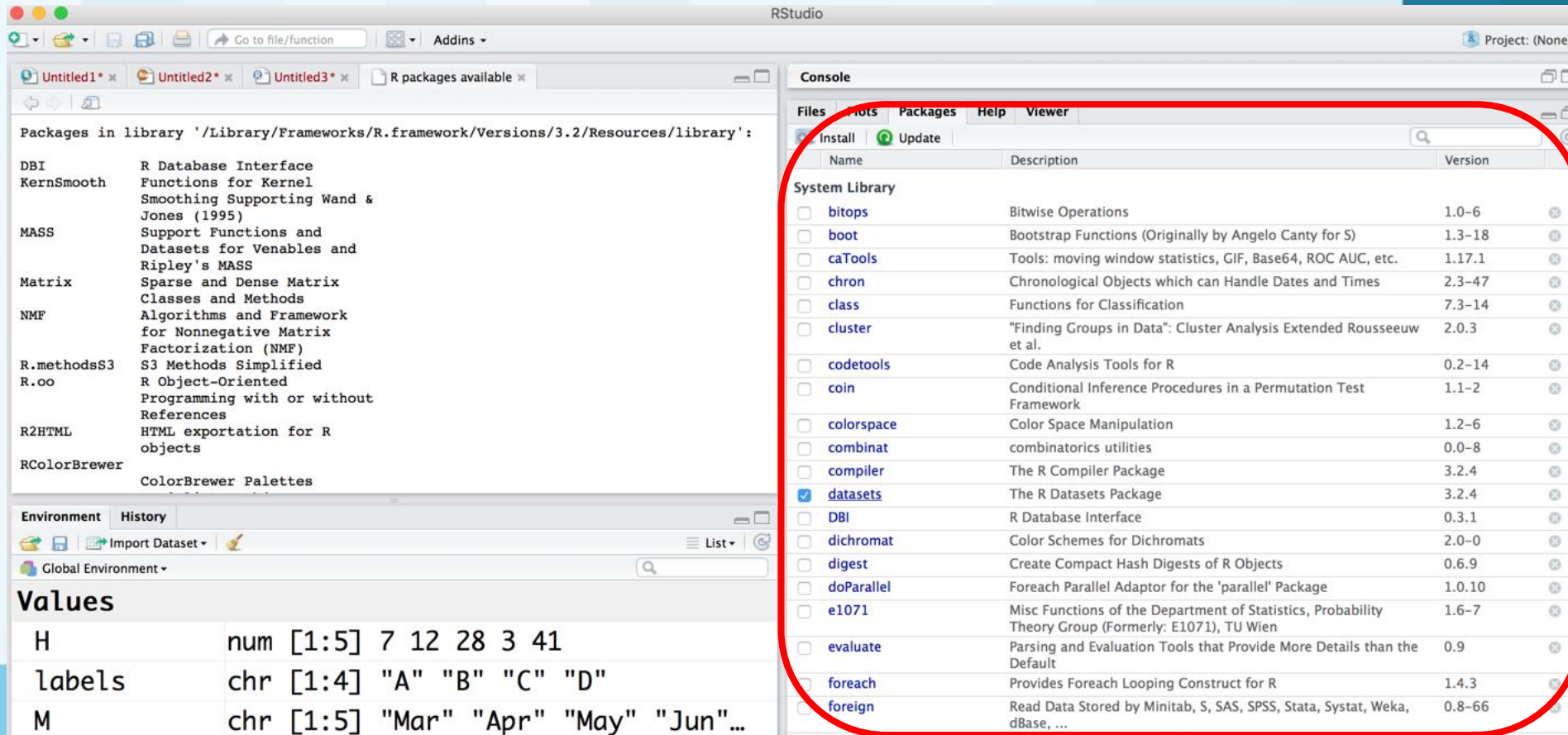
Unload and Remove Package

```
detach("package:ggplot2", unload = TRUE)

remove.packages("ggplot2")
```

RStudio Package Management

install and load package in RStudio

unload and remove a package in RStudio



Data Sets

R Dataset Packages

http://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html

The R Datasets Package

Documentation for package 'datasets' version 3.3.0

- DESCRIPTION file.

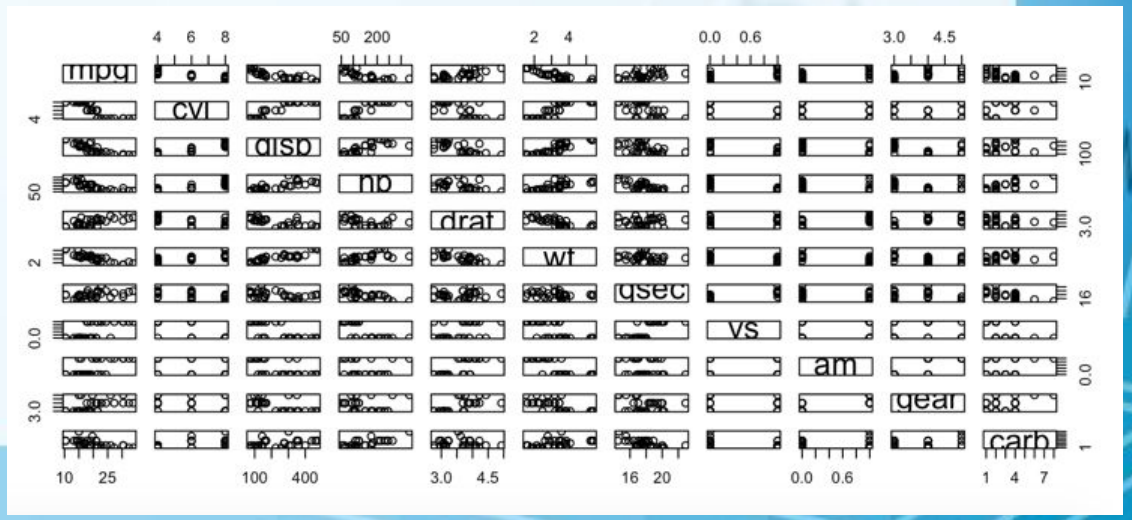
Help Pages

A B C D E F H I J L M N O P Q R S T U V W

datasets-package	The R Datasets Package
	-- A --
ability.cov	Ability and Intelligence Tests
airmiles	Passenger Miles on Commercial US Airlines, 1937-1960
AirPassengers	Monthly Airline Passenger Numbers 1949-1960
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
	-- B --
beaver1	Body Temperature Series of Two Beavers
beaver2	Body Temperature Series of Two Beavers
beavers	Body Temperature Series of Two Beavers
BJsales	Sales Data with Leading Indicator

MTCar Dataset

- The data was extracted from the 1974 Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973–74 models).
- A data frame with 32 observations on 11 variables.




```
data(mtcars)
str(mtcars)
```

MTCar Dataset Variables

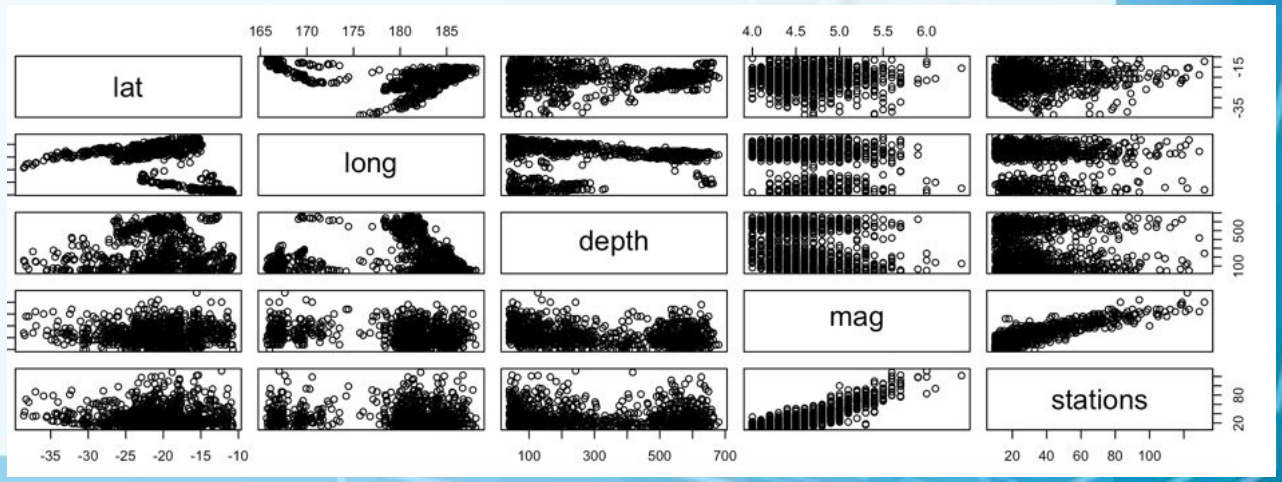
- mpg - Miles/(US) gallon
- cyl - Number of cylinders
- disp - Displacement (cu.in.)
- hp - Gross horsepower
- drat - Rear axle ratio
- wt - Weight (1000 lbs)
- qsec - 1/4 mile time
- vs - V/S
- am - Transmission
- gear - Number of forward gears
- carb - Number of carburetors

Quake Dataset

- The data set give the locations of 1000 seismic events of MB > 4.0. The events occurred in a cube near Fiji since 1964.
- A data frame with 1000 observations on 5 variables.
- lat, long, depth, mag, stations

data(quakes)

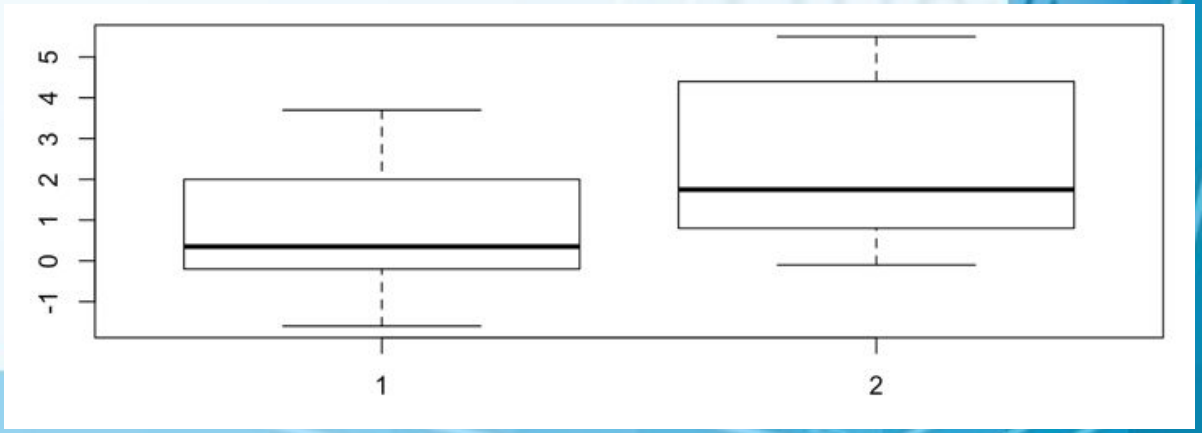
str(quakes)



Sleep Dataset

- Data which show the effect of two soporific drugs (increase in hours of sleep compared to control) on 10 patients.
- A data frame with 20 observations on 3 variables.
- extra, group, ID

```
data(sleep)
str(sleep)
```

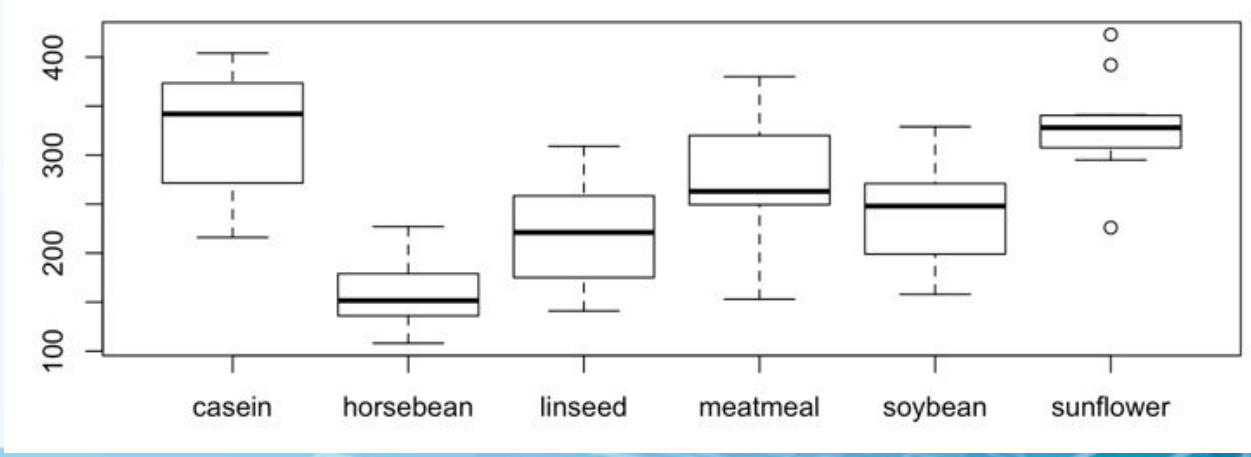


Chickwts Dataset

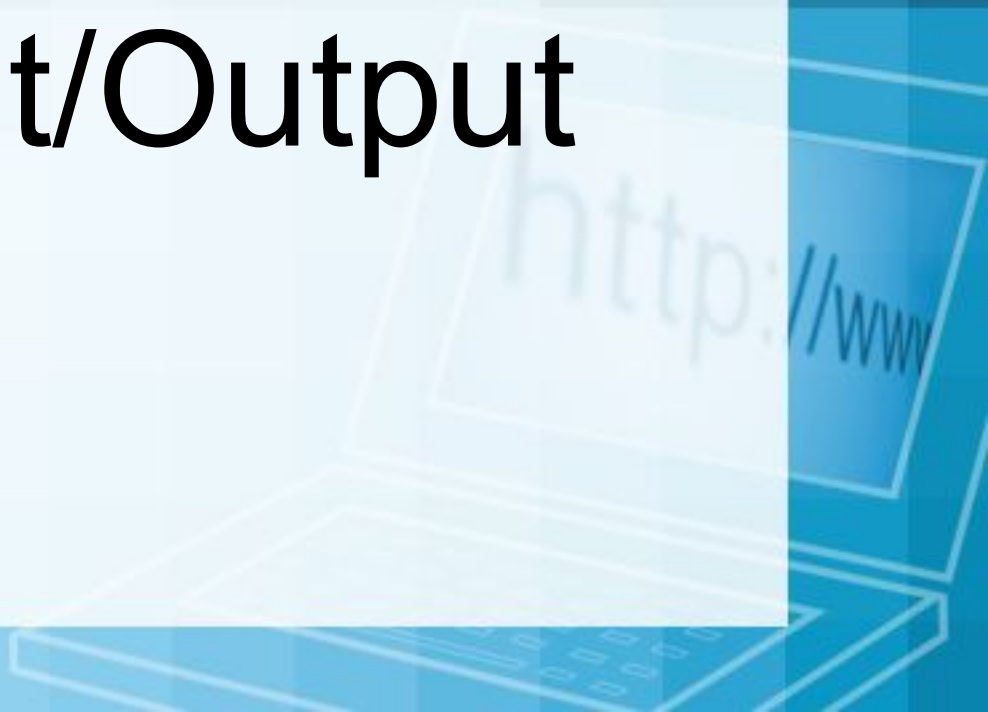
- An experiment was conducted to measure and compare the effectiveness of various feed supplements on the growth rate of chickens.
- A data frame with 71 observations on the following 2 variables.
- weight, feed

data(chickwts)

str(chickwts)



Module 4
Data Input/Output



Working Directory

1. Get current working directory

getwd()

2. Set current working directory

setwd("...")

3. Add function

source("...")

Read Data

Read Data from CSV File

```
a <- read.csv("input.csv", header = TRUE)

View(a)
```

Ex:csvimport.R

Read Data from Web

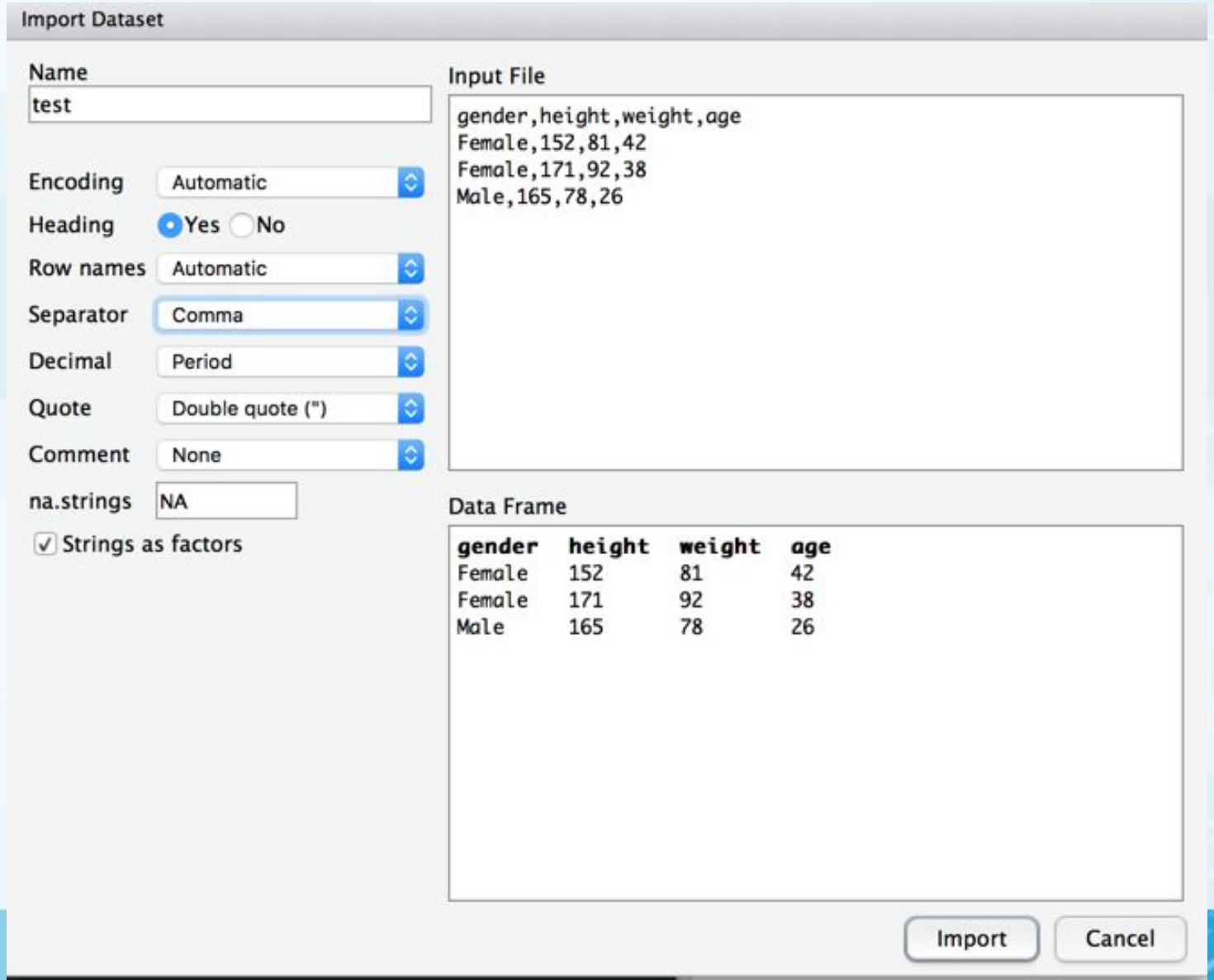
```
url<-"http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data"
read.csv(url, nrows=5, header = FALSE)
```

Read Data from Text File

```
a <- read.table("file.txt", header = TRUE, sep
= "\t")
```

Ex:readwritetext.R

Import Dataset in RStudio



Write Data

Write CSV File

```
write.csv(a,"output.csv")
```

Write Text File

```
write.table(a1, "file.txt", sep="\t")
```

Ex:readwritetext.R

Ex: CSV Read/Write

1. Read a data set from a csv file
2. Add one more row of data
3. Write the data back to a csv file

Time: 5 min

Module 5
Charts

Scatter Plot

Plot Syntax

plot(x,y,type,col,xlab,ylab….)

col - color

main - title

xlab - x label

ylab - y label

type - type of plots

.

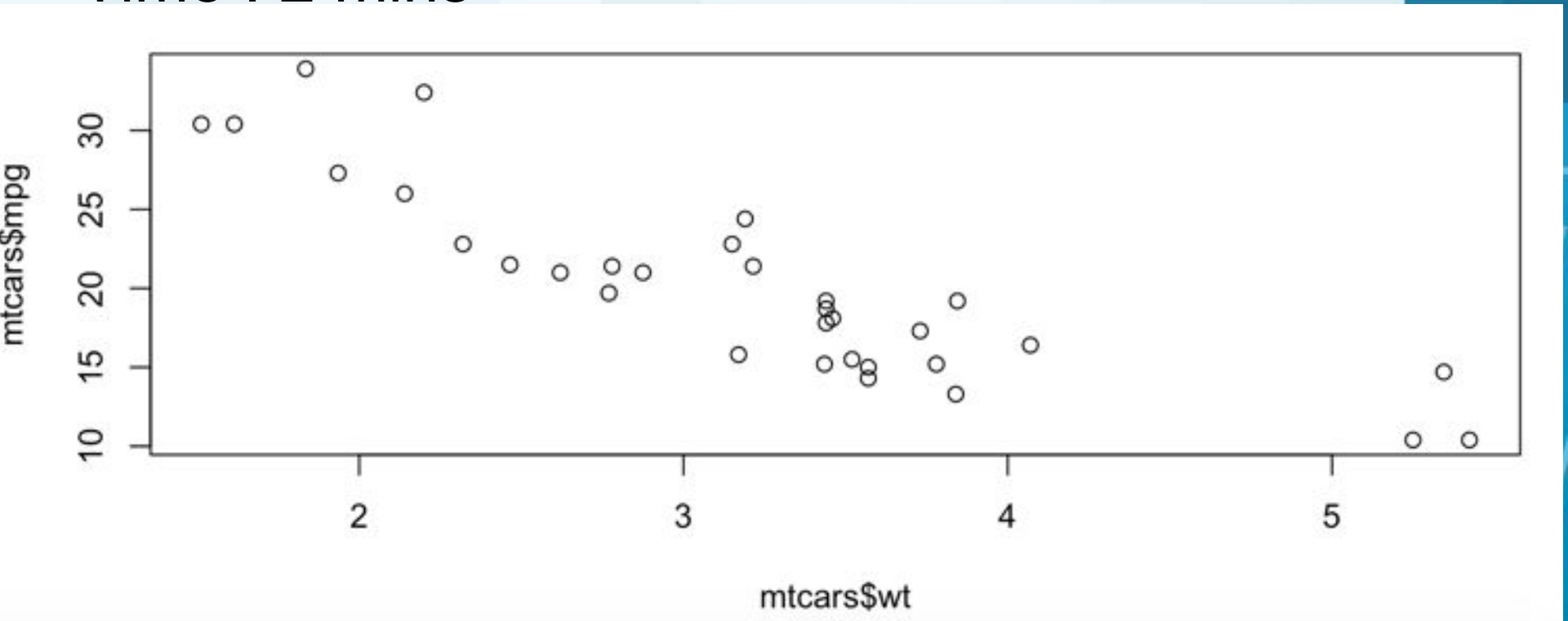
Plot Example

```
x<-seq(-10,10)
y<-x^2
plot(x,y)

plot(x,y,'o')
plot(x,y,'b')
plot(x,y,'p')
```

Ex: Plot for MTCars

Plot the mtcars mpg vs mtcars wt

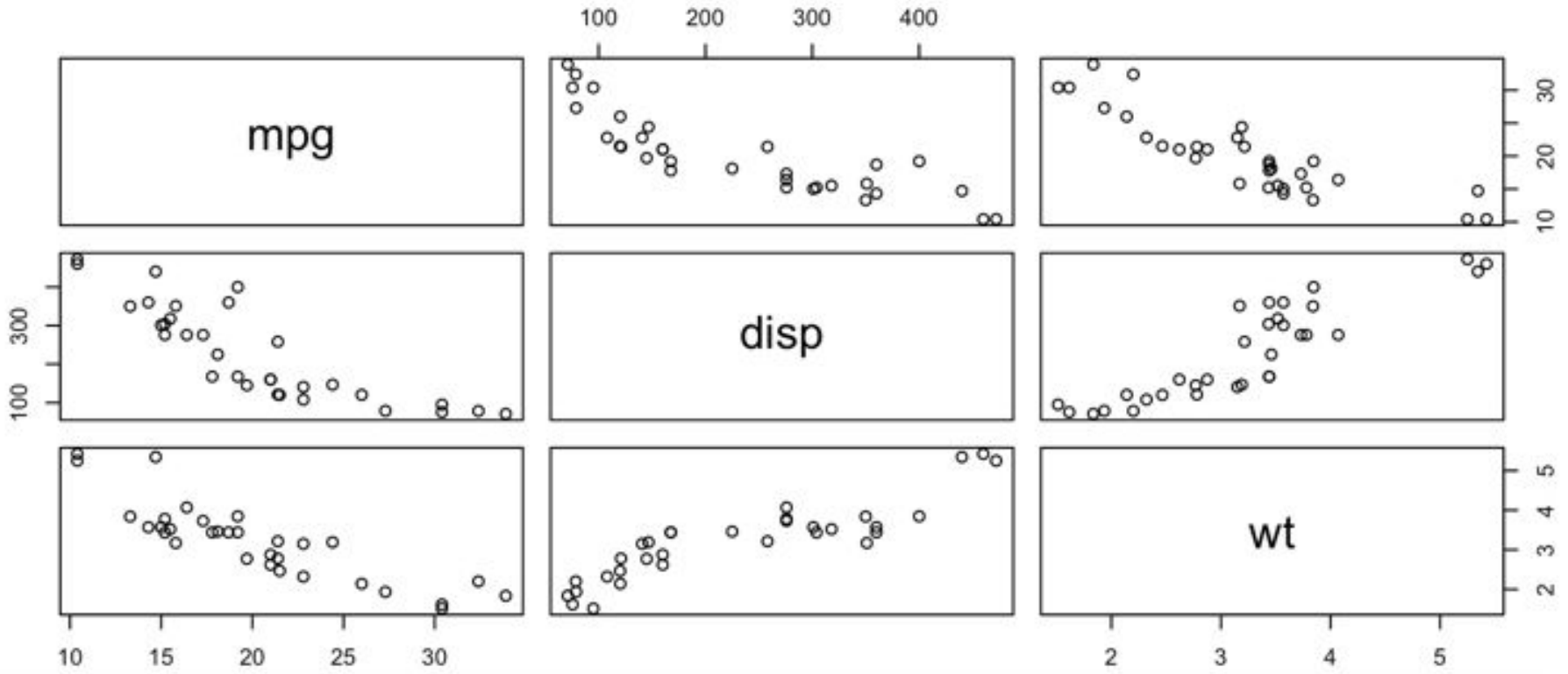
Time : 2 mins



Multiple Scatter Plots

plot(mtcars)

plot(mtcars[,c(1,3,6)])



Box Plot

Boxplot Syntax

boxplot(formula,...)

notch - Draw a notch.

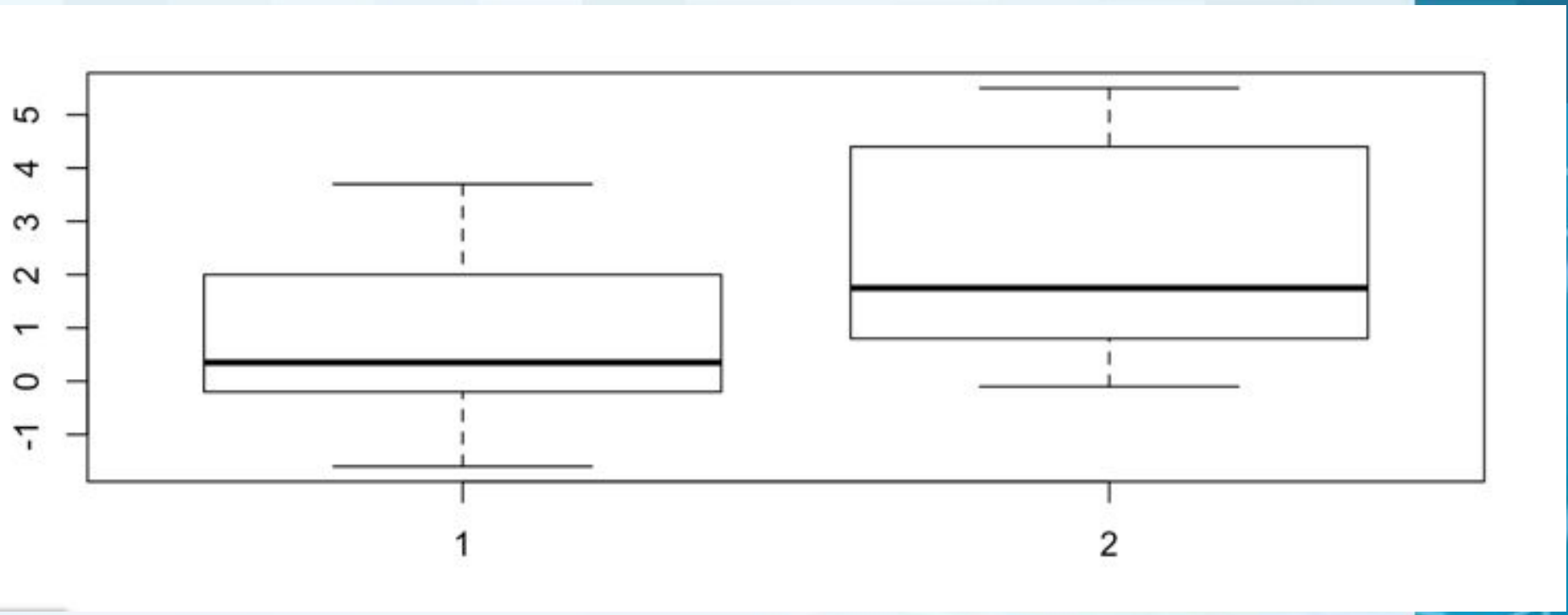
varwidth - Variable box width

names - Group labels

main - Title

Boxplot Example - Sleep

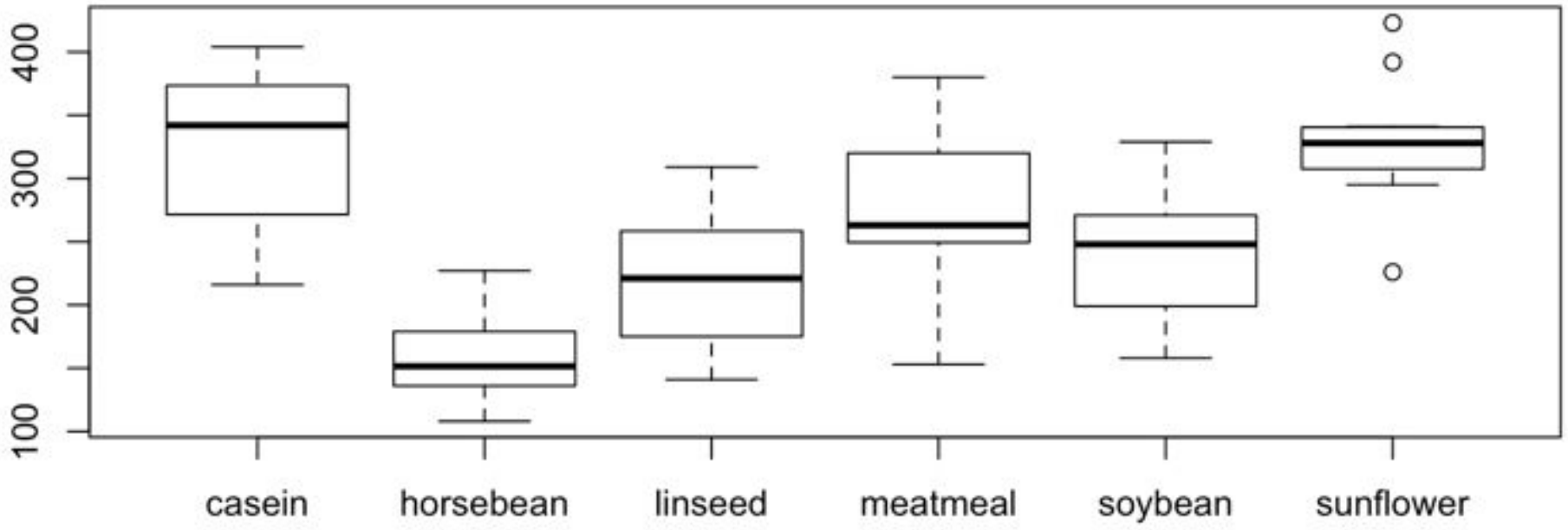
boxplot(extra ~ group, data = sleep)



Ex: Boxplot - Chickwts

Plot the chickwts weight vs feed

Time : 2 mins



Bar Chart

Bar Chart Syntax

barplot(H,xlab,ylab,main, names.arg,col)

H - a vector or matrix

xlab - label for x axis.

ylab - label for y axis.

main - title of the bar chart.

names.arg - vector of names appearing under each bar.

col - colors to the bars in the graph.

Bar Chart Example

```
a <-c(4,5,2,1)
barplot(a)
barplot(a,col="yellow")
```

Pie Chart

Pie Chart Syntax

pie(x, labels, radius, main, col, clockwise)

x - vector

labels - description to the slices.

radius - radius (value between -1 and +1).

main - title of the chart.

col - color palette.

clockwise - drawn clockwise or anti clockwise.

Pie Chart

```
a <-c(4,5,2,1)
pie(a)
```

Histogram

Histogram Syntax

hist(v,main,xlab,xlim,ylim,breaks,col,border)

v - a vector

main - title of the chart.

col - set color of the bars.

border - set border color of each bar.

xlab - description of x-axis.

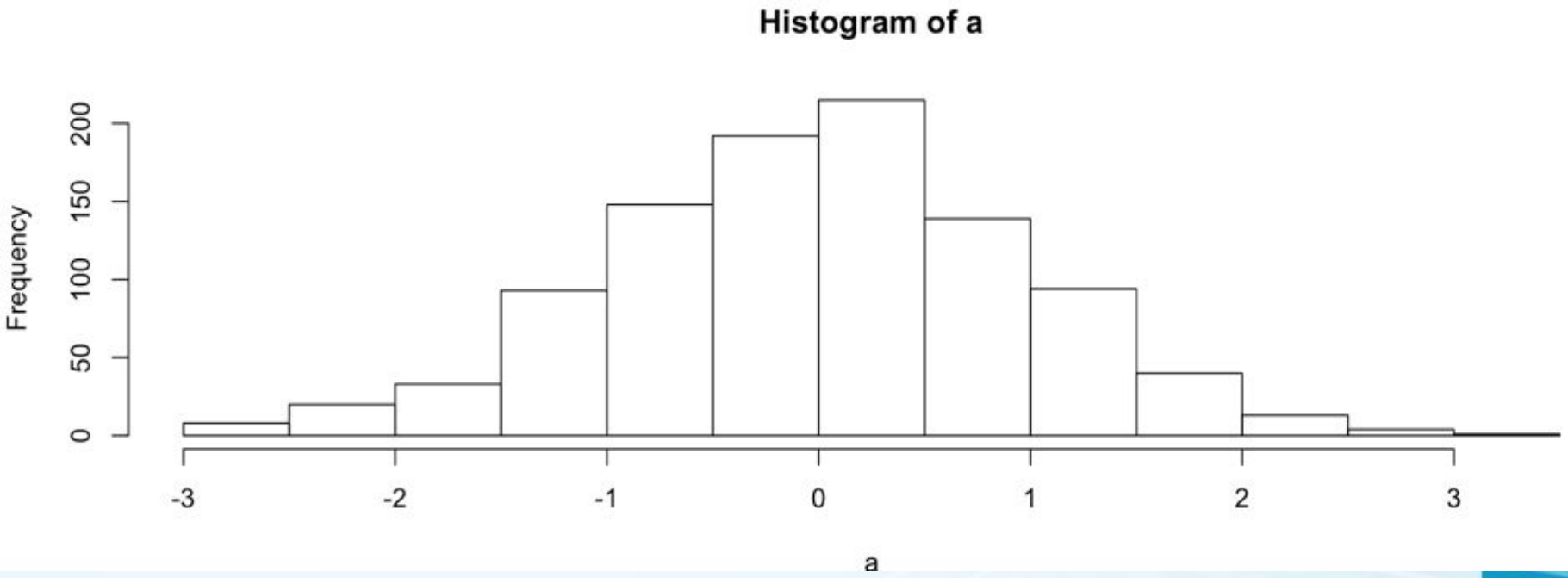
xlim - range of values on the x-axis.

ylim - range of values on the y-axis.

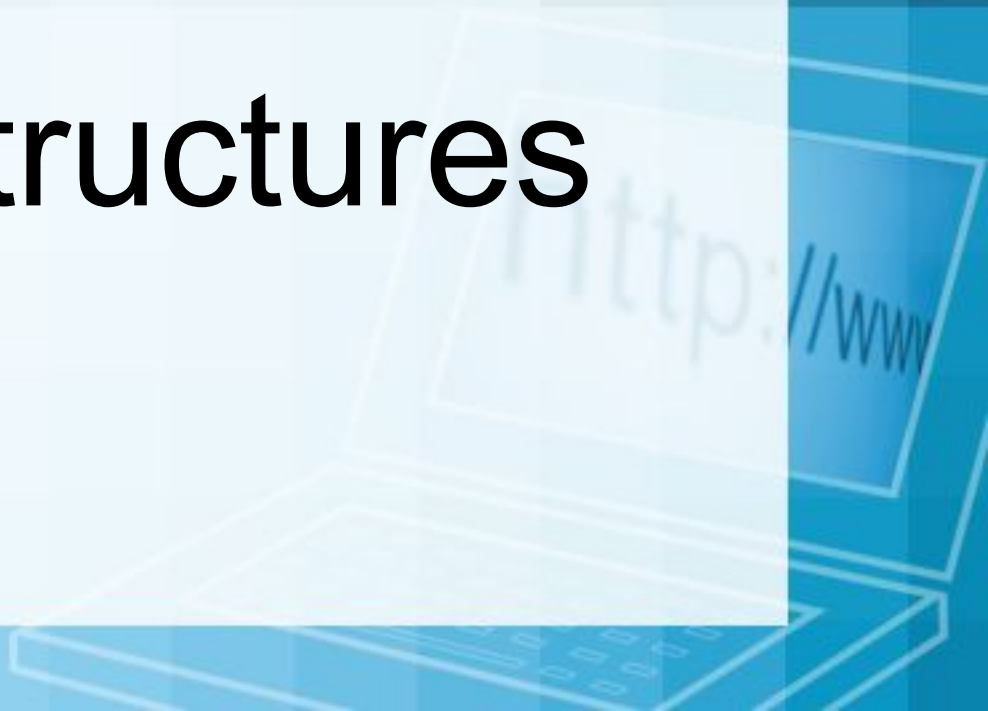
breaks - the width of each bar.

Histogram

```
a <- rnorm(1000)
hist(a)
```

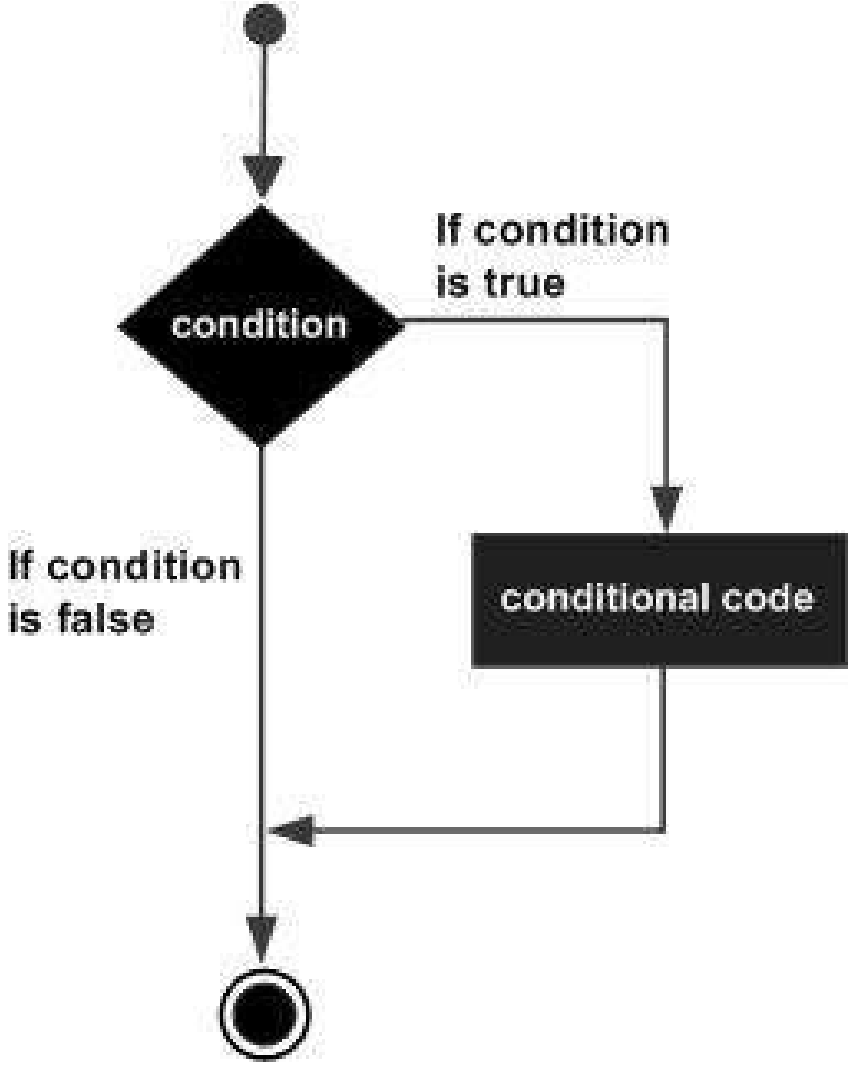


Module 6
Control Structures



Conditional

if Statement



if Syntax

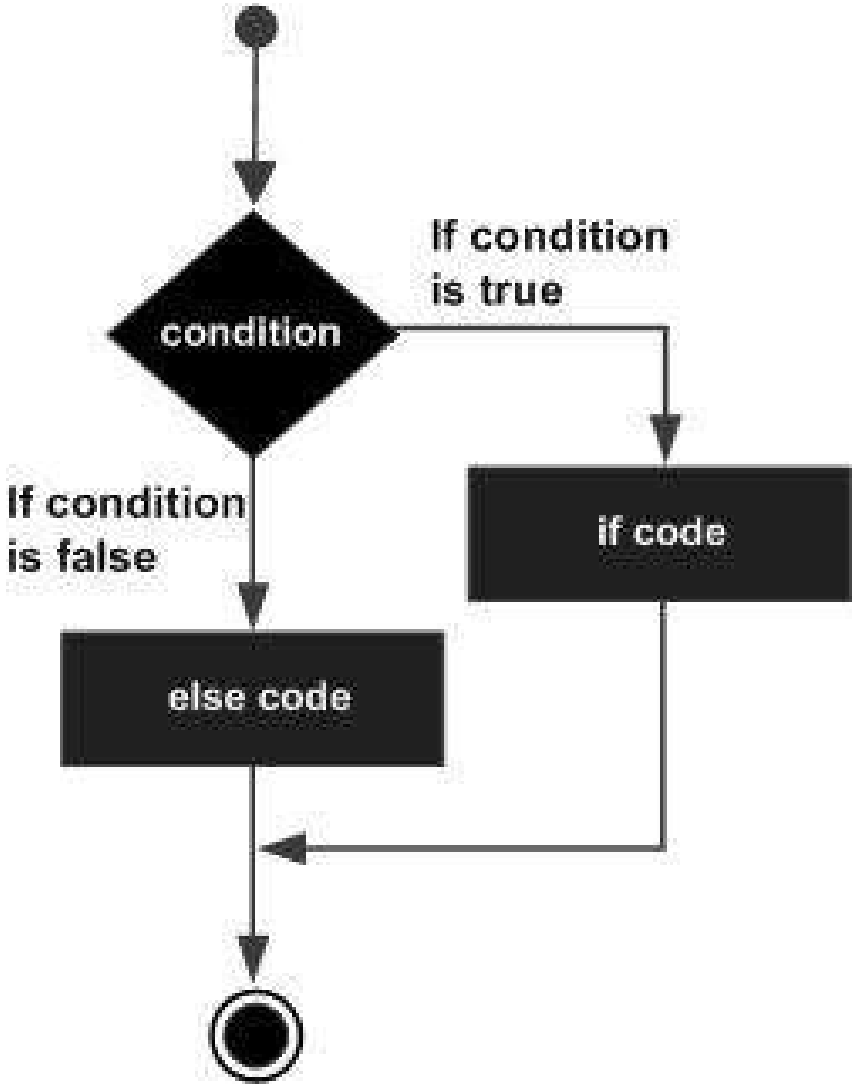
```
if (condition) {
    do Something
    }
```

if Statement Example

```
x <- 3
y <- 4

if (x<y) {
    print("x is smaller than y")
}
```

Ex file: ifelse.R

if-else Statement



if-else Syntax

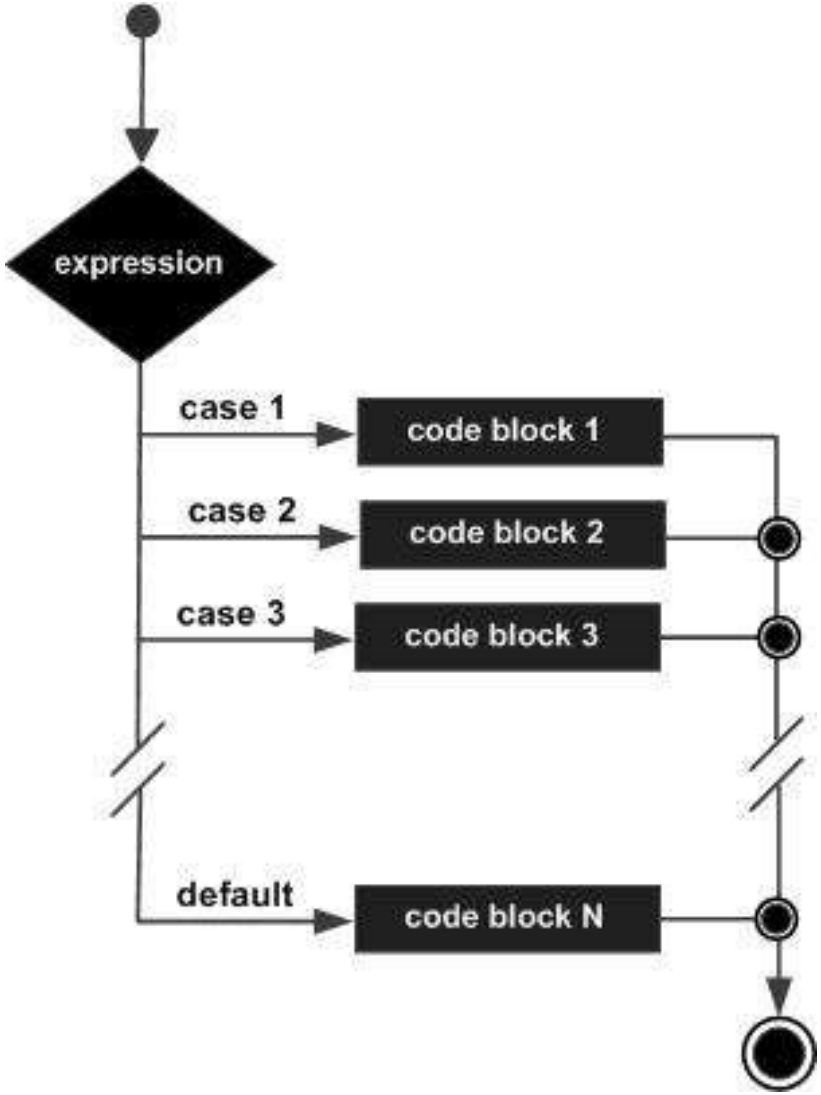
```
if (condition) {
    do Something}
else {
    do Something Else}
```

If-else Statements

```
x <- 5
y <- 4

if (x<y) {
    print("x is smaller than y")
} else {
    print("x is larger than y")
}
```

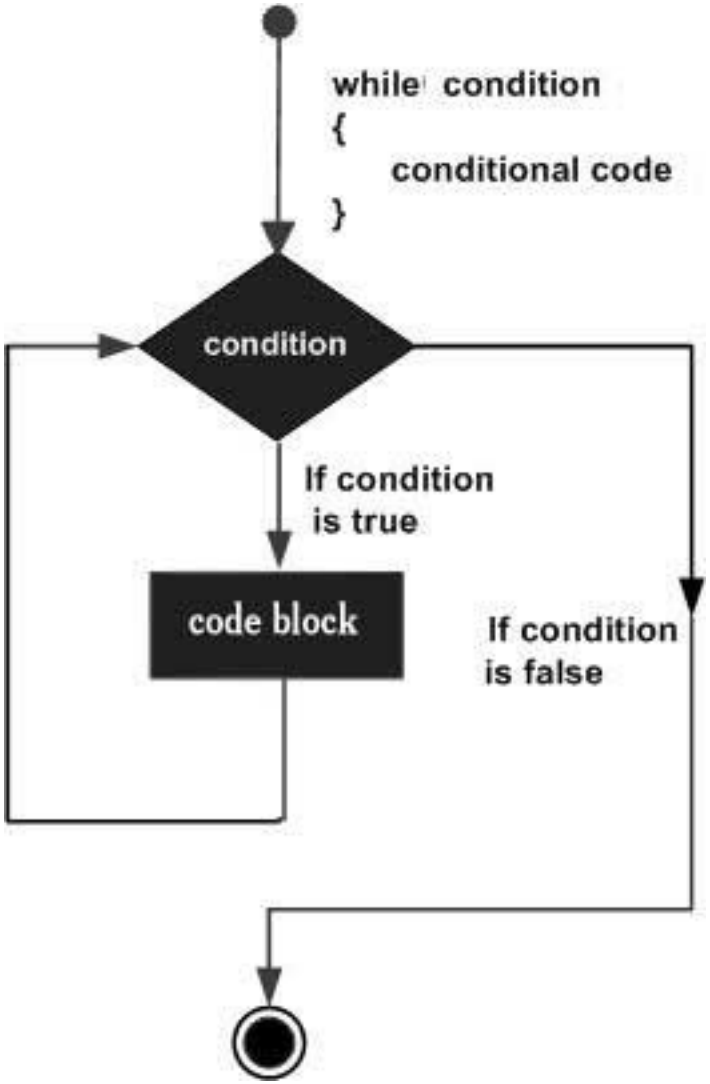
Ex file: ifelse.R

Switch Statement



Loop

While Loop Statement



While Loops Syntax

```
while (condition) {
    do Something
}
```

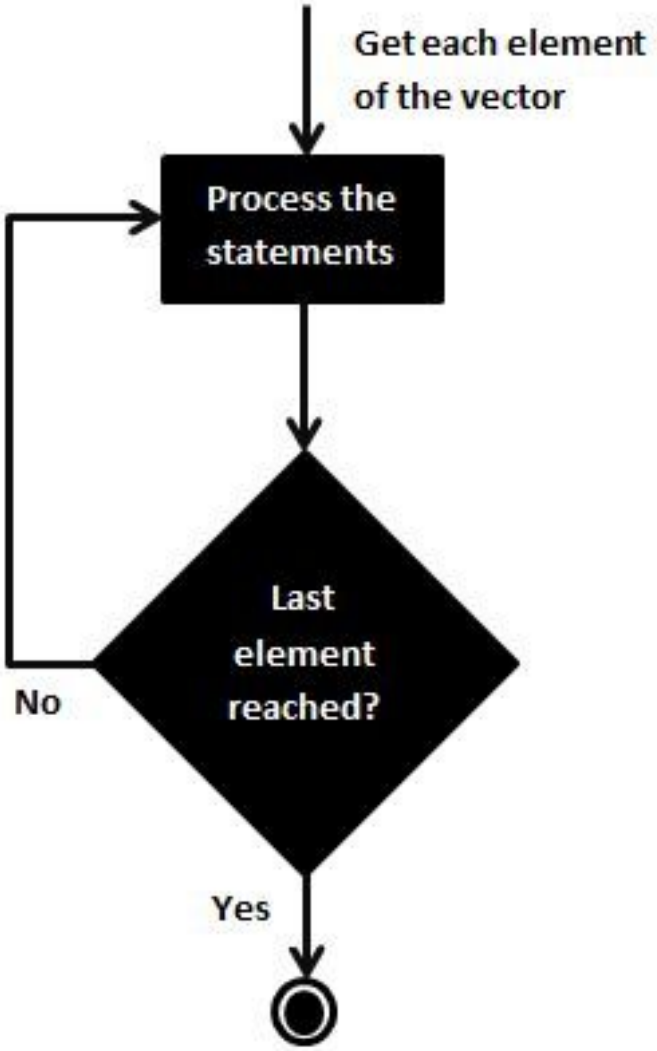
While Loops Examples

```
x <- 0
while (x<10) {
    print(x)
    x <- x+1
  }
```

While loop can potentially be infinite loop, be careful!

Ex file: whileloop.R

For Loop Statement



For Loops Syntax

```
for (variable in vector) {
    statements
}
```

For Loops Example

```
v <- c(1,2,3,4,5)
for ( i in v) {
   print(i)
}
```

Ex file: forloop.R

Next

```
for ( i in 1:10) {
    if (i == 7) {next}
    print(i)
}
```

Ex file: next.R

Break

```
for ( i in 1:10) {
    if (i == 7) {break}
    print(i)
}
```

Ex file: Break.R

Challenge

1. Read data1.csv
2. Do a subset of month of May and June
3. Count the number of days where temp is more than 65

Time: 10 mins

Hint to Challenge

```
data <- read.csv('data1.csv',header=TRUE)
data.subset <- subset(data, Month == 5 | Month == 6)
temp <- data.subset$Temp
count <- 0
for (i in temp) {
  if (i >65) { count <- count +1}
}
print(a)
```

Operators

Arithmetic Operators

Addition	+
Substration	-
Multiplication	*
Division	/
Modulus	%%

Logical Operators

and	&&
or	||
not	!

elementwise	
and	&
or	|

Module 7
Function

Function syntax

```
variable_name  <- function(arg_1, arg_2, ...) {
    Function body
}
```

The last expression is the return value

Built In Functions in R

factorial(3)

mean(1:6)

Function Examples

```
f <- function(x) {
  x*x
}

filter <- function(x) {
  x[x>0]
}
```

Ex file: function.R

Function with Default Args

```
f <- function(x=3,y=4) {
  x*2+y*3
}
```

Ex file: function.R

Named Args

```
f <- function(x,y) {
  x*x+y*y*y
}
f(x=3,y=2)
f(x=2,y=3)
```

Argument Matching

The order of argument matching is

1. Check for exact match for a named argument
2. Check for a partial match
3. Check for a positional match

... Argument

The ... argument indicate a variable number of arguments that are usually passed on to other functions.

```
f <- function(x,y,...) {
  plot(x,y,...)
}
```

Eg

```
f(x,y,col="red",main="sine")
```

Nested Function

```
make.power <- function(n) {
  pow <- function(x) {
    x^n
  }
  pow
}

cube = make.power(3)
cube(4)
```

Challenge: Function

Write a function to roll 2 dices, return the sum of the 2 dices

1 dice : 1, 2,3,4,5,6

Hint to Challenge

```
roll <- function() {
    …
    dice <-sample(1:6,size=2)
    ….
}
```

Module 8
Statistical Application of R

Basic Statistics

Mean Syntax

mean(x, trim = 0, na.rm = FALSE, ...)

trim is used to drop some observations from both end of the sorted vector

na.rm is used to remove the missing values from the input vector.

Eg:

mean(x)

mean(x,trim=0.3)

Ex:selectingdata.R

Median Syntax

median(x, na.rm = FALSE)

x is the input vector.

na.rm is used to remove the missing values from the input vector.

Ex:selectingdata.R

Summary

summary(x)

Min, 1st Q, Median, Mean, 3rd Q, Max

Ex:selectingdata.R

Correlation

Correlation

It provides a measure of strength and direction of linear relationship between two variables

Correlation Example - MTCars

```
corMat <- cor(mtcars[,c(1,3,6)])
round(corMat,2)
```

```
      mpg   disp    wt
mpg  1.00 -0.85 -0.87
disp -0.85  1.00  0.89
wt   -0.87  0.89  1.00
```

Ex:correlation.R

Challenge: Correlation

Perform a correlation for the quakes data -

Time: 2 mins

Linear Regression

Linear Regression

- Finding a straight line that best describes the data
- The best fit line is then used for making prediction

Linear Regression

```
# Fitting linear model
m<-lm(y ~ x,data)

# Intercept and Slope
coef(m)
summary(m)

# Prediction
p <- predict(m, data.frame(x = ..))
```

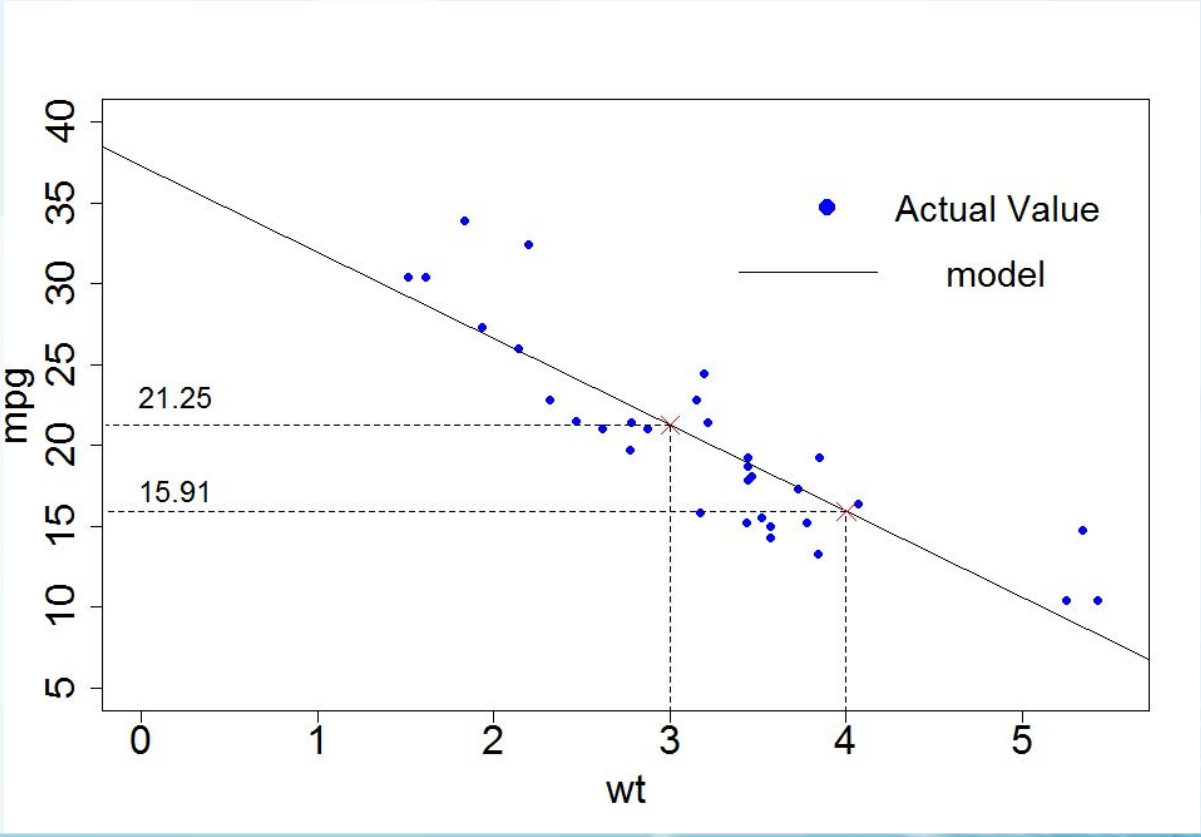
LR Example - MTCars

```
m <- lm(mpg ~ wt, data = mtcars)


coef(m)
(Intercept)          wt
 37.285126       -5.344472
summary(m)

p <- predict(m, data.frame(wt = 3))
```

Linear Regression - MTCars

plot(mpg~wt,data=mtcars)

abline(m)



Challenge: Linear Regression

1. Do a linear regression for the quake dataset - mag vs stations
2. Predict the quake mag when there are 100 stations receiving the quake signal.

Time : 5 mins

Multiple Regression

```
m<-lm(y ~ x1+x2+x1*x2...,data)

coef(m)
summary(m)


p<- predict(m, data.frame(x1=..,x2=..,...))
```

Multiple Regression - MTCars

```
m <- lm(mpg ~ wt+hp, data = mtcars)

coef(m)
(Intercept)          wt          hp
37.22              -3.87       -0.03
summary(m)

p<- predict(m, data.frame(wt = 3, hp = 200))
```

MR with Interaction - MTCars

```
m <- lm(mpg ~ wt*hp, data = mtcars)

coef(m)
(Intercept)          hp          wt       hp:wt
49.80              -0.12      -8.21       0.02
summary(m)

p<- predict(m, data.frame(wt = 3, hp = 200))
```

Logistics Regression

Logistics regression is a nonlinear regression that apply to response (y) that is binary

```
m<-glm(y ~ x1+x2+x3...,data)

coef(m)
summary(m)

p<- predict(m, data.frame(x1=..,x2=..,...))
```

Logistics Regression - MTCars

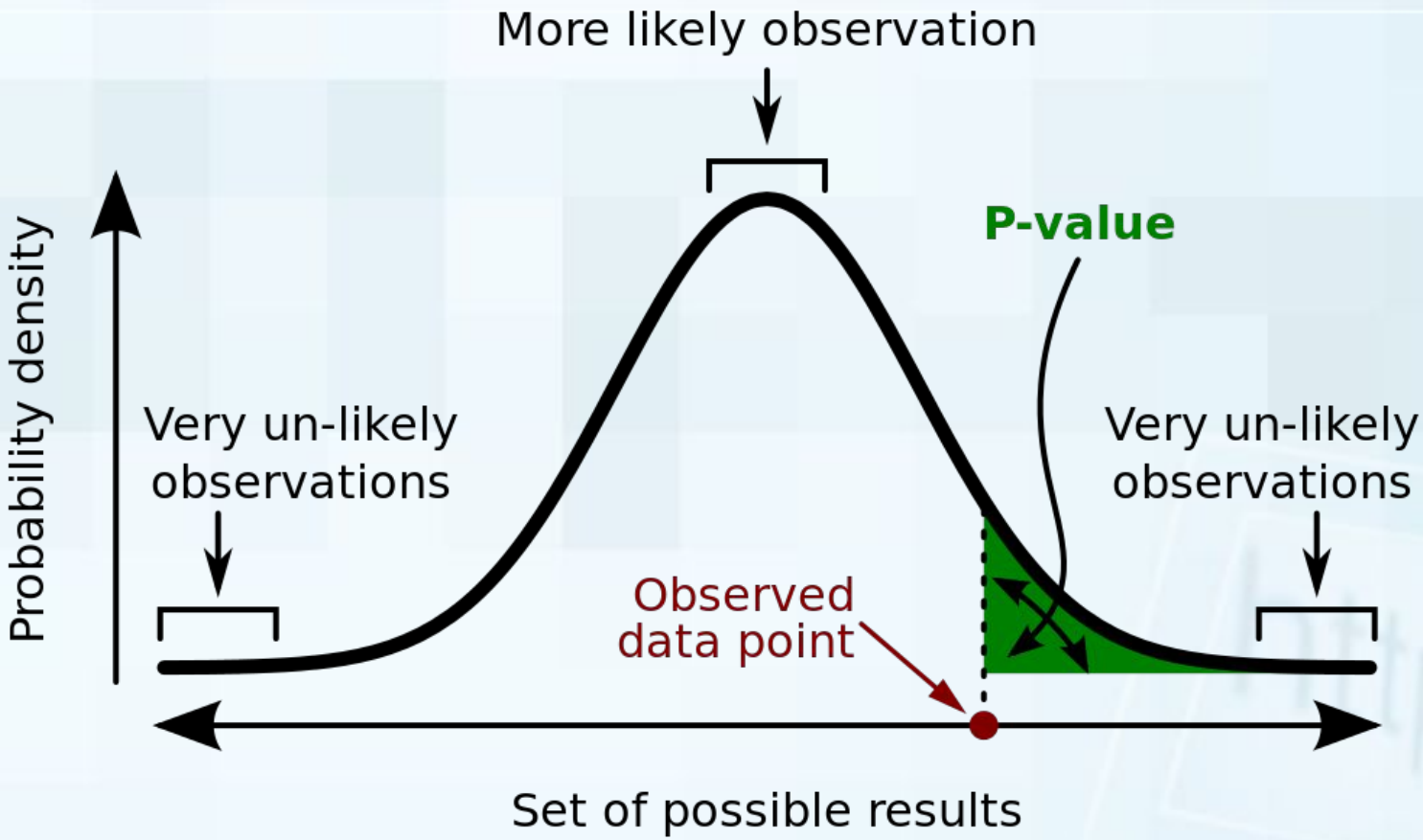
```
m <- glm(am ~ mpg+wt+hp, data = mtcars)

coef(m)
(Intercept)         mpg          wt          hp
 0.195756590  0.036308683 -0.338756898
0.003891913
summary(m)
```

Hypothesis Testing

Hypothesis Testing

- Assume Null Hypothesis is true - No difference or no effect
- Compute the p-value
- If the p-value is small, then reject Null Hypothesis, else do not reject Null Hypothesis

P-Value



A **p-value** (shaded green area) is the probability of an observed (or more extreme) result assuming that the null hypothesis is true.

Guideline for p-value

If p value > .10 → “not significant”

If p value ≤ .10 → “marginally significant”

If p value ≤ .05 → “significant”

If p value ≤ .01 → “highly significant.”

2 Sample t-Test (2 Sided)

```
data(sleep)
extra<-sleep$extra
group<-sleep$group

t.test(extra~group,data=sleep)
```

2 Sample t-Test (One Sided)

```
data(sleep)
extra<-sleep$extra
group<-sleep$group

t.test(extra~group,data=sleep,alternative="less")
```

1 Sample t-Test

t.test(extra ~ group, sleep, paired=TRUE)

Challenge: t-Test

Do a 2 sample t-test to compare the performance of chickwts feed - casein vs horsebean

Time : 5 mins

Hint to Challenge

- subset(chickwts,feed == "casein" | feed == "horsebean")
- Perform the t-Test on the subset

ANOVA

- t-test can do test 2 groups.
- When there are more than 2 groups then use ANOVA to test if there are statistically significant difference between the means.
- ANOVA does this by analysing the variance in the data set.
- ANOVA is similar to multiple regression

ANOVA

m<-aov(y~x1+x2+x1*x2….,data)

summary(m)

ANOVA Example - Chickwts

```
m<-aov(weight~feed,data=chickwts)
summary(m)
```

Alternatively can use Linear Regression Method

```
m <- lm(weight~feed,data=chickwts)
summary(m)
```

Challenge: ANOVA

Perform an ANOVA to determine any difference between the test scores of 3 teaching methods

Method A	Method B	Method C
79	71	82
86	77	68
94	81	70
89	83	76

Hint to Challenge

```
A <- c(79,86,94,89)
B <- c(71,77,81,83)
C <- c(82,68,70,76)
scores = c(A,B,C)
groups <- factor(rep(letters[1:3],each=4))
D <- data.frame(scores,groups)
boxplot(scores~groups,data=D)
D.aov <- aov(scores~groups, data=D)
summary(D.aov)
```

Cluster Analysis

Cluster Analysis

- Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups (clusters)
- Cluster analysis is used in data mining, machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.

Clustering Algorithm

- Hierarchical clustering (connectivity model)
- K Means clustering (centroid model)

Hierarchical Clustering

```
M <- dist(mylris)
hc <- hclust(M)
clusters <- cutree(hc, k = 3)
```

K-means Clustering

kmeans(myIris, centers = 3, nstart = 10)

Resources

- R Packages : https://cran.r-project.org/web/views/
- R Packages http://cran.stat.ucla.edu/web/packages/available_packages_by_name.html
- R Packages http://crantastic.org/

Summary
Parting Message

Q&A
Feedback

Thank You!

Dr. Alfred Ang
96983731
angch@tertiaryinfotech.com