

# Fun with



Bharat Bhushan Verma

<https://www.linkedin.com/in/dearbharat/>

dearbharat@gmail.com

+1 902 418 6649

# Agenda

- Module 1. Getting Started
  - What is R
  - Install R and RStudio / Anaconda
  - Explore Environment
  - 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. Control Structures (Self Study)
  - Conditional
  - Loop
  - Break & Next
  - Operators
- Module 5: Function (Self Study)
  - Function Syntax
  - Function Example
  - Function With Default Arguments



# **GETTING STARTED**

Module 1

# What is R?



Was created by  
Ross Ihaka Robert Gentleman

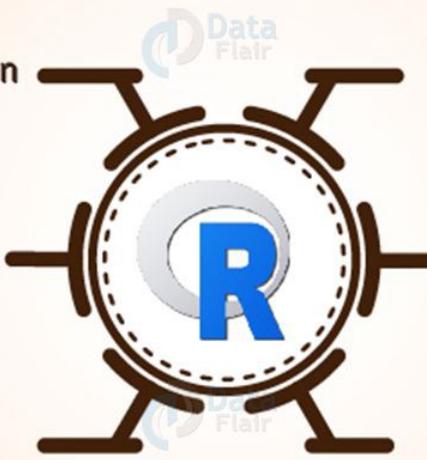
Named after first names  
of first 2 authors

Stable beta version in 2000

Was created at  
Auckland New Zealand

Project considered in 1992

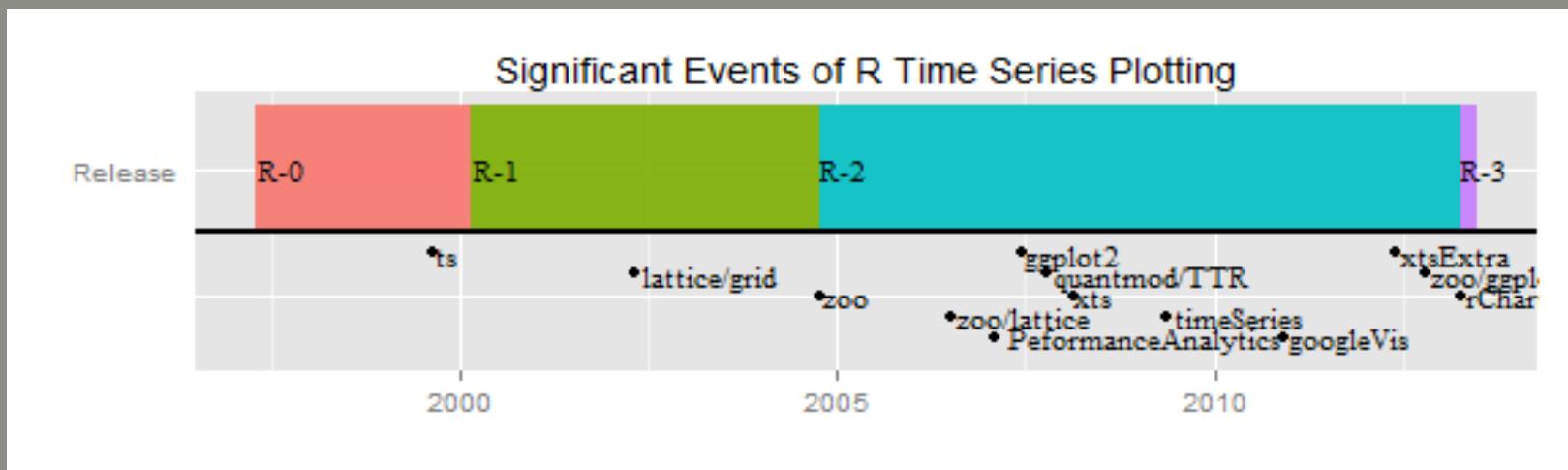
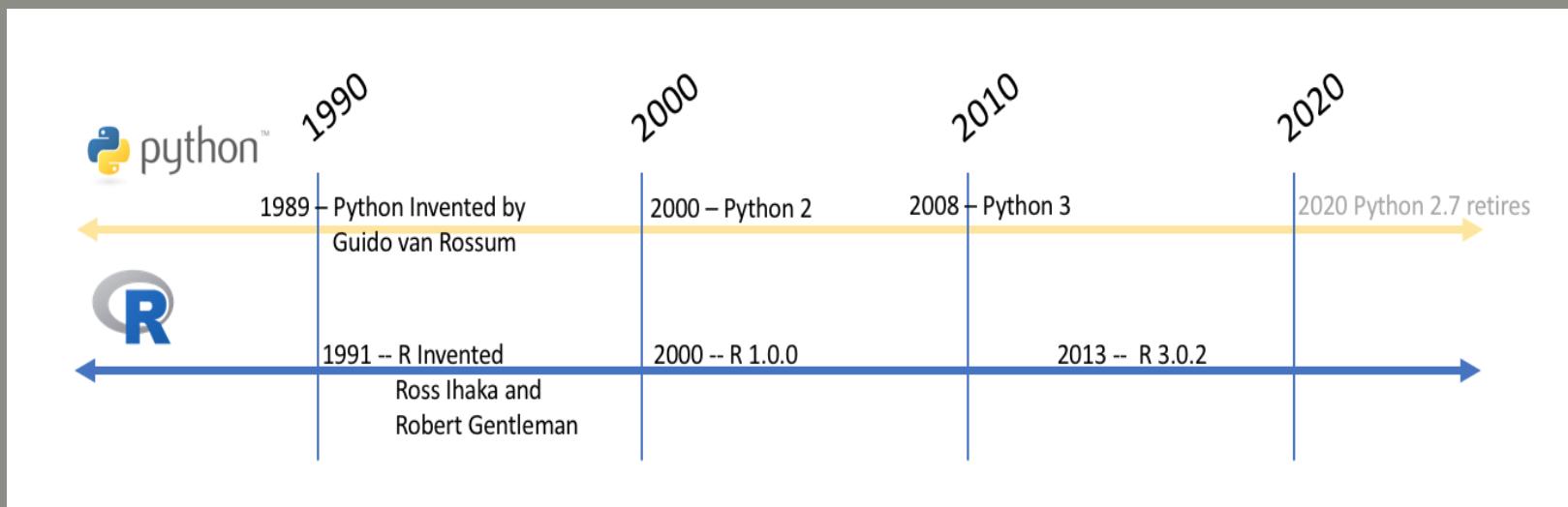
Initial version released in 1995



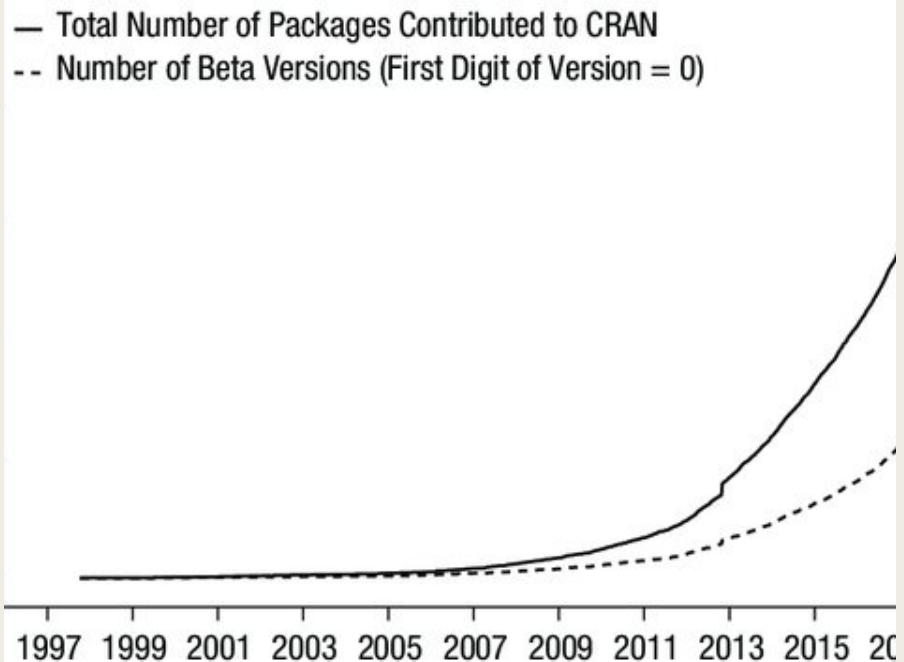
# Why R?

- R is open source and free!
- R is vector-based
- R is a programming environment suited for statistics
- R can be integrated with other software

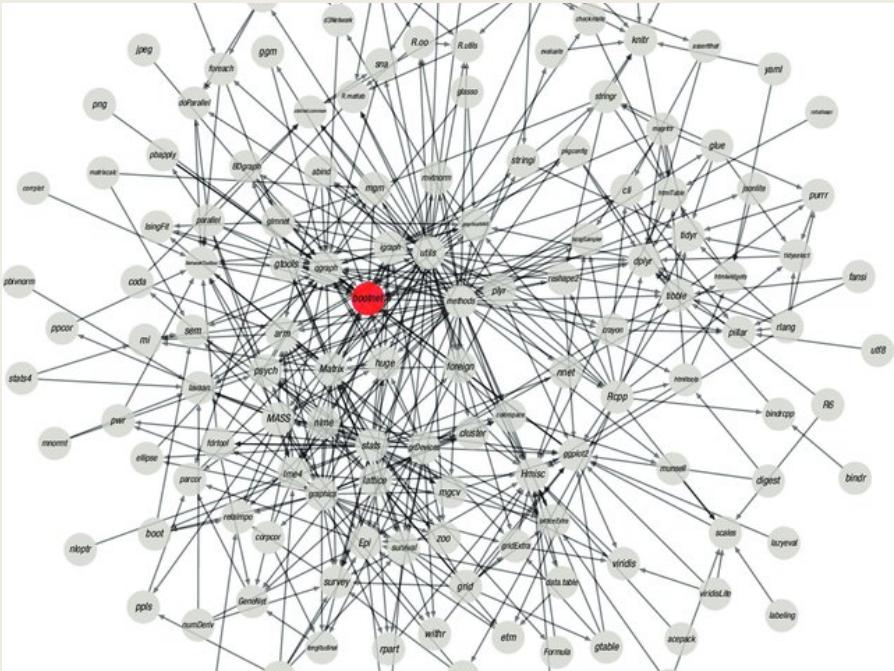
# R Development History



# NUMBER OF PACKAGES IN R



# PACKAGE DEPENDENCY IN R



# Where to get R from?

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



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.

# R Studio IDE

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

The screenshot shows the RStudio website homepage. At the top, there's a navigation bar with the RStudio logo, followed by links for Products, Resources, Pricing, About Us, and Blog, and a search icon. Below the navigation, there's a testimonial from Marina Theodosiou, Risk Analytics Manager at Funding Circle, featuring her photo and a quote about Shiny. A button below the quote says "Click to download the full story". The main content area has three sections: "Powerful IDE for R" with a lightning bolt icon, "R Packages" with a gift icon, and "Bring R to the web" with a cloud icon.

**R Studio**

Products   Resources   Pricing   About Us   Blog  

 "The establishment of interactive feedback loops through Shiny allows us to continually optimize our decision processes."

- Marina Theodosiou, Risk Analytics Manager at [Funding Circle](#)

Click to download the full story

 Powerful IDE for R

RStudio IDE is a powerful and productive user interface for R. It's free and open source, and works great on Windows,

 R Packages

Our developers and expert trainers are the authors of several popular R packages, including ggplot2, plyr,

 Bring R to the web

Shiny is an elegant and powerful web framework for building interactive reports and visualizations using R —

# Explore RStudio Interface

**SCRIPT**

```
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, 100, 300, 500, 3000, 3500, 7000),
9           #           col = "steelblue" Or else: col = colors() [626]
10          #           main = "Histogram of Annual Canadian Lynx Trappings\n1821-1934",
11          #           xlab = "Number of Lynx Trapped")
12
13
14 rm(list = ls()) # Clean up
```

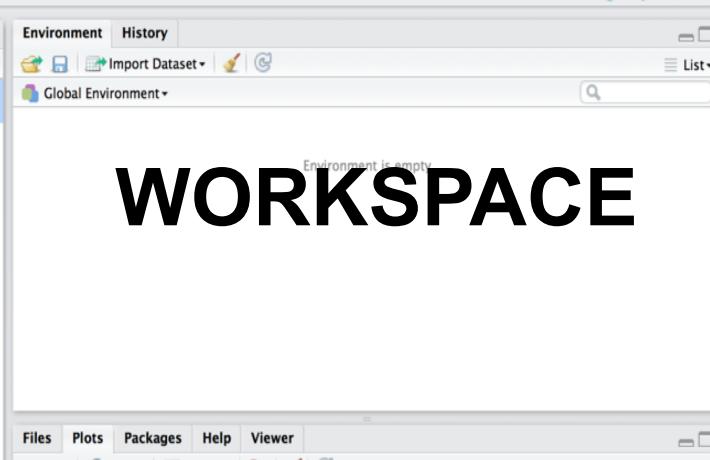
**CONSOLE**

```
> 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
> #
```

**WORKSPACE**

Environment is empty

**PACKAGE/ PLOTS/HELP**



A histogram titled "Histogram of Annual Canadian Lynx Trappings 1821-1934". The x-axis is labeled "Number of Lynx Trapped" and ranges from 0 to 7000. The y-axis is labeled "Frequency" and ranges from 0 to 40. The histogram shows a distribution with a peak frequency of approximately 40 at the lowest value (around 200). The distribution is right-skewed, with most values falling between 0 and 3000.

Number of Lynx Trapped (Bin)	Frequency
0 - 100	~40
100 - 200	~18
200 - 300	~7
300 - 400	~6
400 - 500	~4
500 - 600	~2
600 - 700	~1
700 - 800	~1
800 - 900	~1
900 - 1000	~1
1000 - 1100	~1
1100 - 1200	~1
1200 - 1300	~1
1300 - 1400	~1
1400 - 1500	~1
1500 - 1600	~1
1600 - 1700	~1
1700 - 1800	~1
1800 - 1900	~1
1900 - 2000	~1
2000 - 2100	~1
2100 - 2200	~1
2200 - 2300	~1
2300 - 2400	~1
2400 - 2500	~1
2500 - 2600	~1
2600 - 2700	~1
2700 - 2800	~1
2800 - 2900	~1
2900 - 3000	~1
3000 - 3100	~1
3100 - 3200	~1
3200 - 3300	~1
3300 - 3400	~1
3400 - 3500	~1
3500 - 3600	~1
3600 - 3700	~1
3700 - 3800	~1
3800 - 3900	~1
3900 - 4000	~1
4000 - 4100	~1
4100 - 4200	~1
4200 - 4300	~1
4300 - 4400	~1
4400 - 4500	~1
4500 - 4600	~1
4600 - 4700	~1
4700 - 4800	~1
4800 - 4900	~1
4900 - 5000	~1
5000 - 5100	~1
5100 - 5200	~1
5200 - 5300	~1
5300 - 5400	~1
5400 - 5500	~1
5500 - 5600	~1
5600 - 5700	~1
5700 - 5800	~1
5800 - 5900	~1
5900 - 6000	~1
6000 - 6100	~1
6100 - 6200	~1
6200 - 6300	~1
6300 - 6400	~1
6400 - 6500	~1
6500 - 6600	~1
6600 - 6700	~1
6700 - 6800	~1
6800 - 6900	~1
6900 - 7000	~1

# Interactive R Environment in Anaconda

The screenshot shows the Anaconda Navigator interface. On the left is a sidebar with icons for Home, Environments, Learning, and Community, along with links to Documentation and Developer Blog. At the bottom are social media links for Twitter, YouTube, and GitHub, and buttons for Create, Clone, Import, and Remove environments.

The main area displays the 'base (root)' environment, which contains an 'R' entry. To the right is a list of installed packages:

Name	Description	Version
_r-mutex		1.0.0
appnope	Disable app nap on os x 10.9	0.1.0
attrs	Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods).	19.3.0
certifi		3.1.0
cctools	Assembler, archiver, ranlib, libtool, otool et al for darwin mach-o files	895
clang	A c language family frontend for llvm	4.0.1
clang_osx-64		4.0.1
clangxx	A c language family frontend for llvm	4.0.1
clangxx_osx-64		4.0.1
compiler-rt	Compiler-rt runtime libraries	4.0.1
curl	Tool and library for transferring data with url syntax	7.67.0

A modal window titled 'Create new environment' is open in the center. It has fields for 'Name' (set to 'New environment name'), 'Location' (set to 'Python 2.7'), and 'Packages' (with 'R' checked). Buttons for 'Cancel' and 'Create' are at the bottom.

At the bottom of the main pane, it says '237 packages available'.

# Interactive R Environment in Google Colab

- <http://bit.ly/SMURProgGoogle>
- Using R in python environment on Colab
  - `!jupyter-kernelspec list`
  - `%load_ext rpy2.ipython`
  - `%%R`
  - `x <- 42`
  - `print(x)`
- <https://github.com/IRkernel/IRkernel/tree/master/examples-notebooks>

# Quiz

1. What are the three properties of a vector, other than its contents?
2. What are the four common types of atomic vectors? What are the two rare types?
3. What are attributes? How do you get them and set them?
4. How is a list different from an atomic vector? How is a matrix different from a data frame?
5. Can you have a list that is a matrix? Can a data frame have a column that is a matrix?



# DATA TYPES

Module 2

# Comments and Help

Single line comment

#

Help

?....

help(...)

# Variables

a <- 1

(Most common)

a = 1

1 -> a

# 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 & Joining Text

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

```
a<-"bharat.verma"  
b<-"smu.ca"  
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)

# 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
```

# 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:  $a>0$

# 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)
```



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","row2")
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
multiplication	
%*%	- dot product



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')  
)
```

# 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/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 FRAME

a[a\$gender=="Male",]

# 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

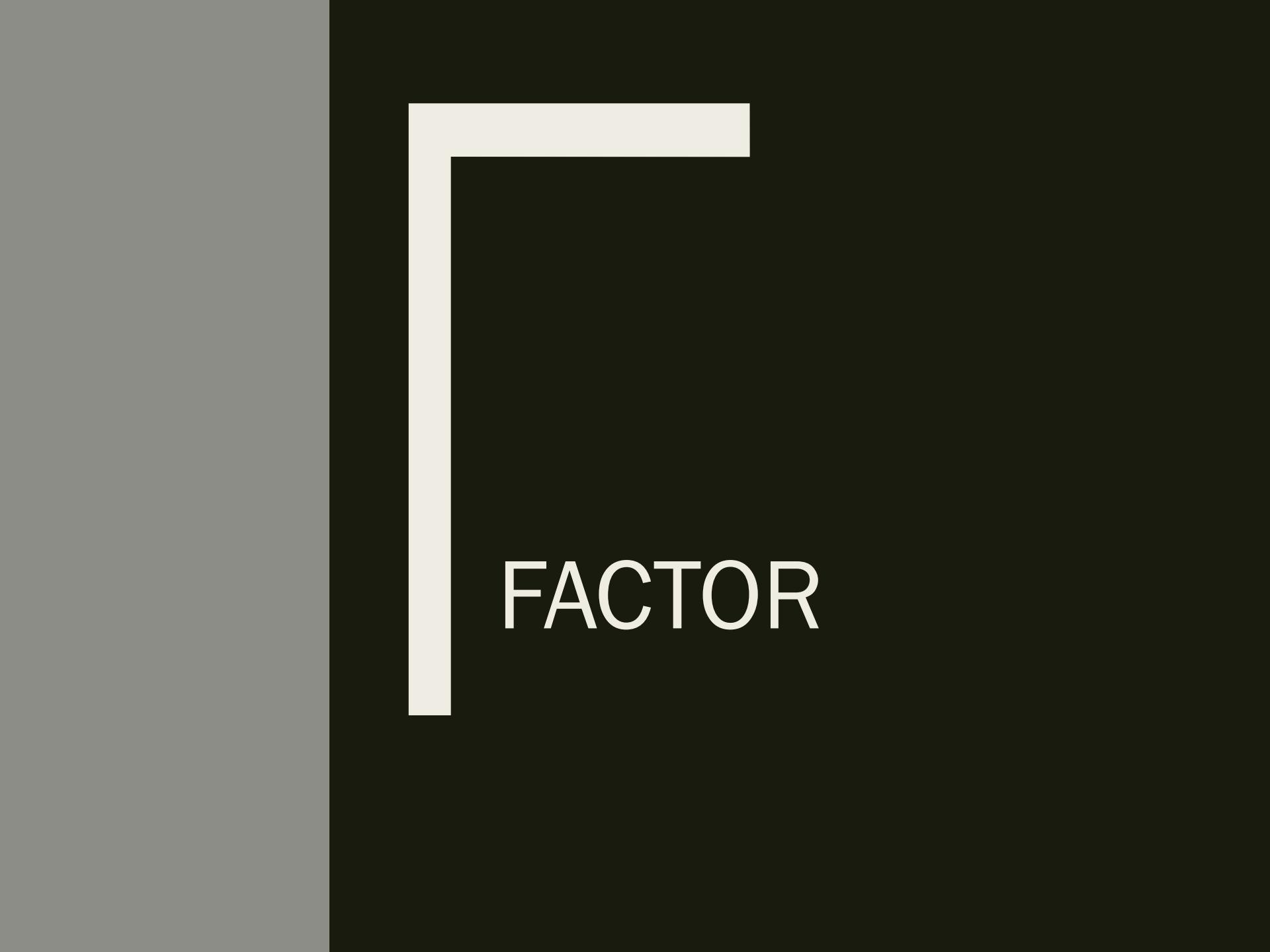
```
a$name =c('Ally','Belinda','Jane')
```

# 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 (categorical) data such as colors, countries, digits, letters

```
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

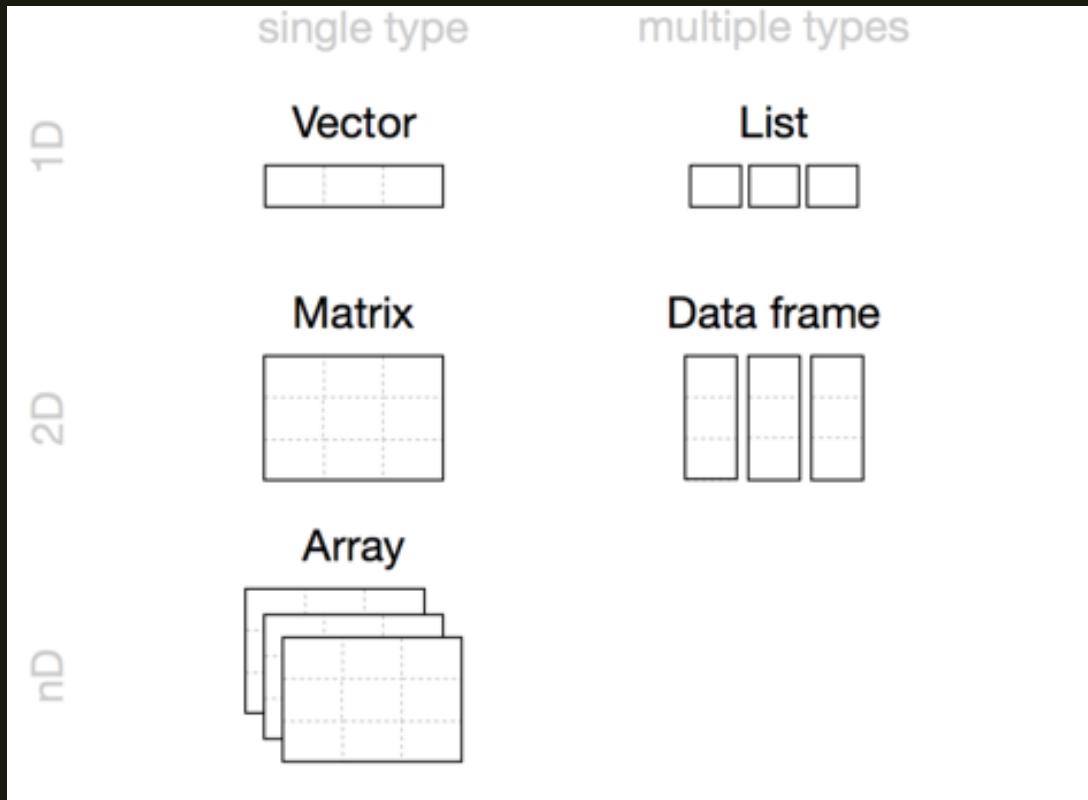
# Working with Date

`weekdays(xd)`

`months(xd)`

`xd+7`

# DATA TYPES SUMMARY





# PACKAGES & DATA SETS

Module 3

# PACKAGES

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

# R PACKAGES

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

# RStudio Package Management

Install and load package in RStudio  
Unload and remove a package in RStudio

The screenshot shows the RStudio interface with the following details:

- Left Panel (Console):** Displays the packages available in the system library:

```
Packages in library '/Library/Frameworks/R.framework/Versions/3.2/Resources/library':  
DBI      R Database Interface  
KernSmooth Functions for Kernel  
MASS     Smoothing Supporting Wand &  
         Jones (1995)  
Matrix    Support Functions and  
         Datasets for Venables and  
         Ripley's MASS  
NMF      Sparse and Dense Matrix  
         Classes and Methods  
R.methodsS3 Algorithms and Framework  
         for Nonnegative Matrix  
         Factorization (NMF)  
R.oo      S3 Methods Simplified  
         R Object-Oriented  
         Programming with or without  
         References  
R2HTML   HTML exportation for R  
         objects  
RColorBrewer ColorBrewer Palettes
```
- Right Panel (Packages Tab):** Shows the System Library with the `datasets` package selected (indicated by a red circle).

Name	Description	Version
bitops	Bitwise Operations	1.0-6
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-18
caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1
chron	Chronological Objects which can Handle Dates and Times	2.3-47
class	Functions for Classification	7.3-14
cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
codetools	Code Analysis Tools for R	0.2-14
coin	Conditional Inference Procedures in a Permutation Test Framework	1.1-2
colorspace	Color Space Manipulation	1.2-6
combinat	combinatorics utilities	0.0-8
compiler	The R Compiler Package	3.2.4
<b>datasets</b>	The R Datasets Package	3.2.4
DBI	R Database Interface	0.3.1
dichromat	Color Schemes for Dichromats	2.0-0
digest	Create Compact Hash Digests of R Objects	0.6.9
doParallel	Foreach Parallel Adaptor for the 'parallel' Package	1.0.10
e1071	Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien	1.6-7
evaluate	Parsing and Evaluation Tools that Provide More Details than the Default	0.9
foreach	Provides Foreach Looping Construct for R	1.4.3
foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, ...	0.8-66
- Bottom Panel (Values):** Displays the values of variables H, labels, and M.

# Install/Load Packages by Command Line

1. Install package

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

2. Load a package

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

# Unload and Remove Package

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

```
remove.packages("ggplot2")
```

# DATA SETS



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  
Passenger Miles on Commercial US Airlines, 1937-1960[airmiles](#)  
[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](#)[beaver2](#)[beavers](#)[RIndex](#)

Body Temperature Series of Two Beavers

Body Temperature Series of Two Beavers

Body Temperature Series of Two Beavers

[Sales Data with Leading Indicator](#)

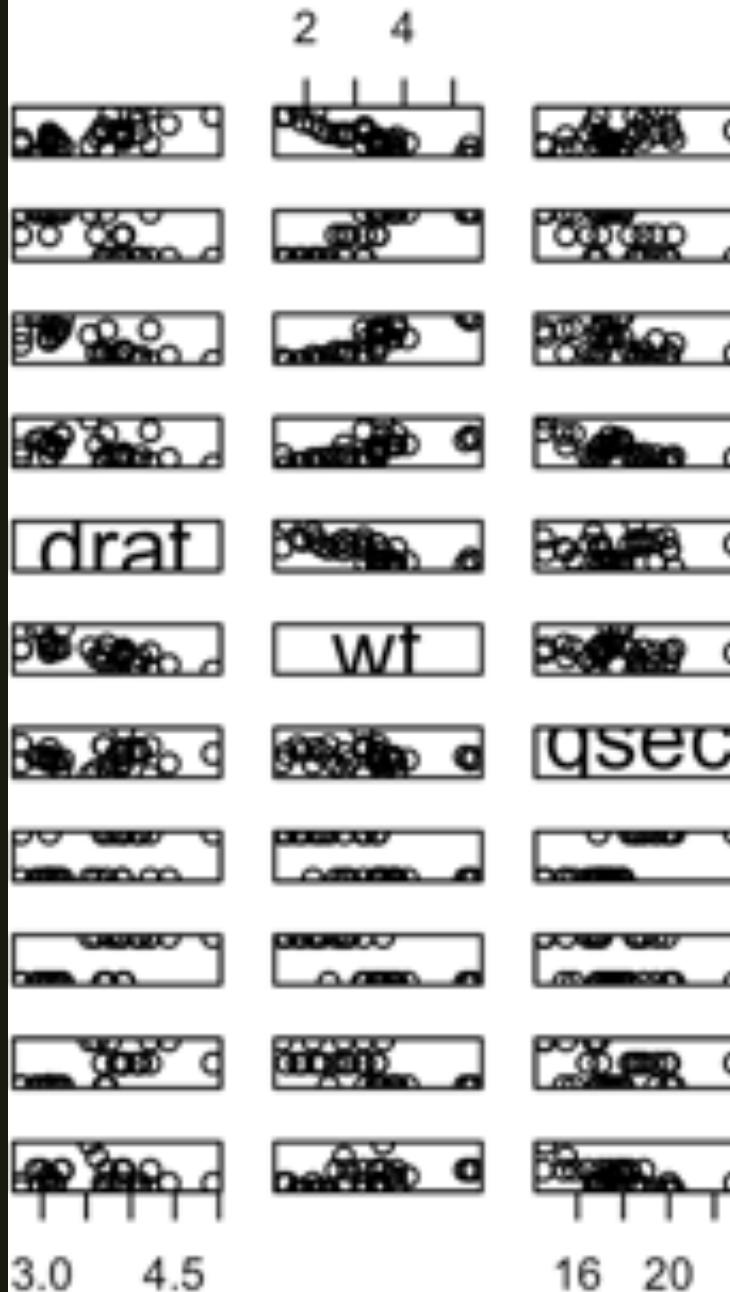
# R DATASET PACKAGES

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

# 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)
```



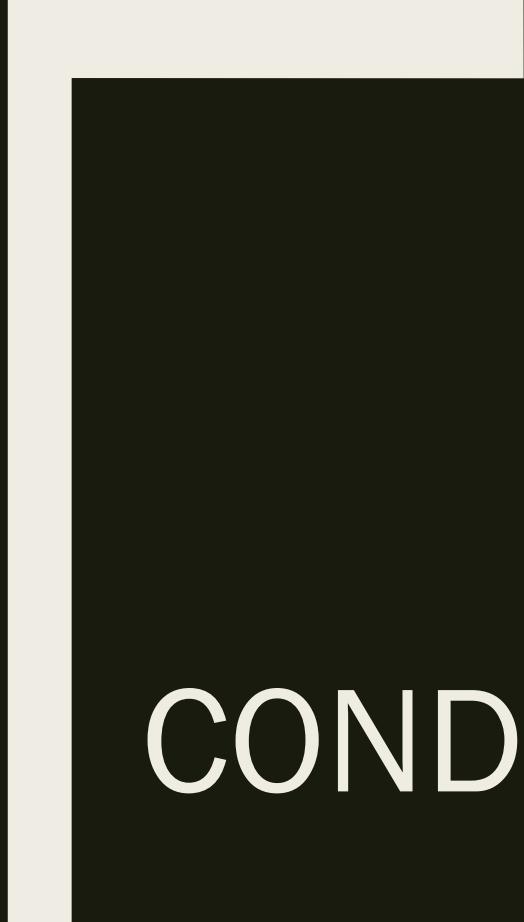
# MTCar Dataset Variables

- 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



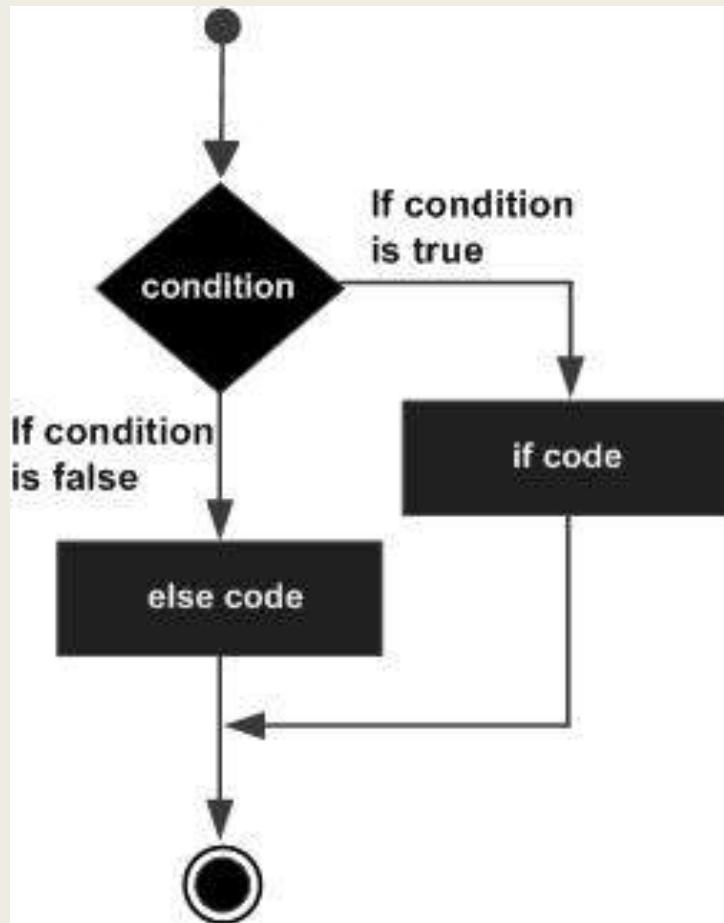
# CONTROL STRUCTURES

Module 4



CONDITIONAL

# IF-ELSE STATEMENT



# if-else Syntax

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

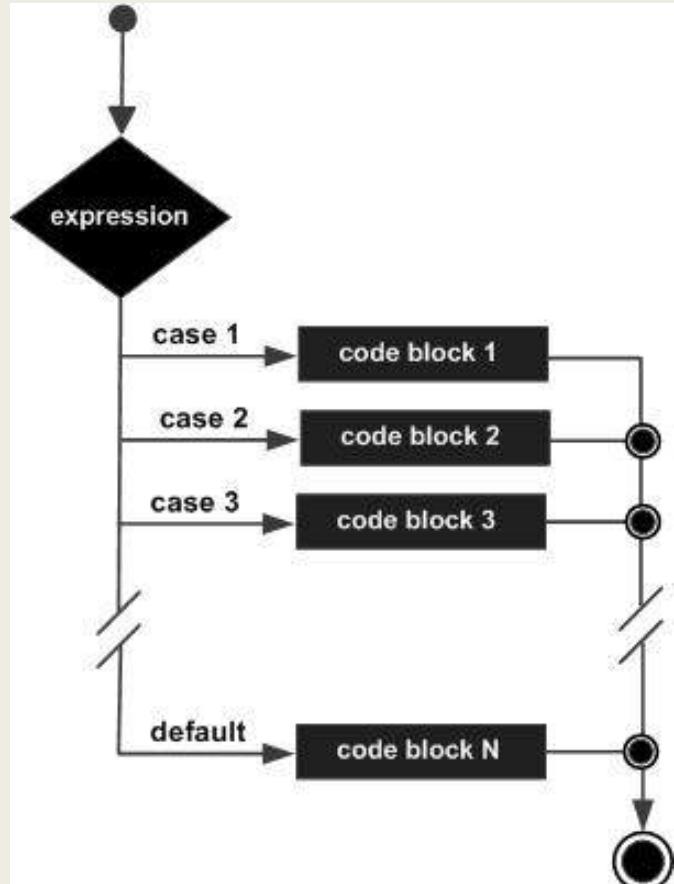
# If-else Statements

```
x <- 5
```

```
y <- 4
```

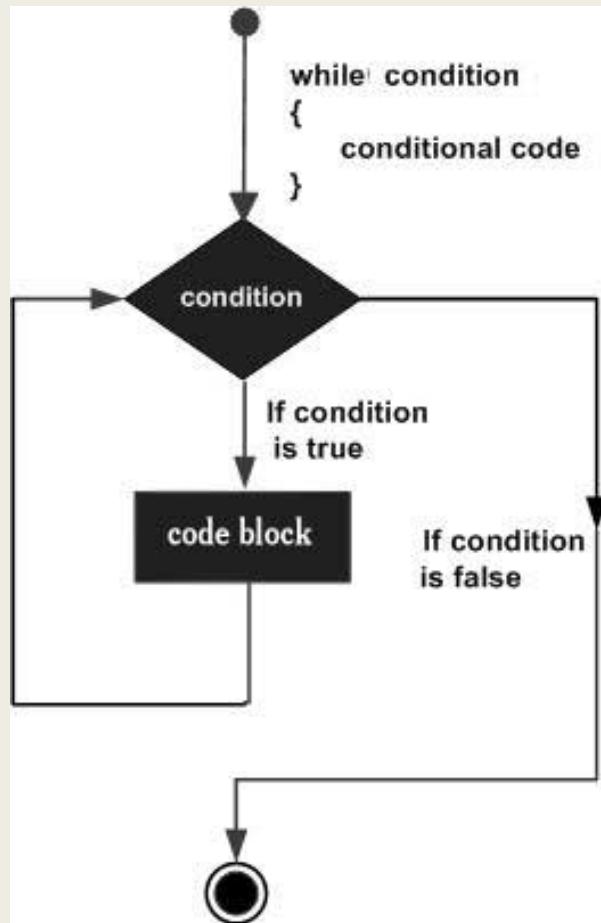
```
if (x<y) {  
  print("x is smaller than y")  
} else {  
  print("x is larger than y")  
}
```

# SWITCH STATEMENT



LOOP

# WHILE LOOP STATEMENT



# While Loops Syntax

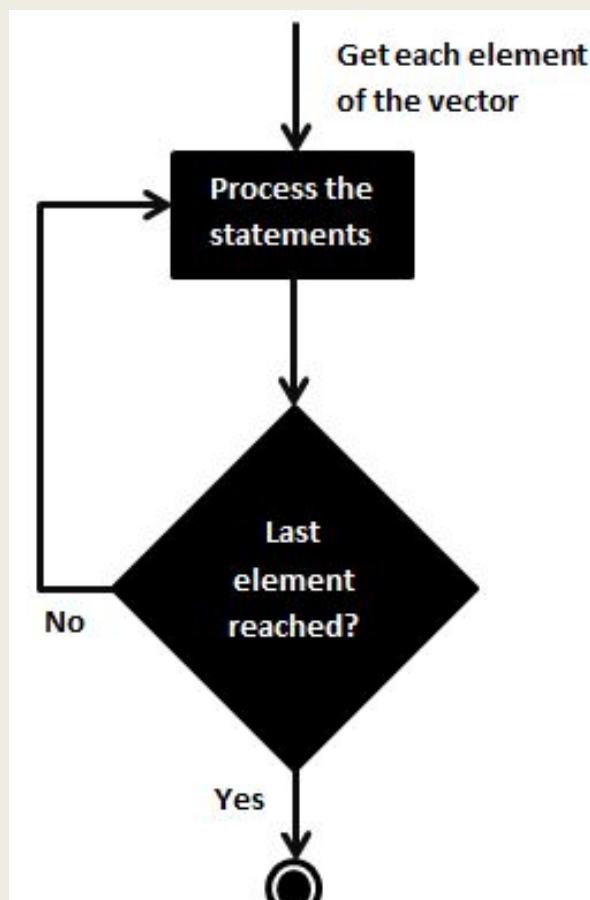
```
while (condition) {  
    do Something  
}
```

# While Loops Examples

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

While loop can potentially be infinite loop, be careful!

# 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)
}
```

# Next

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

# Break

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

# 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                         `|`



# FUNCTION

Module 5

# 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]  
}
```

# Function with Default Args

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

# 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)  
  ....  
}
```

# Quiz Solution

1. The three properties of a vector are type, length, and attributes.
2. The four common types of atomic vector are logical, integer, double (sometimes called numeric), and character. The two rarer types are complex and raw.
3. Attributes allow you to associate arbitrary additional metadata to any object. You can get and set individual attributes with `attr(x, "y")` and `attr(x, "y") <- value`; or get and set all attributes at once with `attributes()`.
4. The elements of a list can be any type (even a list); the elements of an atomic vector are all of the same type. Similarly, every element of a matrix must be the same type; in a data frame, the different columns can have different types.
5. You can make “list-array” by assuming dimensions to a list. You can make a matrix a column of a data frame with `df$x <- matrix()`, or using `I()` when creating a new data frame `data.frame(x = I(matrix()))`.

# Resources

- R Packages : <https://cran.r-project.org/web/views/>
- R Packages  
[http://cran.stat.ucla.edu/web/packages/available\\_packages\\_by\\_name.html](http://cran.stat.ucla.edu/web/packages/available_packages_by_name.html)
- R Packages <http://crantastic.org/>