Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Lecture 1
What is Data Science?

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Why data is needed?

Why do we collect the data?

To quench the thirst for knowledge.

We want to know the reason - How, Why, Where, Who…?

Easiest option- get some data about the relevant question.

Answer all the questions on the basis of collected data.

Data and Statistics

Data: Very important source of information but deaf and dumb.

Data has its own language like sign language.

Statistics is the language of data.

Collection, analysis and drawing inferences from numerical facts, referred as data analysis.

Statistics is a science of turning data into information to be used for decision making.

Data and Statistics

Proper interpretation of inferences which are drawn is important.

Statistics can not do miracles.

Statistics can not change the process or phenomenon.

Its a scientific way of extracting and retriving information.

Why collect data?

- **To verify theoretical findings,**
- **Draw inferences just on the basis of collected data,**
- **Developing statistical models, which can be further used for policy decisions, classification, forecasting etc.**

Data and Statistics

Statistics is a language of data.

	Correct data	Wrong data
Correct statistical tool	Correct decision	Incorrect decision
Wrong statistical tool	Incorrect decision	Incorrect decision

Rule: Garbage in – Garbage out

Statistics has its own derived rules.

Rules are framed such that correct decisions, as indicated by the data and based on the hidden information, are taken.

It does forecasting but not like astrologer‘s parrot.

Statistics and Data Science

How Statistics got transformed to Data Science?

What is expected from Data Science which was not expected from "Statistics".

Advent and rapid development in computers have impacted Statistics.

Earlier, it was difficult to collect the data and even many times the data was not available.

Now, data is easily available and too much data is available.

Big data analysis is the latest news, petabyte is the unit of data size.

Statistics, Computers and Data Science

Earlier, the emphasis was on theoretical developments in Statistics.

Computers helped in the development of from "Computational Statistics".

If theory and mathematical analysis became complicated, the computational statistics supplemented it.

With the computational support , the theoretical developments in statistics gained more relevance and applications.

The computations and statistics became the two inseperable parts of data science.

Statistics, Computers and Data Science

Once we adventure into the Computational Statistics, the role and use of computers became very important.

Computers require programming language, software, data management and several other aspects.

The areas of applications of statistics have increased.

Topics like artificial intelligence, machile learning, supervised learning, unsupervised learning, reinforcement learning are based on statistics but they are heavily based on compters.

Statistics, Computer Science and Data Science

Data science has various ingredients- Statistics, mathematics, computer science, ...

Objectives of statistics and data science are the same.

Statistics aims to extract the information contained in the data and so is the aim of data science.

Data science, when applied to different fields can lead to incredible new insights.

Statistics, Computer Science and Data Science

The only form of data that matters in decision science is digital data.

Digital data is information that is not so easily interpretable by an individual. It depends upon machines to interpret/ process and/or alter it.

What we see on a computer screen – text, photo, movie etc., they are the digital letters which is essentially a systematic collection of coded ones and zeros.

Expectation from Data Scientist

What is needed to become the data scientist?

First decide what we want to become- A Doctor or a Compounder?

Decide-

Want to only use the tools?

Want to understand the utility of tools?

Or want to develop the tools?

In my opinion- all are needed.

Role of Statistics in Data Science

Statistics is the soul of data science.

• Descriptive statistics	• Nonparametric inference
• Probability theory	• Multivariate analysis
• Statistical inference	• Linear regression analysis
• Decision theory	• Nonlinear regression analysis
• Bayesian inference	• Simulation techniques
• Frequentist inference	• Monte Carlo methods
• Parametric inference	• … … …

Role of Statistics in Data Science

The theoretical developments are essential which are needed to be exposed to computational procedures.

Computational procedures have their own limitations and so optimization methods are required.

The implementation of statistical, mathematical, optimization methods etc. are to be simultaneously implemented over a data set and for that, data management is required.

All these aspects are logically implemented in a systematic way and correct statistical inferences are drawn.

Role of Statistics in Data Science

Based on the obtained inferences, proper interpretations are made and used for policy formulation, policy prescription and further applications like forecasting etc.

So proper knowledge of all the fields is required to become a data scientist.

My role?

My job?

Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 2

Installation and Working with R

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

R Software

Use of a software is desirable and moreover an essential part of any analysis.

Some popular statistical software are SPSS, SAS, Minitab, Stata, Matlab etc.

Another software is R.

R is a free software.

Developers of R Software

Currently developed by the R Development Core Team.

Available at www.r-project.org

It supports many free packages which helps the data scientist and analyst.

What is R?

R is an environment for data manipulation, statistical computing, graphics display and data analysis.

Effective data handling and storage of outputs is possible.

Simple as well as complicated calculations are possible.

Simulations are possible.

What is R?

Graphical display on-screen and hardcopy are possible.

Programming language is effective which includes all possibilities just like any other good programming language.

R has a statistical computing environment.

R is free (open source) software and therefore is not a black box.

Switching to R

R is available for Windows, Unix, Linux and Macintosh platforms.

Built in and contributed packages are available, and users are provided tools to make packages.

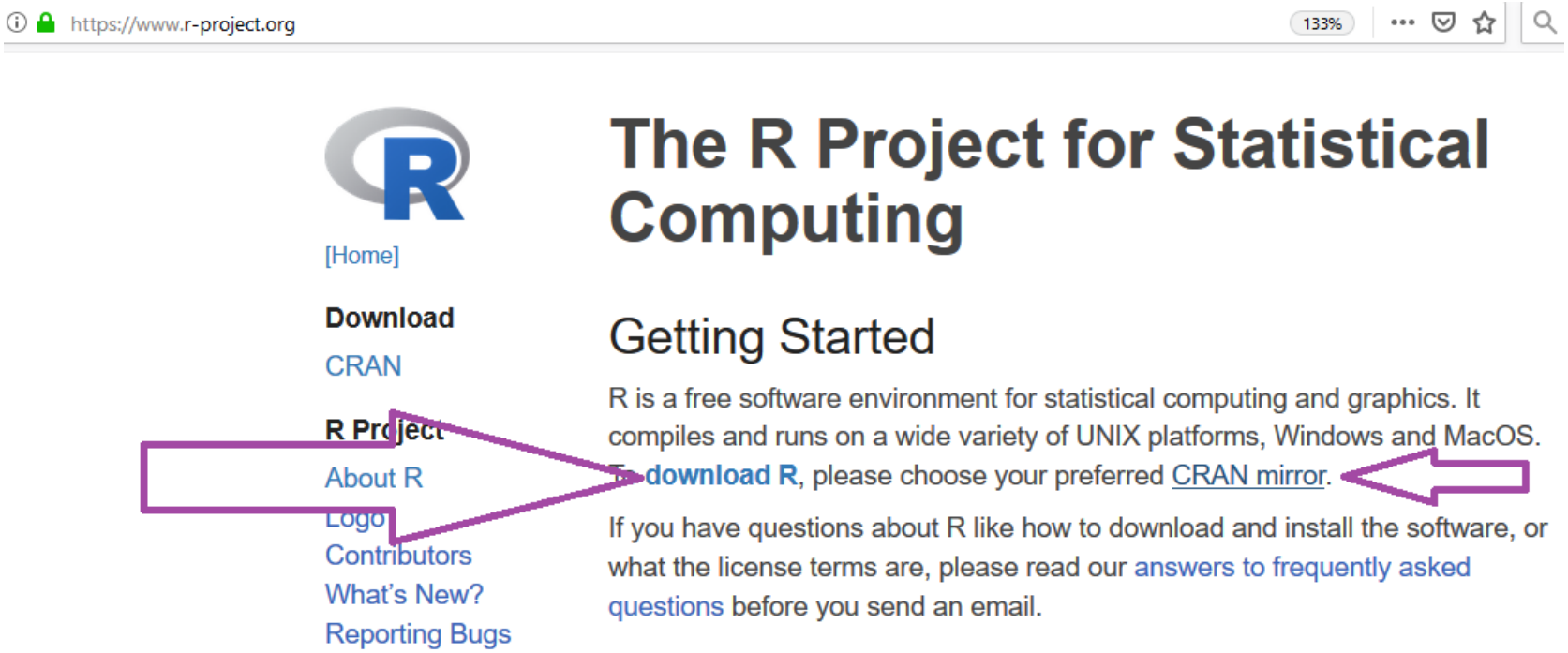
It is possible to contribute own packages.

The commands can be saved, run and stored in script files.

Graphics can be directly saved in a Postscript or PDF format.

Installing R

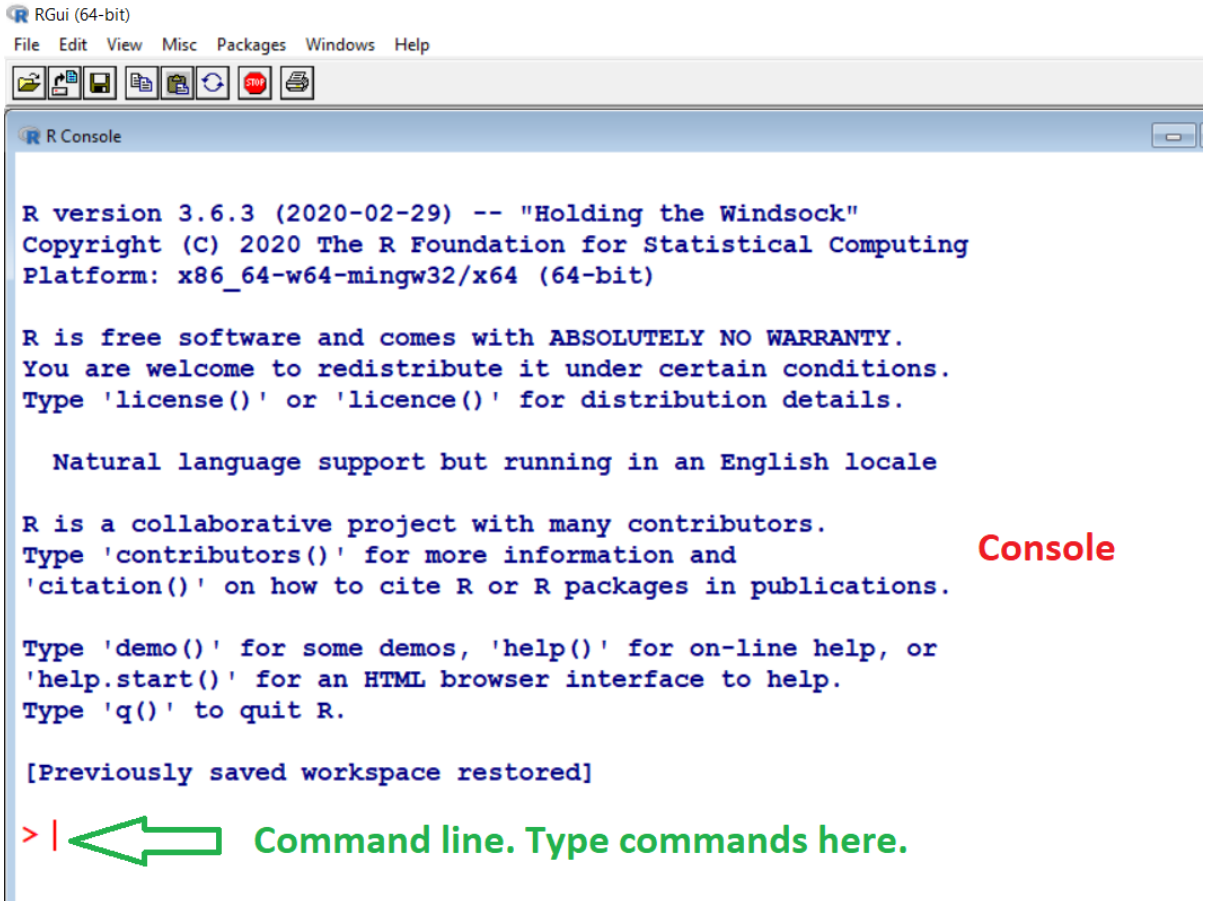
You may install R in a windows or Mac platform by downloading from the Comprehensive R Archive Network (CRAN) website: www.r-project.org or directly from http://cran.r-project.org/



Installing R

Icon will appear.

Double click on this icon will start the software.



Installing Packages and Libraries

The base R package contains programs for basic operations.

It does not contain some of the libraries necessary for advanced statistical work.

Specific requirements are met by special packages.

They are downloaded and their downloading is very simple.

Installing Packages and Libraries

The base R package contains some necessary libraries only.

Other libraries are required for advanced statistical work which are downloaded and installed as and when required.

Run the R program, then use the `install.packages` command to download the libraries.

Examples :

`install.packages("ggplot2")` **: installs package** `ggplot2`

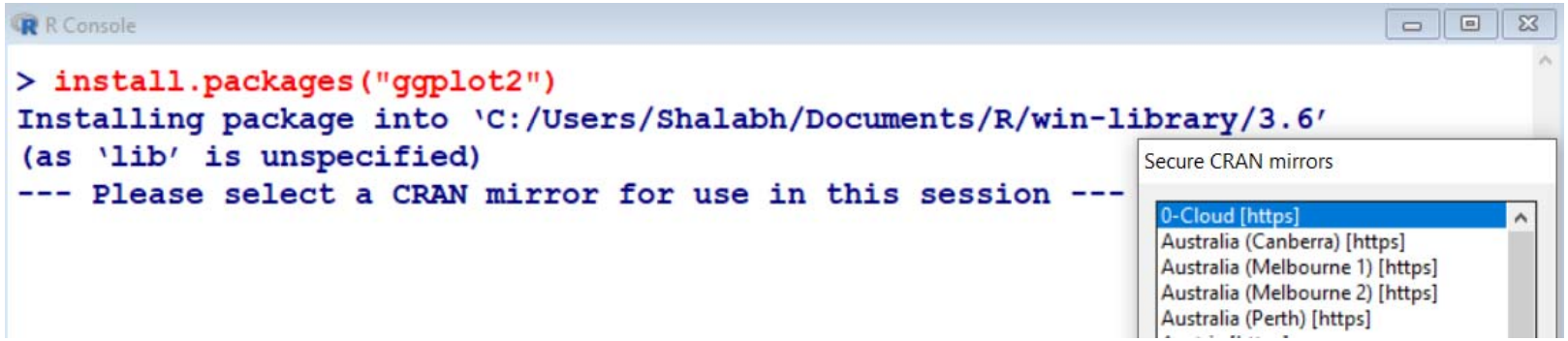
`install.packages("agricolae")`**: installs package** `agricolae`

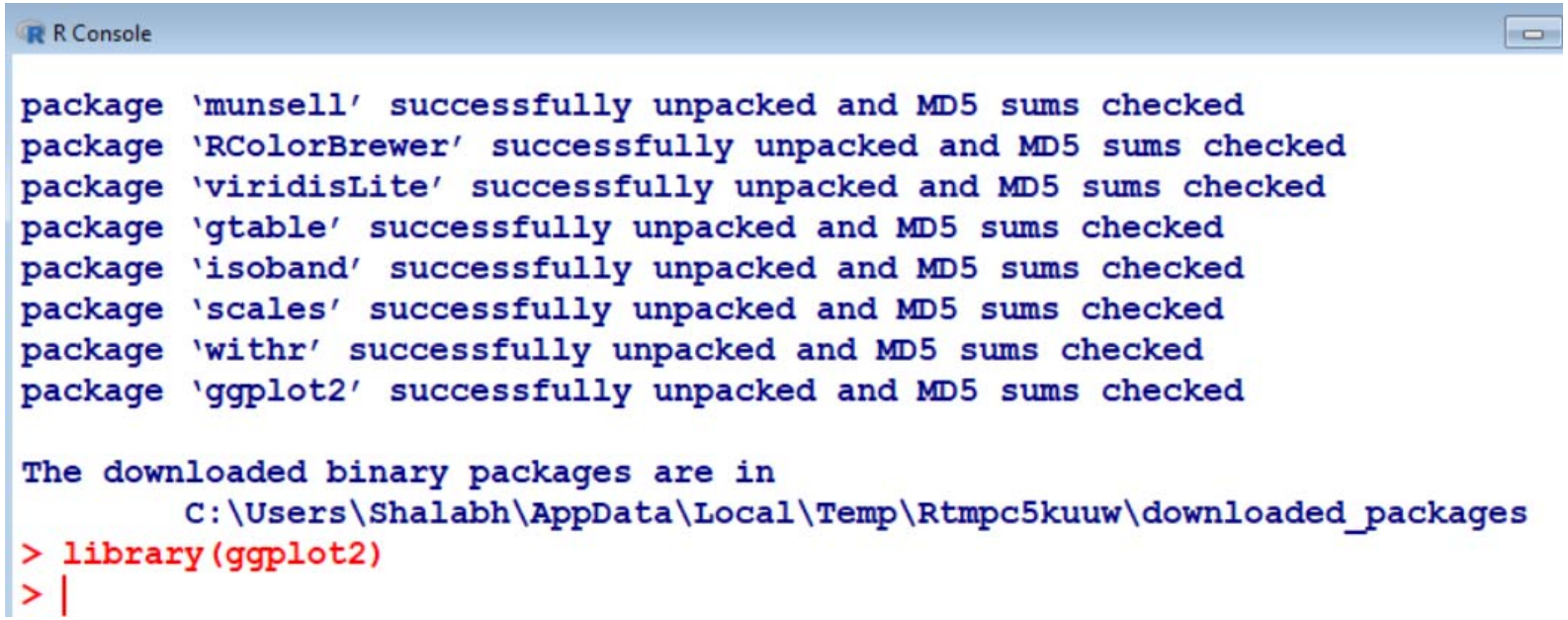
`install.packages("DoE.base")` **: installs package** `DoE.base`

Installing Packages and Libraries

Example

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





Libraries in R

To use a library, type the `library` function with the name of the library in brackets.

Thus to load the `ggplot2` library type:

```
library(ggplot2)
```

Similarly,

`library(agricolae)` : loads package `agricolae`

`library(DoE.base)`: loads package `DoE.base`

Libraries in R

Examples of libraries that come as a part of base package in R.

`MASS` : package associated with Venables and Ripley's book entitled *Modern Applied Statistics using S-Plus.*

`library(MASS)` loads package `MASS`

Contents of Libraries

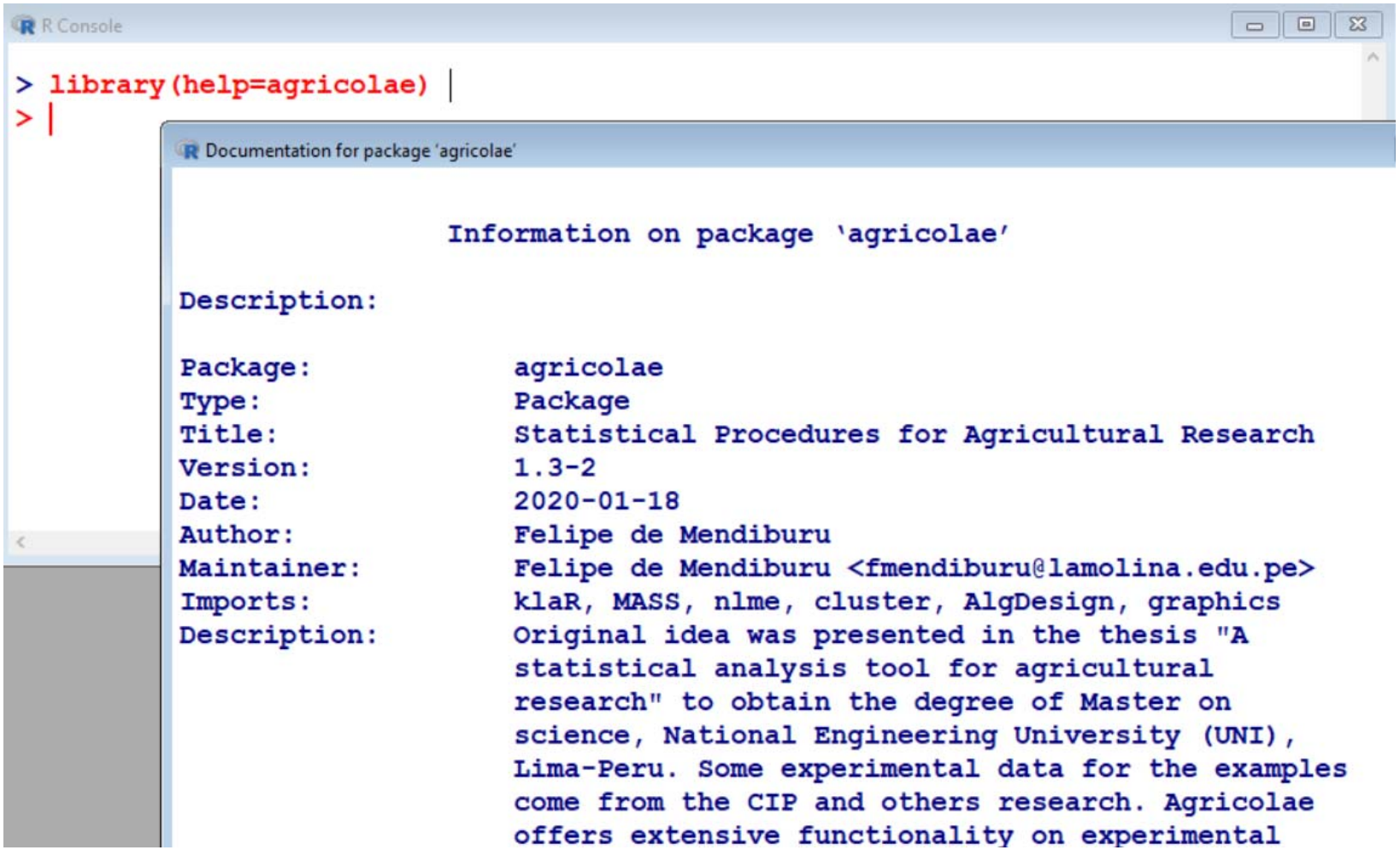
Use `help` function to get the detailed contents of library packages.

We find out about the contents of the `agricolae` library using `library(help=agricolae)` command

```
                Information on package 'agricolae'
Description:

Package:            agricolae
Type:               Package
Title:              Statistical Procedures for Agricultural Research
Version:            1.3-2
Date:               2020-01-18
Author:             Felipe de Mendiburu
Maintainer:         Felipe de Mendiburu <fmendiburu@lamolina.edu.pe>
Imports:            klaR, MASS, nlme, cluster, AlgDesign, graphics
Description:        Original idea was presented in the thesis "A
                    statistical analysis tool for agricultural … …
… … …
```

followed by a list of all the functions and data sets.

Contents of Libraries



Cleaning up the Windows

We assign names to variables when analyzing any data.

It is good practice to remove the variable names given to any data frame at the end each session in R.

`rm()` **command removes variable names**

For example,

`rm(x,y,z)` **removes the variables** `x`**,** `y` **and** `z`**.**

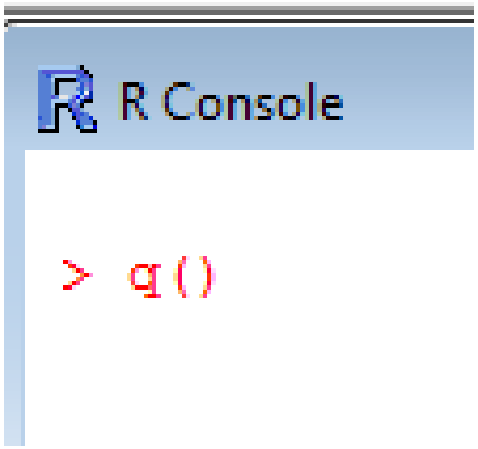
How to clear the screen in R

Press ctrl + L to clear the screen of R console.

and

How to quit in R

Type q( ) to quit R.



Working with R

Use command line to type and execute the commands.

Some free software like R Studio, Tinn R etc. are also available to work with R software.

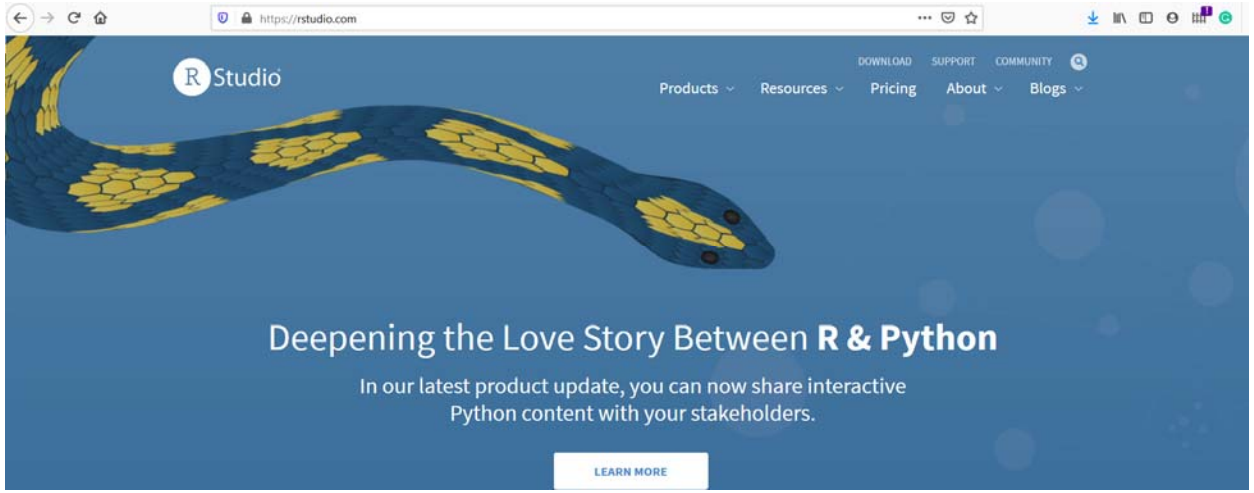
They are the interface between R and us and help in running the R software. Such software make coding and execution of programmes easier.

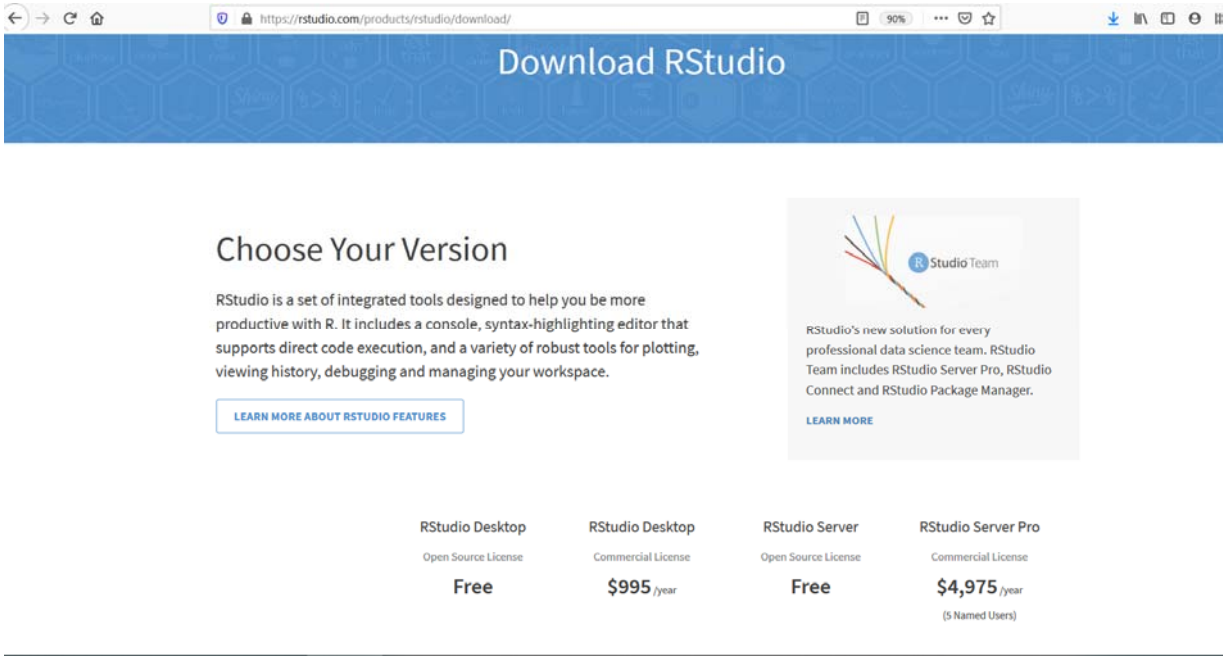
R Studio is available at https://www.rstudio.com/
Rstudio is written in C++ programming language.

Rstudio is a free and open-source integrated development environment (IDE) for R.

Tinn R is available at https://sourceforge.net/projects/tinn-r/

Installing R Studio

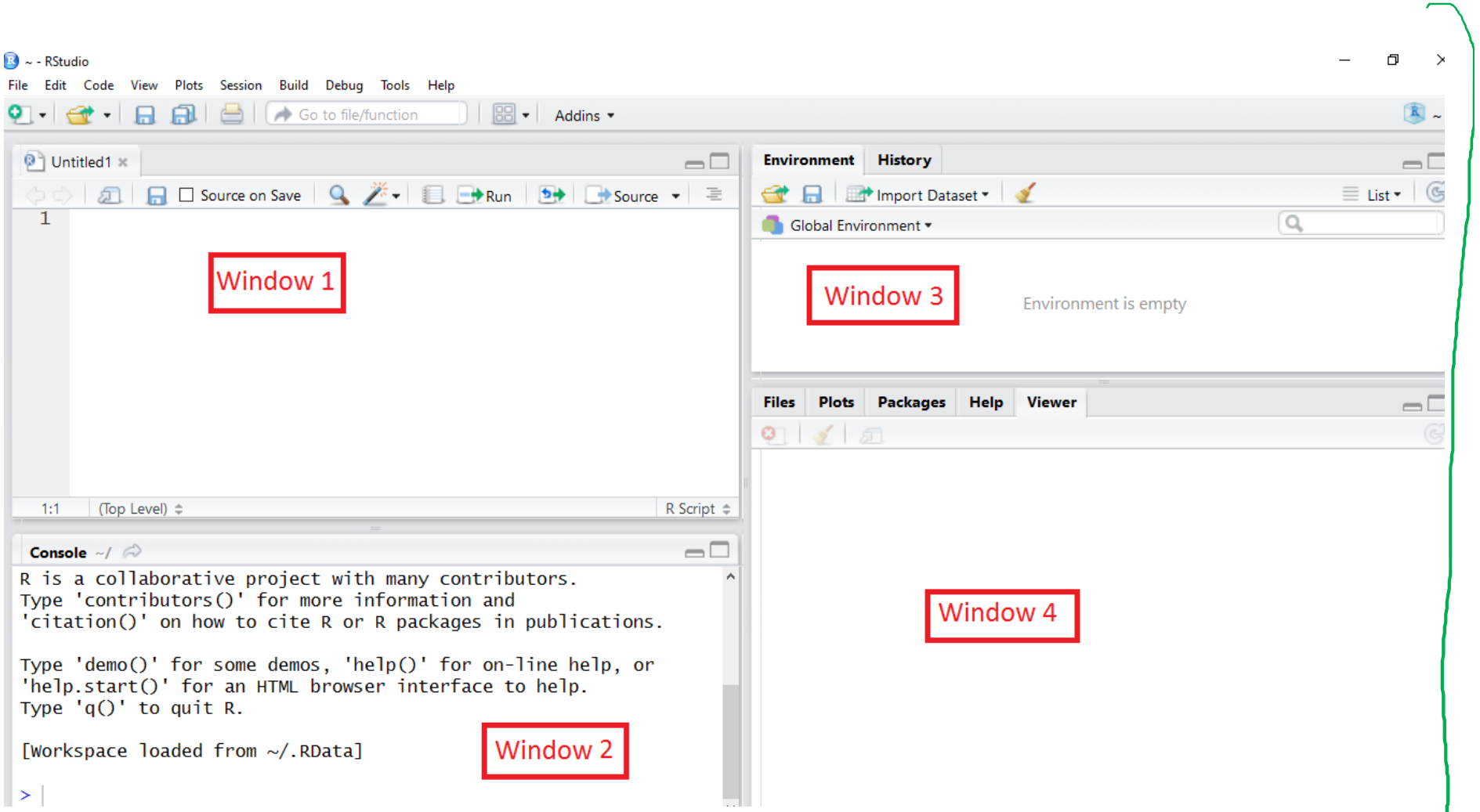




Download and double click on the downloaded file.

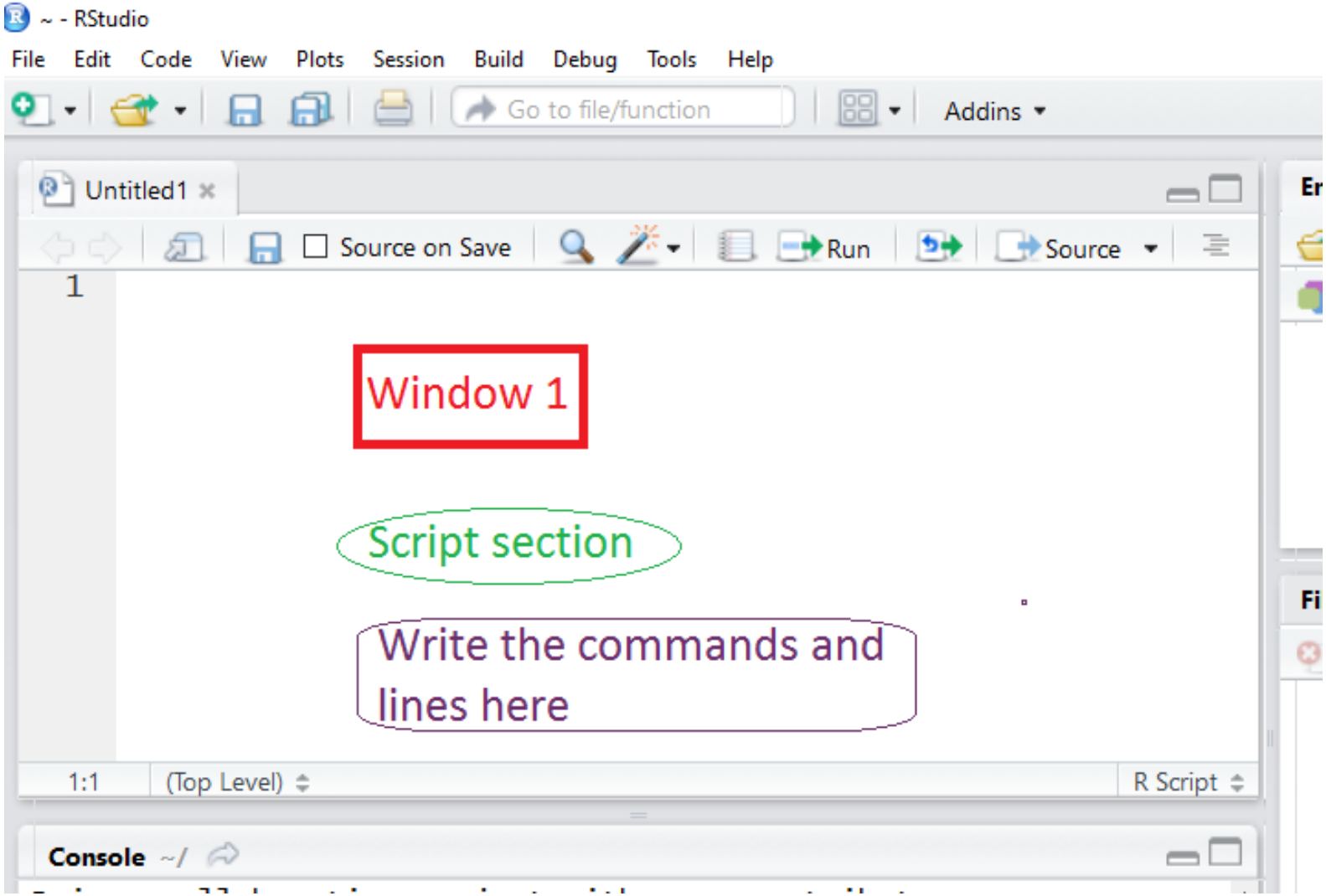
Introduction to R Studio

First opening window of Rstudio is as follows having four windows.



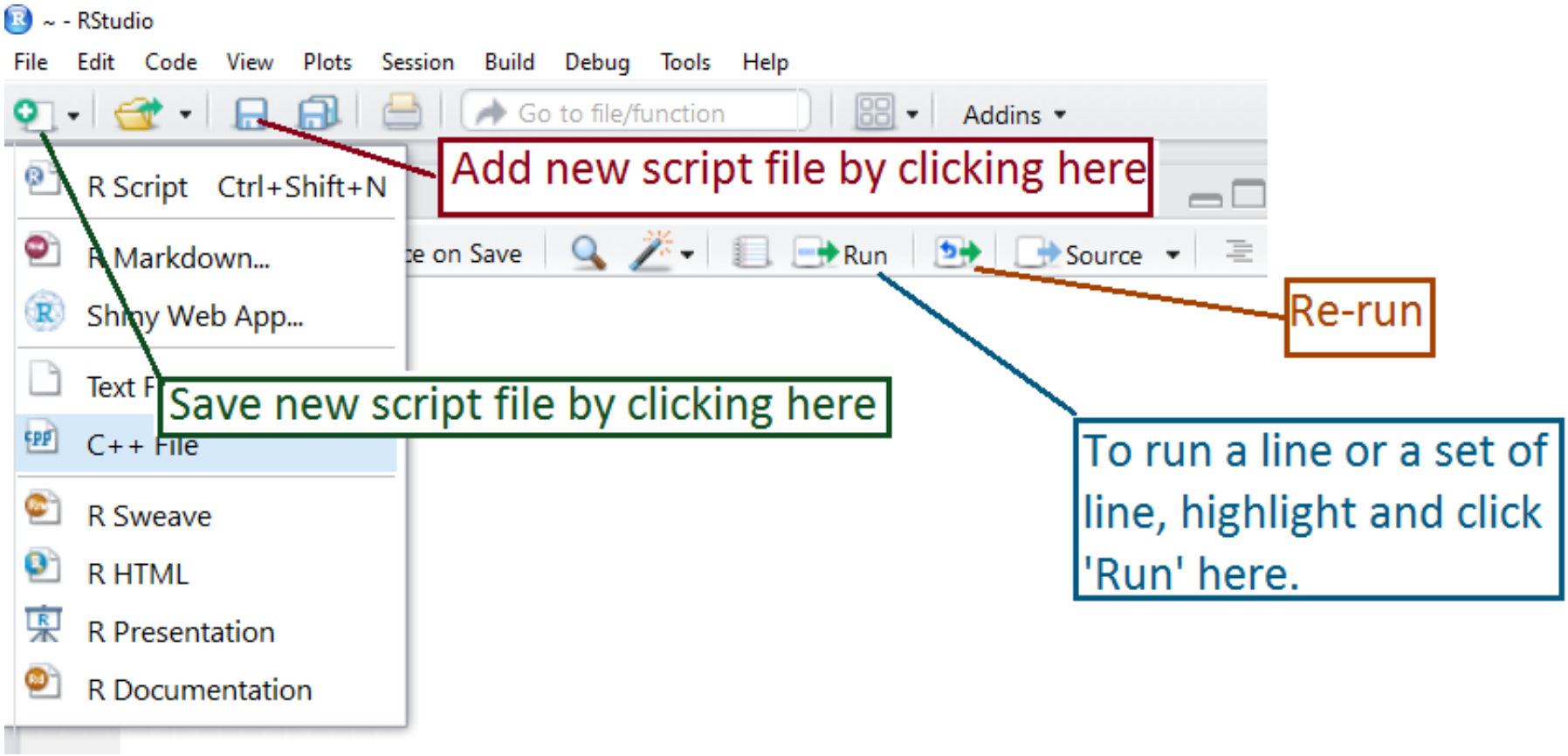
Introduction to R Studio

Description of Window 1



Introduction to R Studio

Description of Window 1



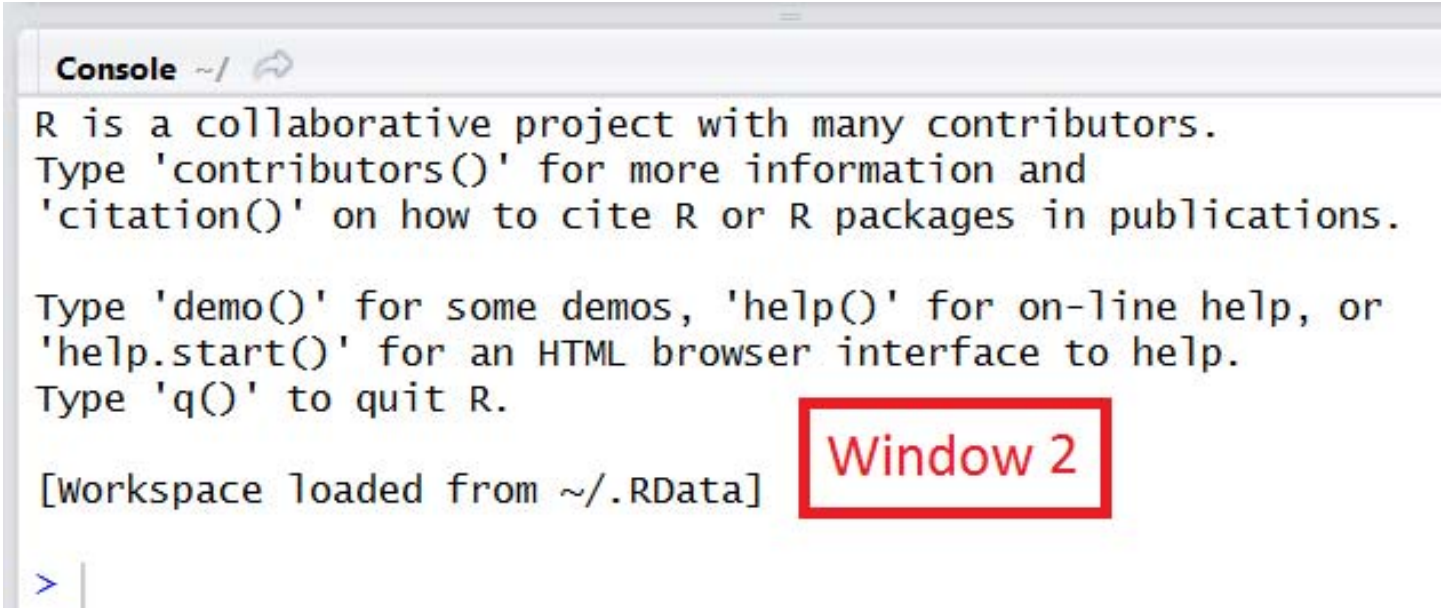
Introduction to R Studio

Description of Window 2 : Console

R program window appears here.

Calculations take place inconsole window.

One can write programmes in console also but it is hard to make corrections and experiments with the coding.



Introduction to R Studio

Description of Window 3 : Environment window

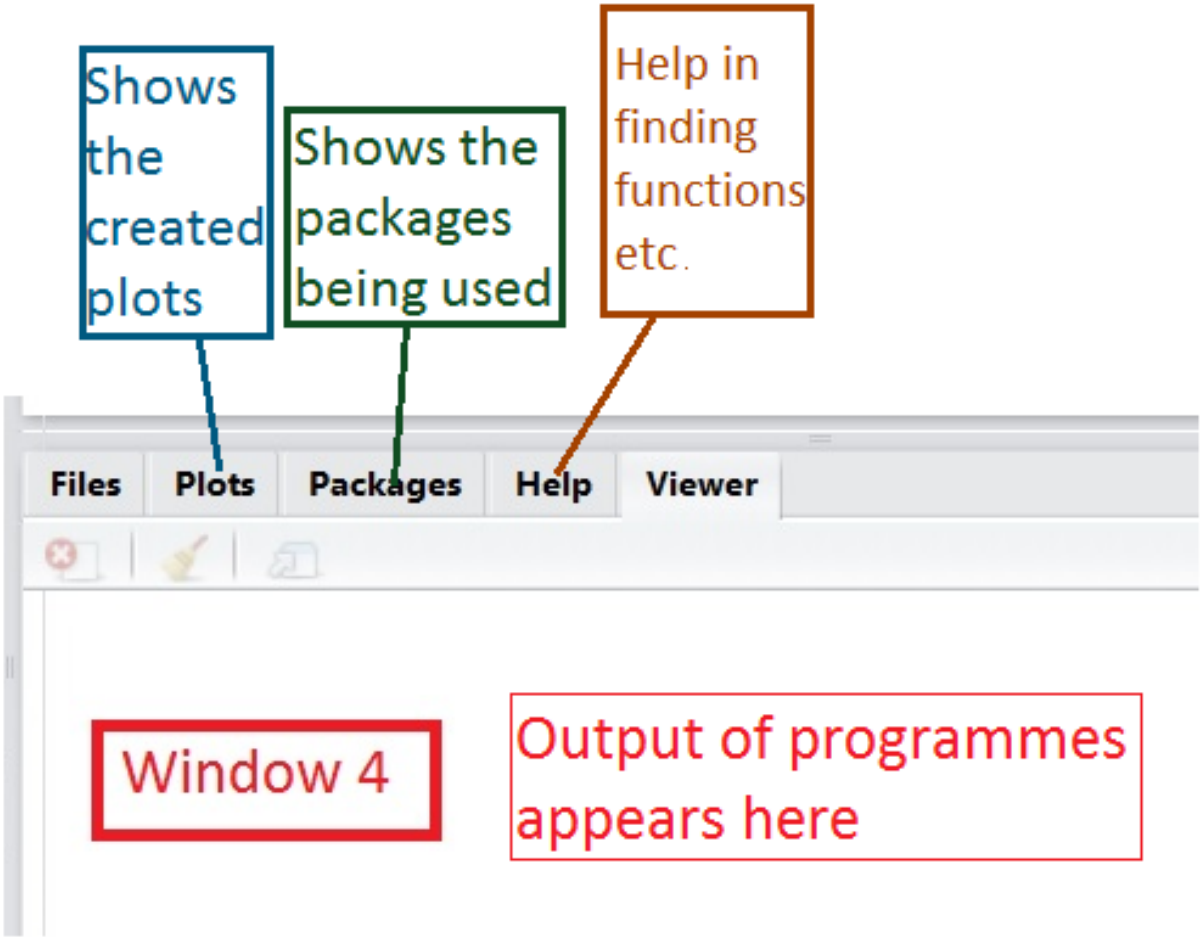
All the variables and objects used in the programme appear here.

The nature and values of variables and objects also appear here.

Introduction to R Studio

Description of Window 4 : Output window

The output of programmes appears in this window.



Introduction to R Studio

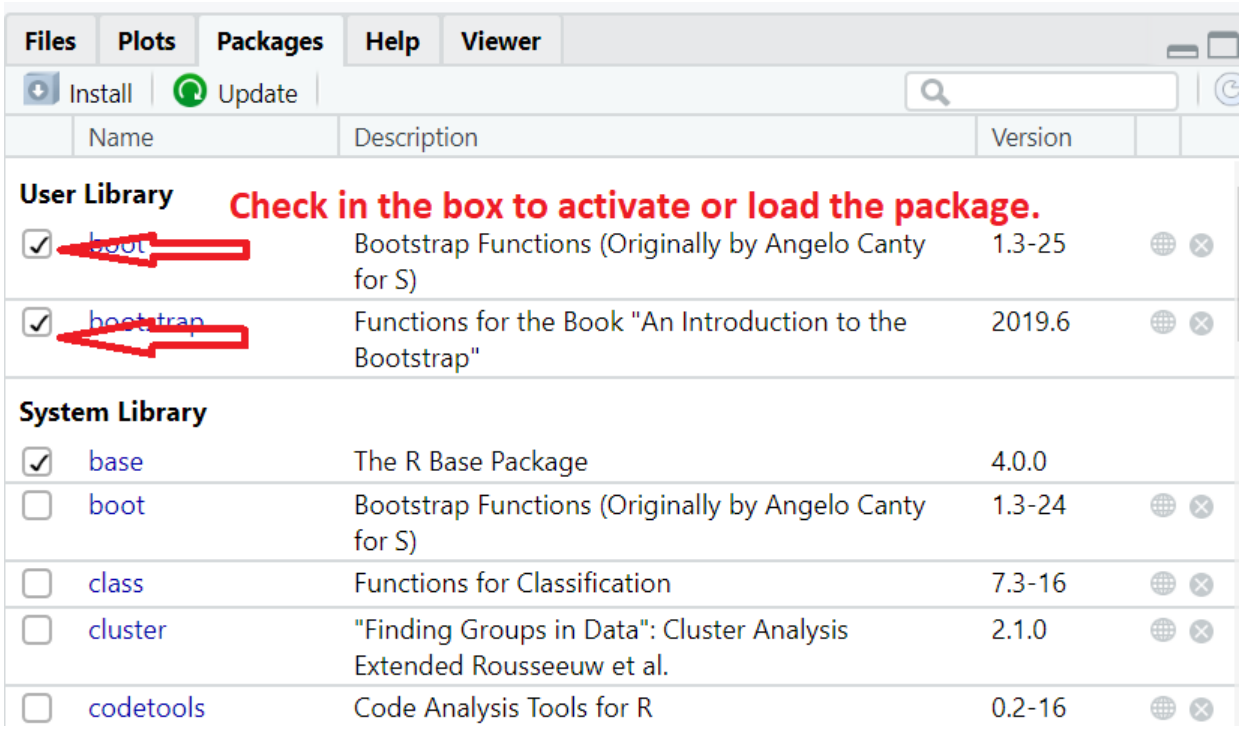
Description of Window 4 : Output window

Packages:

All the packages being installed appear here.

Packages are not active.

Check mark in the boxes to activate them.



Introduction to R Studio

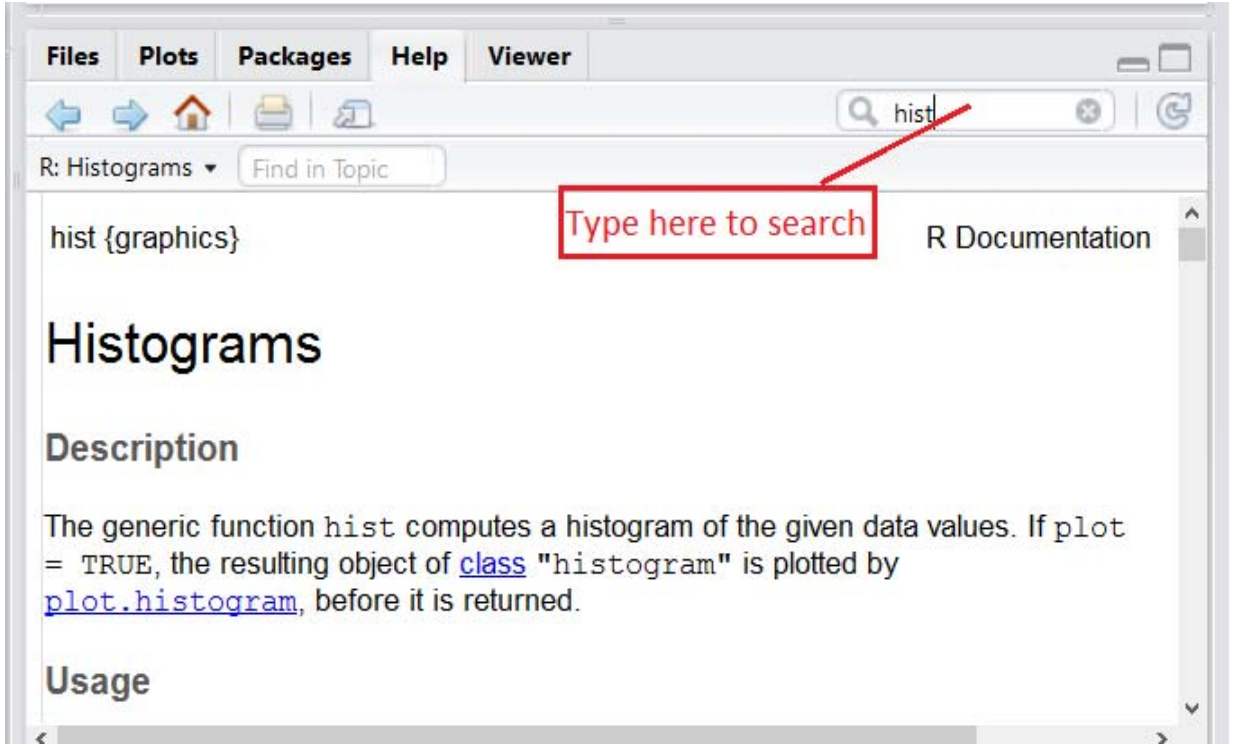
Window 4 : Output window

Help:

Various types of help can be asked.

E.g., to know about histogram, type hist.

Information appears.



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 3

Calculations with R as a Calculator

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

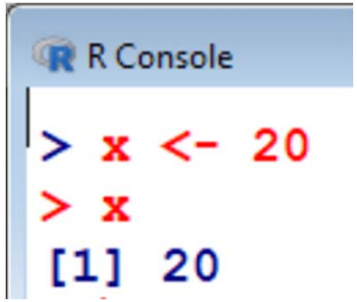
Basics

`>` is the prompt sign in R.

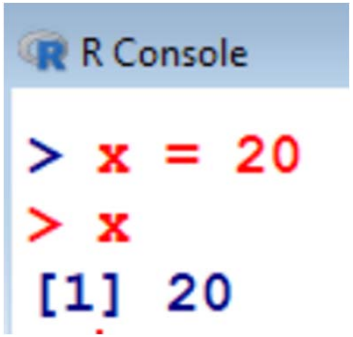
The assignment operators are the left arrow with dash `<-`

and equal sign `=`.

`> x <- 20` assigns the value 20 to `x`.



`> x = 20` assigns the value 20 to `x`.

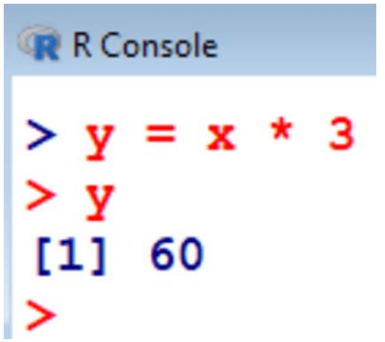


Initially only `<-` was available in R.

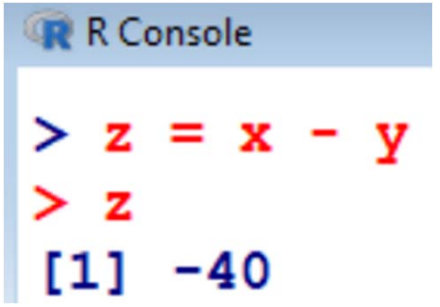
Basics

`> x = 20` assigns the value 20 to `x`.

`> y = 3 * x` assigns the value `3 * x` to `y`.



`> z = x - y` assigns the value `x - y` to `z`.



Basics

The command `c(1,2,3,4)` combines the numbers 1,2,3 and 4 to a vector.

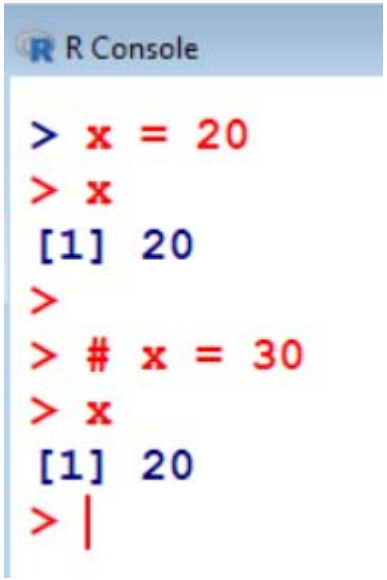
Basics

`#` : The character # marks the beginning of a comment.

All characters until the end of the line are ignored.

`> # mu` is the mean

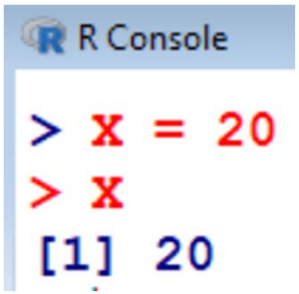
`> # x = 20` is treated as comment only

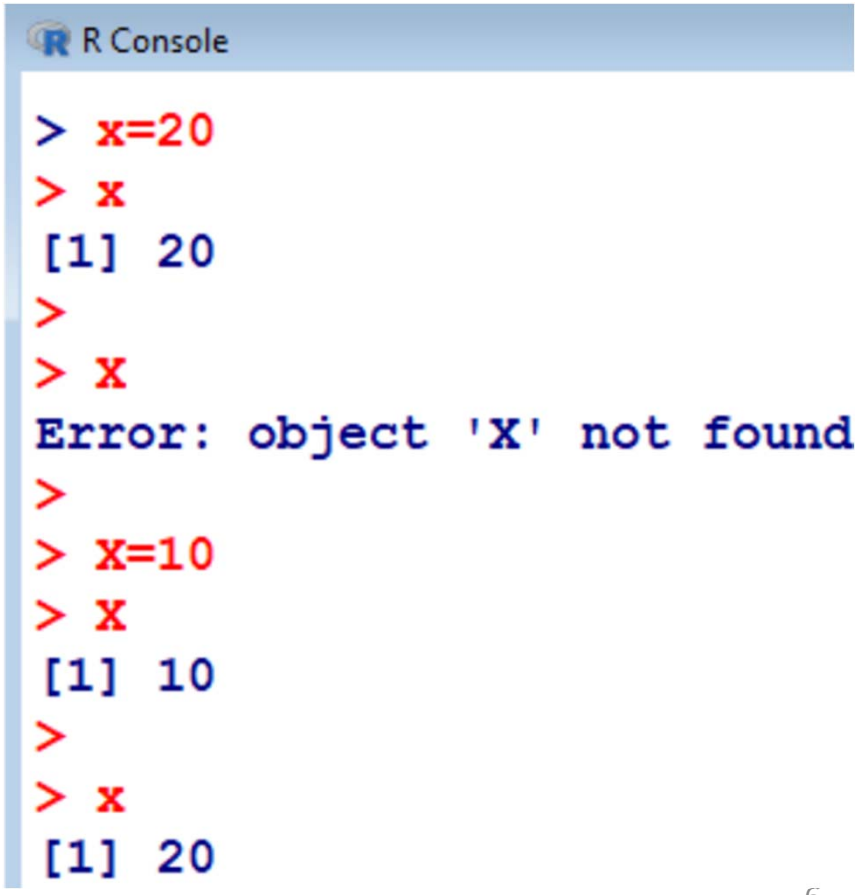


Basics

Capital and small letters are different.

`> X = 20` and `> x = 20` are different

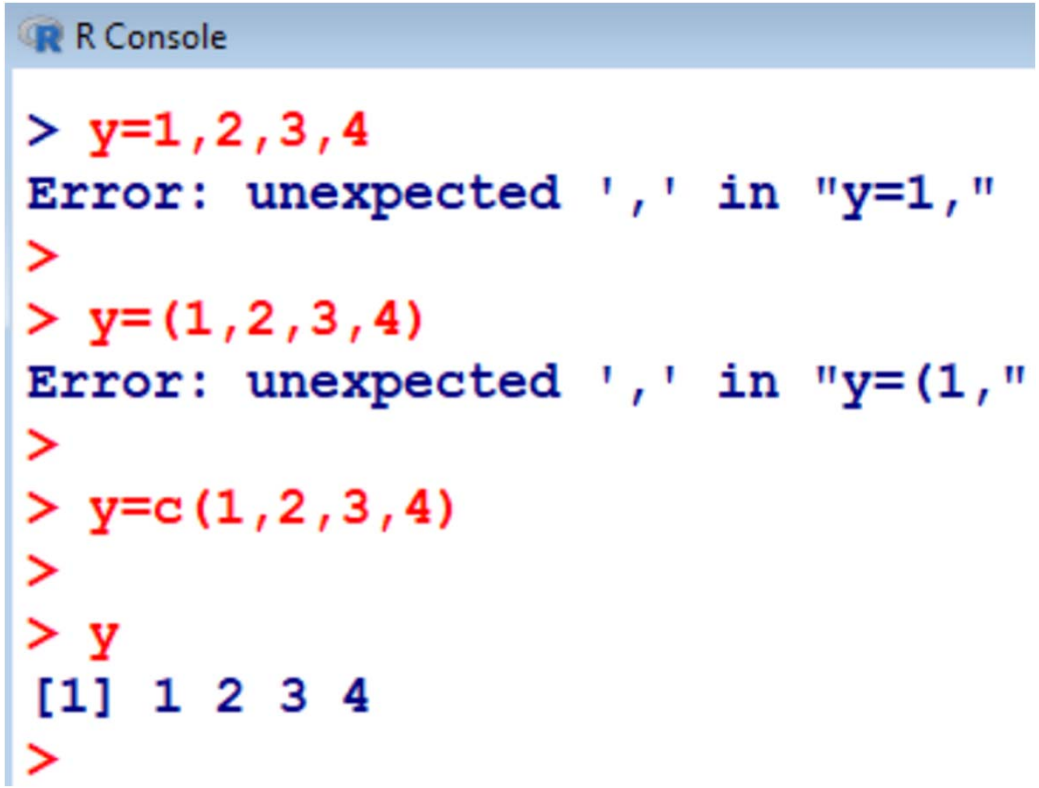




Basics

The command `c(1,2,3,4)` combines the numbers 1,2,3 and 4 to a vector.

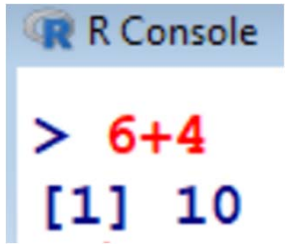
The command `c(1,2,3,4)` combines the numbers 1,2,3 and 4 to a vector.



R as a calculator

Addition

```
> 6+4          # Command
[1] 10         # Output
```



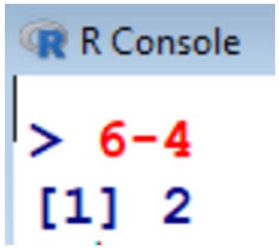
Multiplication

```
> 6*4          # Command
[1] 24         # Output
```

R as a calculator

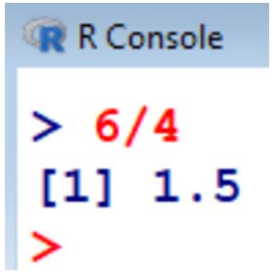
Subtraction

```
> 6-4          # Command
[1] 2          # Output
```



Division

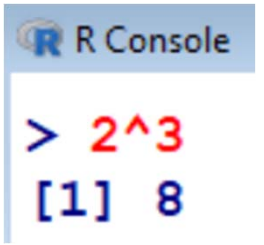
```
> 6/4          # Command
[1] 1.5        # Output
```



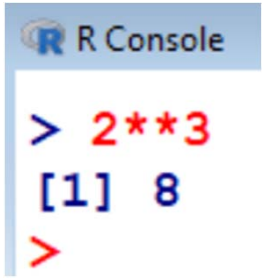
R as a calculator

Power

```
> 2^3              # Command
[1] 8              # Output
```




```
> 2**3             # Command
[1] 8              # Output
```

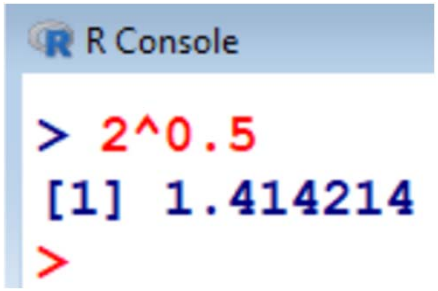


2^3

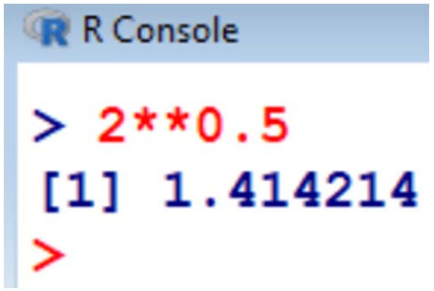
R as a calculator

Power

```
> 2^0.5          # Command
[1] 1.732051     # Output
```




```
> 2**0.5         # Command
[1] 1.732051     # Output
```

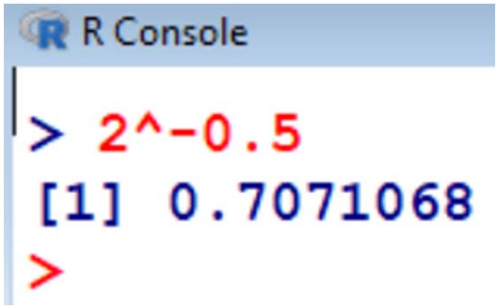


$2^{1/2}$

R as a calculator

Power

```
> 2^-0.5        # Command
[1] 0.5773503 # Output
```

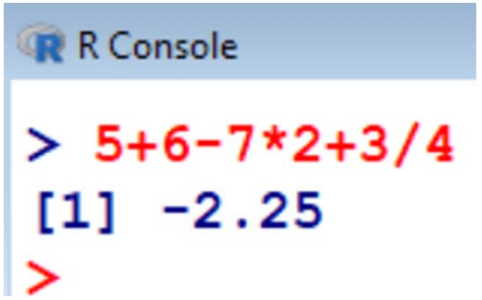


$2^{-1/2}$

Multiple operators (BODMAS)

Bracket, Of, Division, Multiplication, Addition, and Subtraction

```
> 5+6-7*2+3/4   # Command
[1] -2.25         # Output
```



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 4

Calculations with Data Vectors

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Calculations with data vectors

How R behaves with data vectors?

What happens when a scalar is added/subtracted/multiplied/ divided in a data vector?

R as a calculator

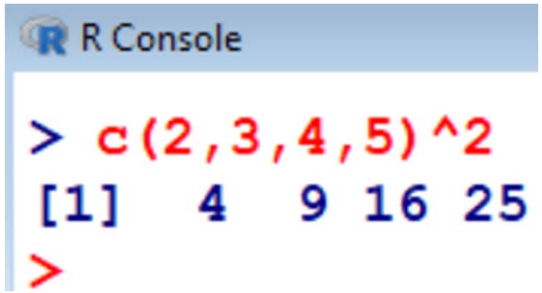
Power operators with vector versus scalar

```
> c(2,3,4,5)^2
[1] 4 9 16 25
```

 # command: application to a vector

 # output

$2^2, 3^2, 4^2, 5^2$

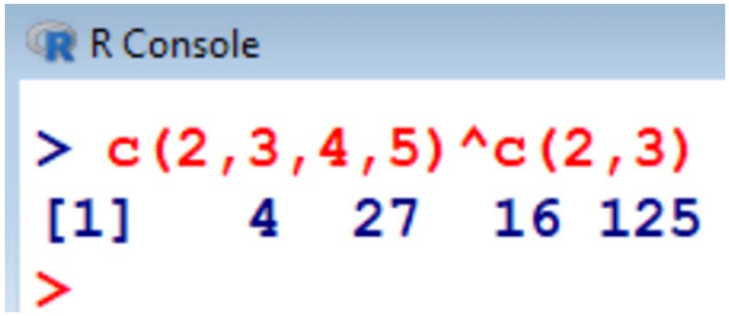


R as a calculator

Power operators with vector versus vector

```
> c(2,3,4,5)^c(2,3)   # !!ATTENTION! Observe the
                        operation
[1]  4 27 16 125       # output
```

$2^2, 3^3, 4^2, 5^3$



R as a calculator

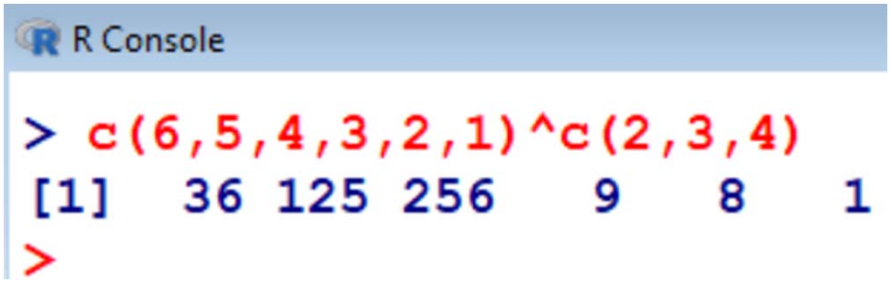
Power operators with vector versus vector

```
> c(6,5,4,3,2,1)^c(2,3,4)
[1]  36 125 256 9 8 1
```

 # command: application to a vector with vector

 # output

$6^2, 5^3, 4^4, 3^2, 2^3, 1^4$

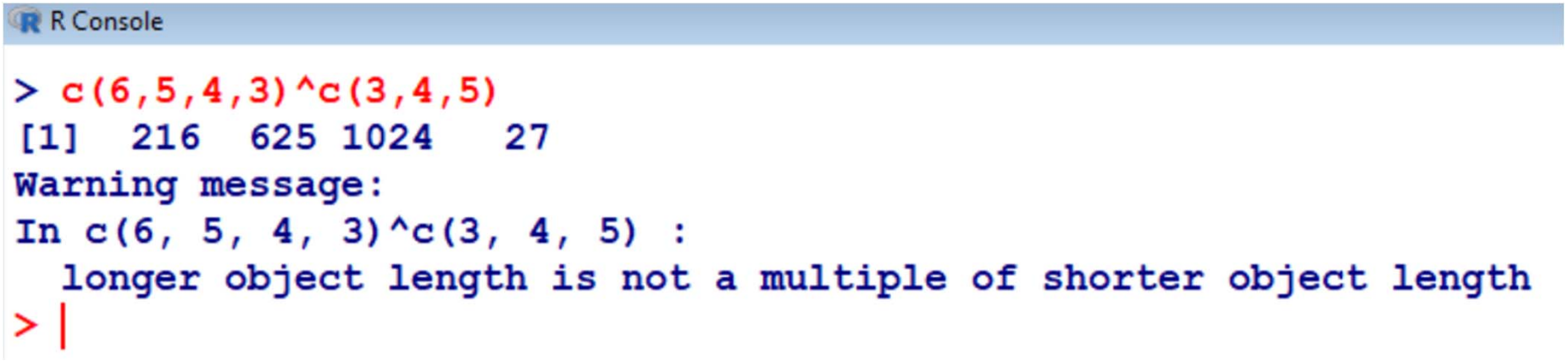


R as a calculator

Power operators with vector versus vector

```
> c(6,5,4,3)^c(3,4,5)      #Warning message
[1] 216  625 1024   27      # output
Warning message:
In c(6,5,4,3)^c(3,4,5) :longer object length is
not a multiple of shorter object length
```

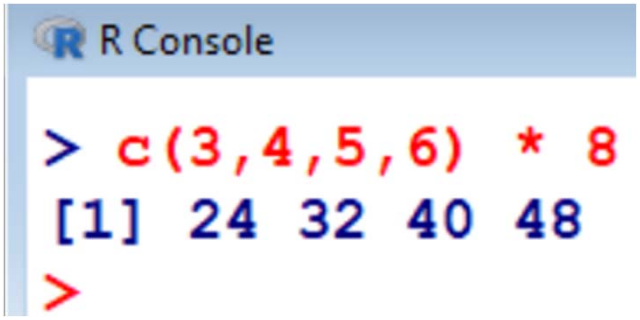
$6^3, 5^4, 4^5, 3^3$



R as a calculator

Multiplication with vector versus scalar

```
> c(3,4,5,6) * 8
[1] 24 32 40 48
```



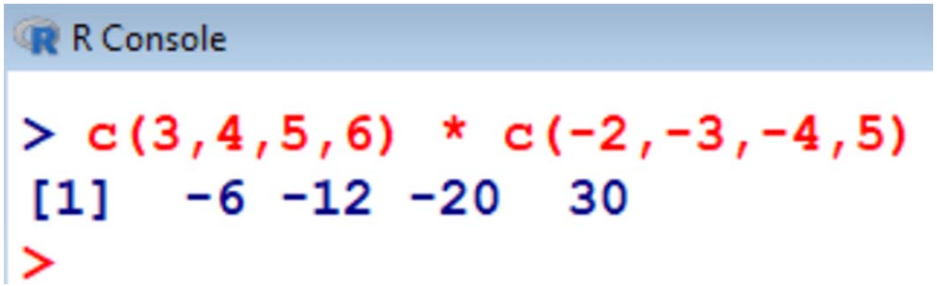
3x8, 4x8, 5x8, 6x8

R as a calculator

Multiplication with vector versus vector

```
> c(3,4,5,6) * c(-2,-3,-4,5)
[1] -6  -12 -20  30
```

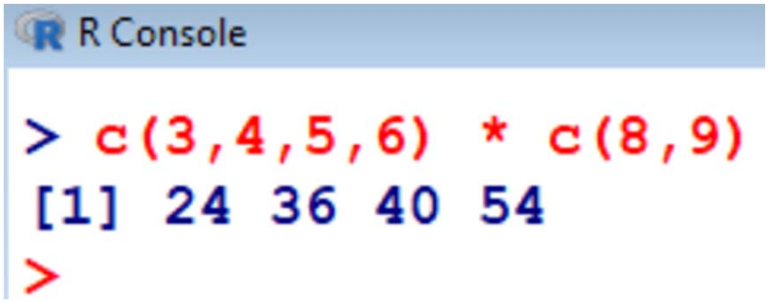
3x(-2), 4x(-3), 5x(-4), 6x5



R as a calculator

Multiplication with vector versus vector

```
> c(3,4,5,6) * c(8,9)
[1] 24 36 40 54
```



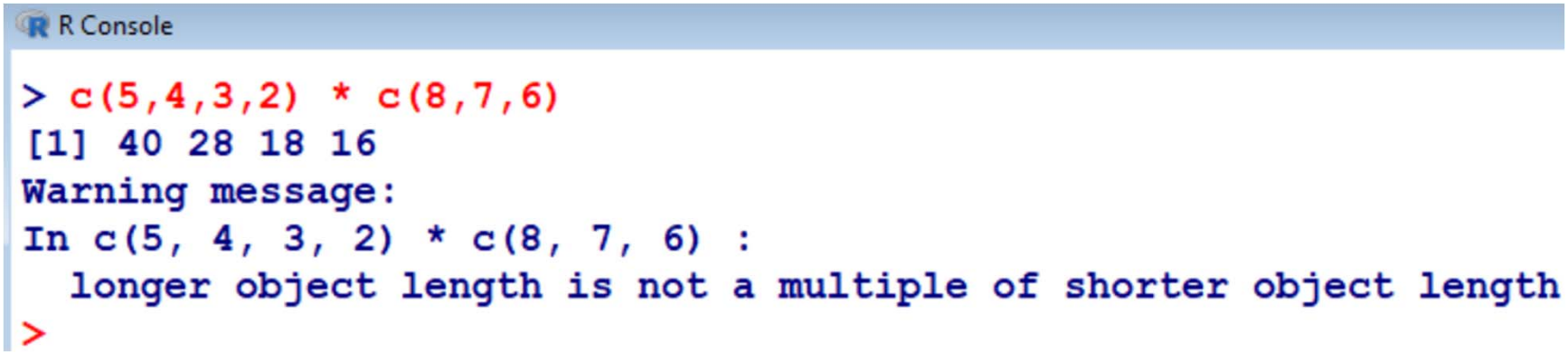
3x8, 4x9, 5x8, 6x9

R as a calculator

Multiplication with vector versus vector

```
> c(5,4,3,2) * c(8,7,6) # Warning message
[1] 40 28 18 16
 Warning message:
In c(5,4,3,2) * c(8,7,6) :longer object length
is not a multiple of shorter object length
```

5x8, 4x7, 3x6, 2x8

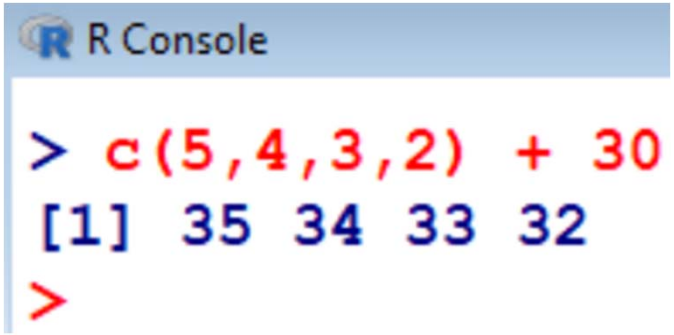


R as a calculator

Addition with vector versus scalar

```
> c(5,4,3,2) + 30
[1] 35 34 33 32
```

5+30, 4+30, 3+30, 2+30

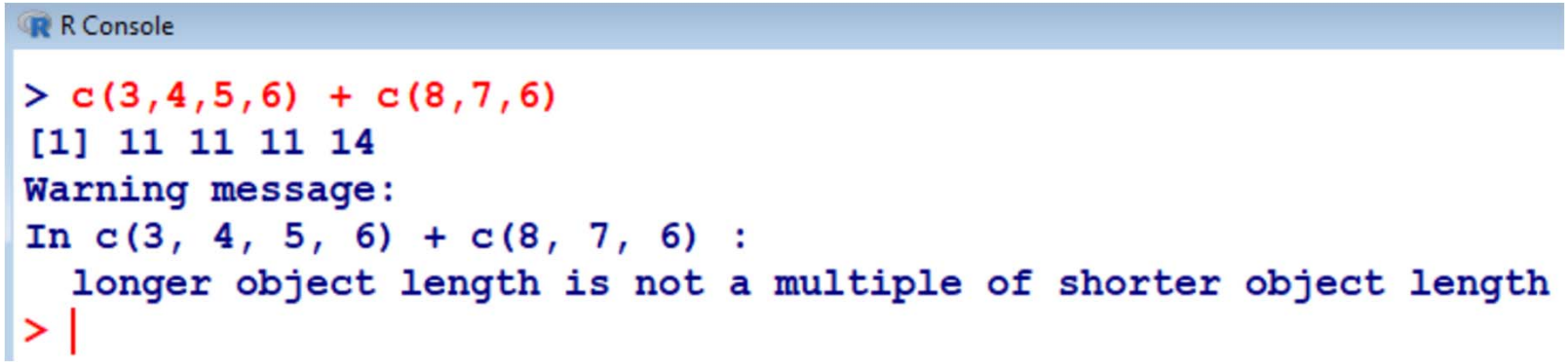


R as a calculator

Addition with vector versus vector

```
> c(3,4,5,6) + c(8,7,6) # Warning message
[1] 11 11 11 14
Warning message:
In c(3,4,5,6) + c(8,7,6) :
  longer object length is not a multiple of
shorter object length
```

3+8, 4+7, 5+6, 6+8



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 5

Built-in Commands and Missing Data Handling

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

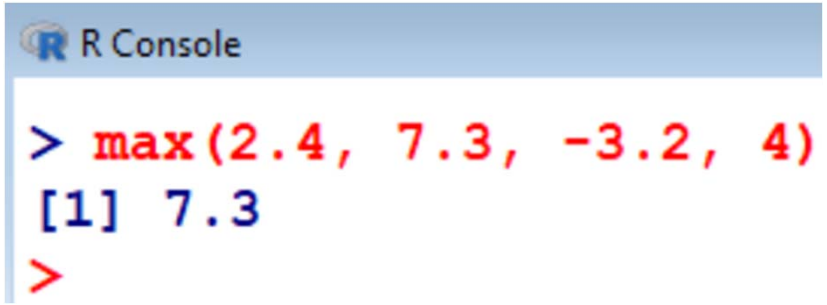
Built in commands

Some commands are readily available in R to compute the mathematical functions.

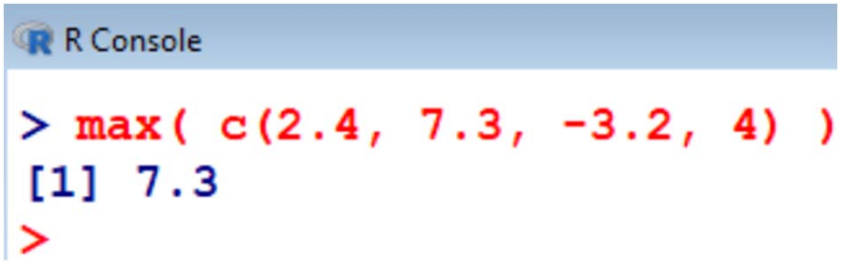
How to use them and utilize them in computing various quantities?

Maximum

```
> max(2.4, 7.3, -3.2, 4)
[1] 7.3
```

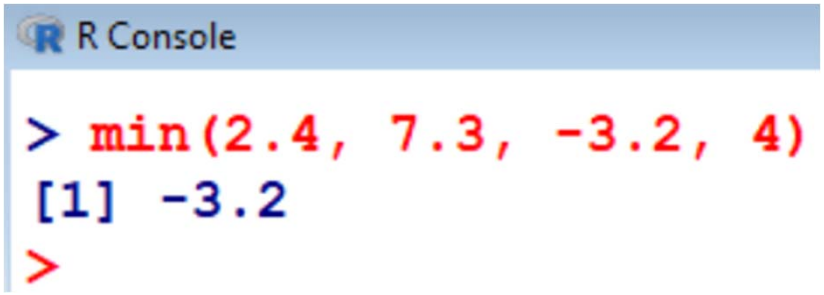



```
> max( c(2.4, 7.3, -3.2, 4) )
[1] 7.3
```

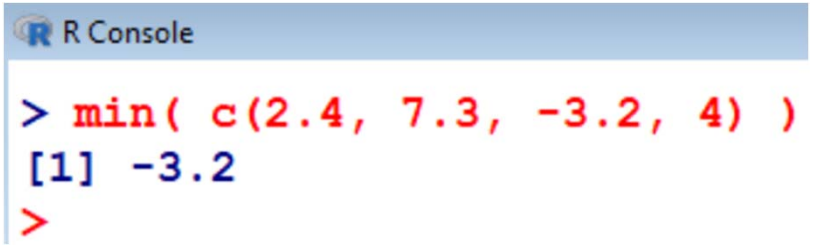


Minimum

```
> min(2.4, 7.3, -3.2, 4)
[1] -3.2
```




```
> min( c(2.4, 7.3, -3.2, 4) )
[1] -3.2
```

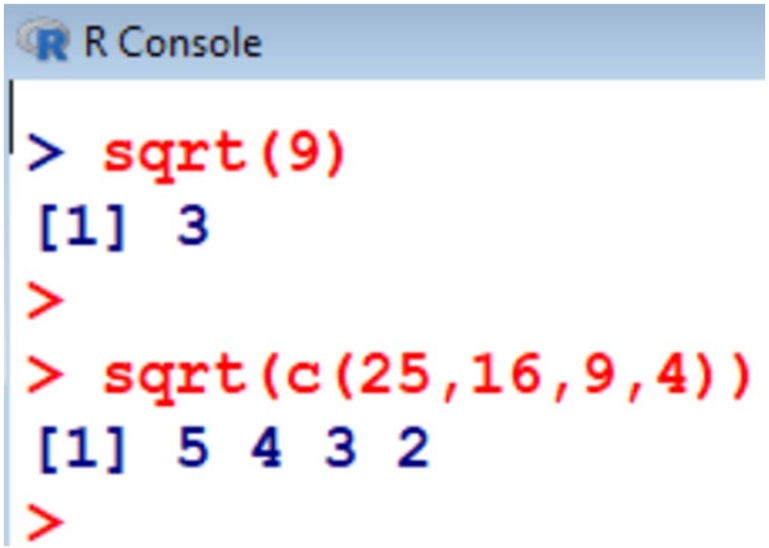


Overview Over Other Functions

`abs()`	Absolute value
`sqrt()`	Square root
`round(), floor(), ceiling()`	Rounding, up and down
`sum(), prod()`	Sum and product
`log(), log10(), log2()`	Logarithms
`exp()`	Exponential function
`sin(), cos(), tan(),` `asin(), acos(), atan()`	Trigonometric functions
`sinh(), cosh(), tanh(),` `asinh(), acosh(), atanh()`	Hyperbolic functions

Examples

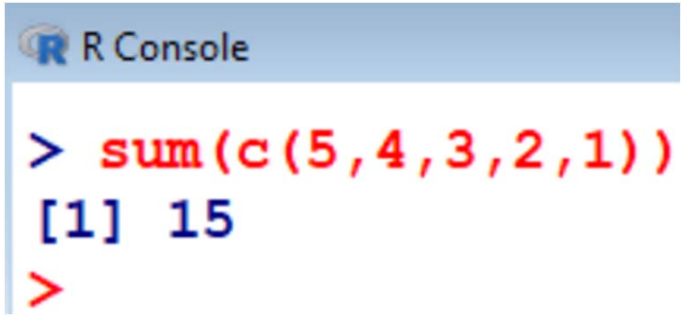
```
> sqrt(9)
[1] 3
```



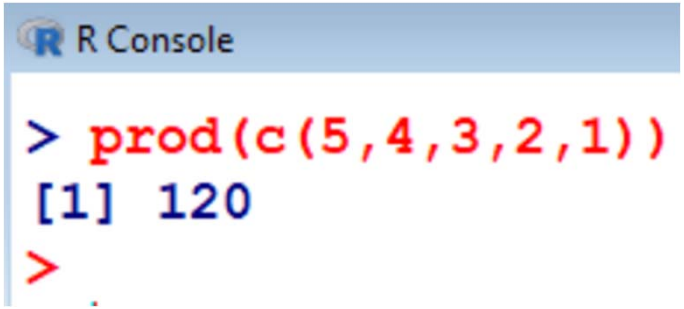

```
> sqrt(c(25,16,9,4))
[1] 5 4 3 2
```

Examples

```
> sum(c(5,4,3,2,1))
[1] 15
```



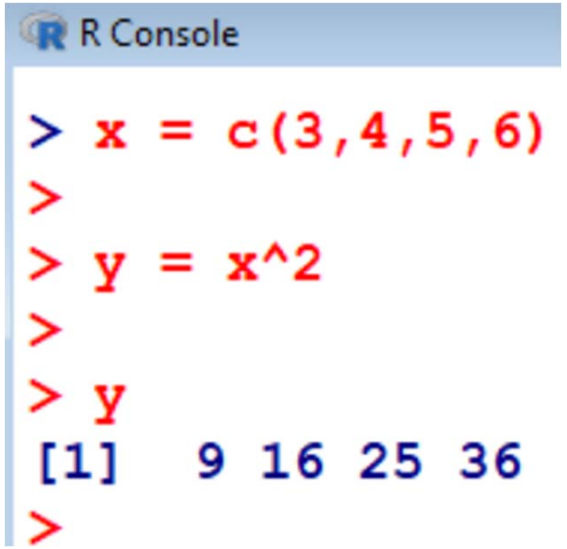

```
> prod(c(5,4,3,2,1))
[1] 120
```



Assignments

An assignment can also be used to save values in variables:

```
> x = c(3,4,5,6)

> y = x^2

> y
[1] 9 16 25 36
```



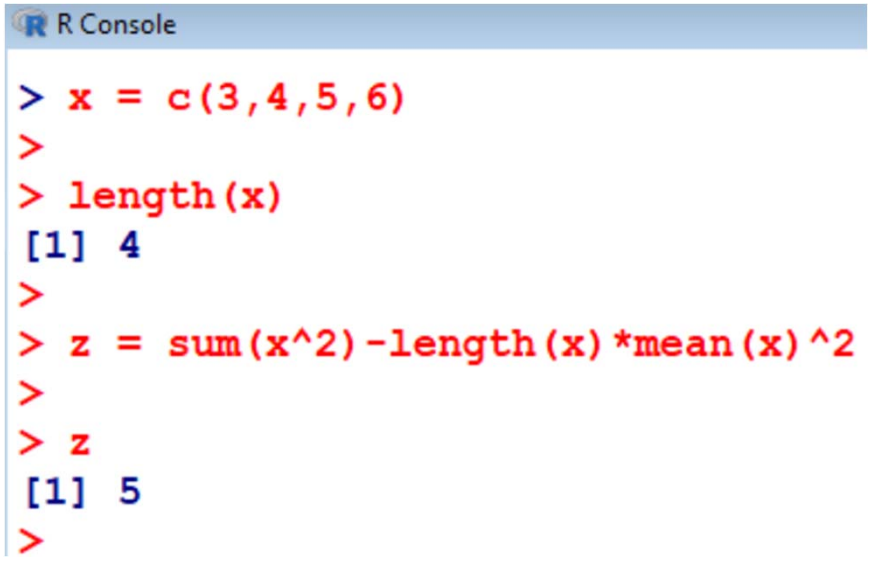
Examples

To find sum of squares of deviation from mean

```
> x = c(3,4,5,6)

> length(x)
[1] 4

> z = sum(x^2)-length(x)*mean(x)^2
```

$$z = \sum_{i=1}^{n} (x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2$$

```
> z
[1] 5
```



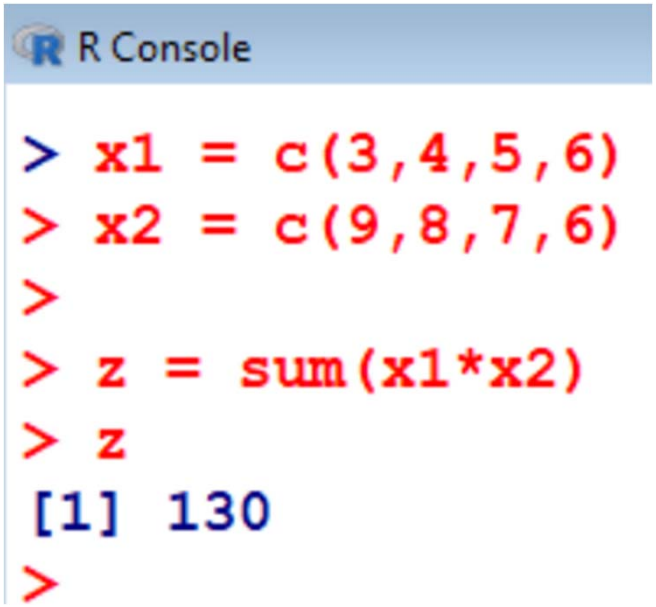
Examples

To find sum of cross product:

```
> x1 = c(3,4,5,6)
> x2 = c(9,8,7,6)
```

3x9 + 4x8 + 5x7 + 6x6

```
> z = sum(x1*x2)
> z
[1] 130
```



Missing data

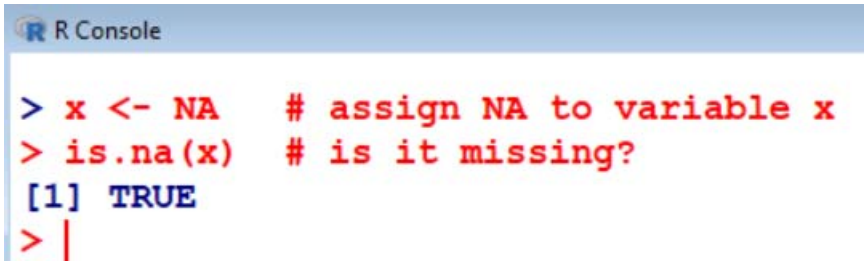
R represents missing observations through the data value NA

NA : Reserved word

Missing values can be detected in a data vector by using is.na

Create a data with NA.

```
> x <- NA      # assign NA to variable x
> is.na(x)     # is it missing?
 [1] TRUE
```



Logical operators

TRUE and FALSE are logical operators that are used to compare expressions.

TRUE and FALSE are reserved words.

T can also be used in place of TRUE .

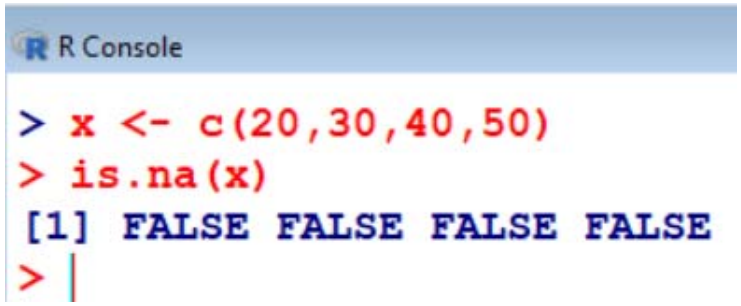
F can also be used in place of FALSE.

TRUE and FALSE are not the same as `true` and `false` respectively.

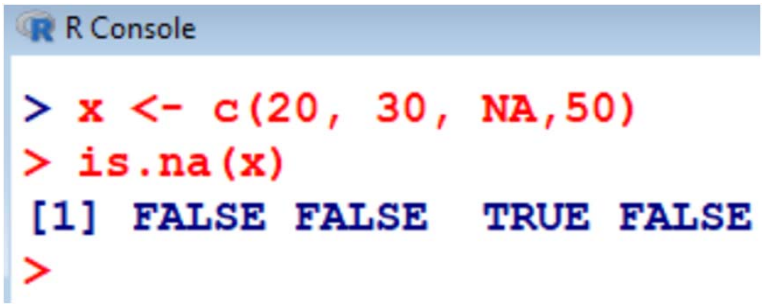
Missing data

How to know if any value is missing in a data vector?

```
> x <- c(20,30,40,50)
> is.na(x)
 [1] FALSE FALSE FALSE FALSE
```



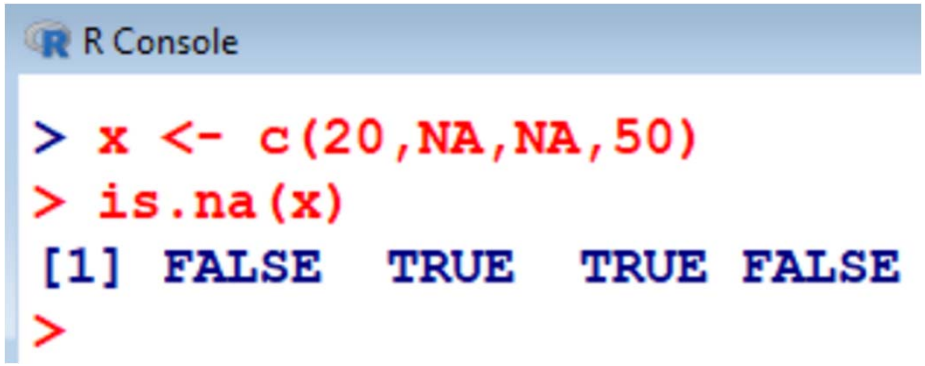

```
> x <- c(20, 30, NA,50)
> is.na(x)
 [1] FALSE FALSE TRUE FALSE
```



Missing data

How to know if any value is missing in a data vector?

```
> x <- c(20,NA,NA,50)
> is.na(x)
 [1] FALSE  TRUE TRUE FALSE
```



Example : How to work with missing data

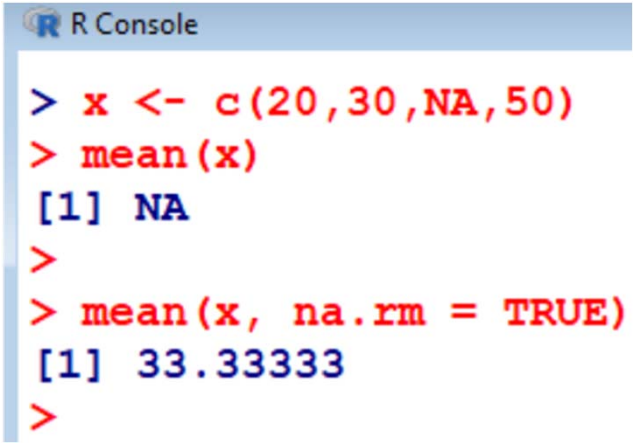
```
> x <- c(20,30,NA,50)
```
data vector

```
> mean(x)
 [1] NA
```

$$\frac{20 + 30 + NA + 50}{4}$$

```
> mean(x, na.rm = TRUE)
 [1] 33.33333
```
NAs can be removed

$$\frac{20+30+50}{3} = 33.33$$

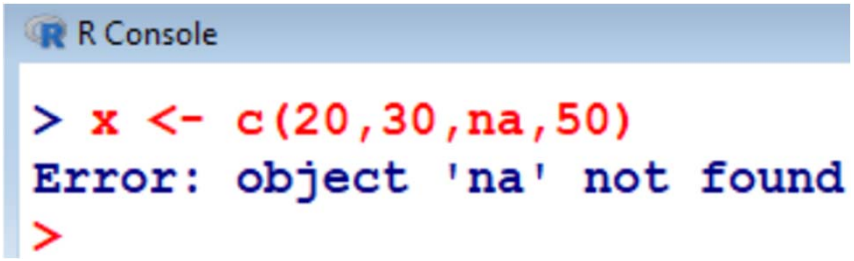


Example : How to work with missing data

The null object, called NULL, is returned by some functions and expressions.

Note that NA and NULL are not the same.

Note that NA and na are not the same.



NA is a placeholder for something that exists but is missing.

NULL stands for something that never existed at all.

Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 6

Operations with Matrices

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Matrix

A matrix is a rectangular array with *p* rows and *n* columns.

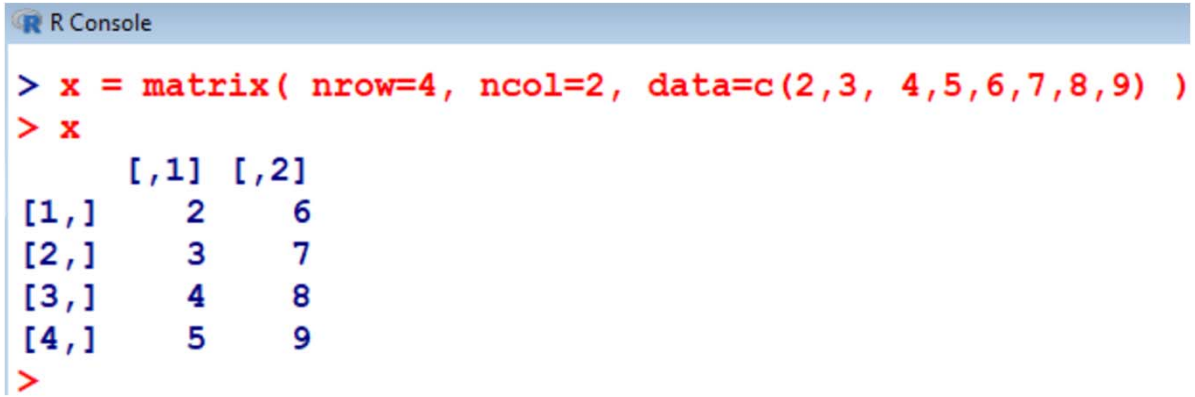
An element in the *i*-th row and *j*-th column is denoted by X_{ij} (book version) or *X*[*i*, *j*] ("program version"), *i* = 1,2,...,*n*, *j* = 1,2,...,*p*.

We consider only numerical matrices, whose elements are generally real numbers.

Matrix

In R, a 4 × 2-matrix *X* can be created with a following command:

```
> x = matrix( nrow=4, ncol=2, data=c(2,3,
4,5,6,7,8,9) )

> x
     [,1] [,2]
[1,]    2    6
[2,]    3    7
[3,]    4    8
[4,]    5    9
```



Matrix

We see:

- **The parameter `nrow` defines the number of rows of a matrix.**

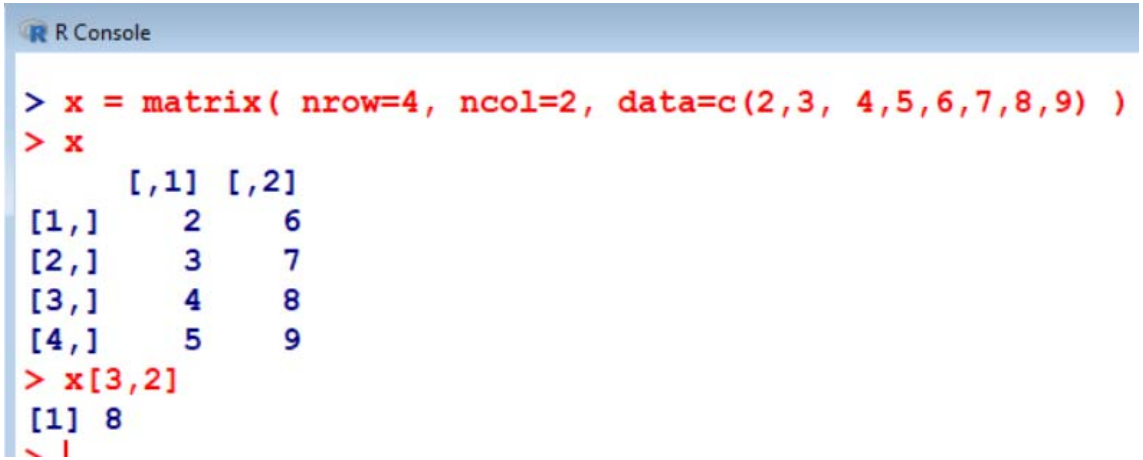
- **The parameter `ncol` defines the number of columns of a matrix.**

- **The parameter `data` assigns specified values to the matrix elements.**

- **The values from the parameters are written column-wise in matrix.**

Matrix

```
> x
     [,1] [,2]
[1,]    2    6
[2,]    3    7
[3,]    4    8
[4,]    5    9
```



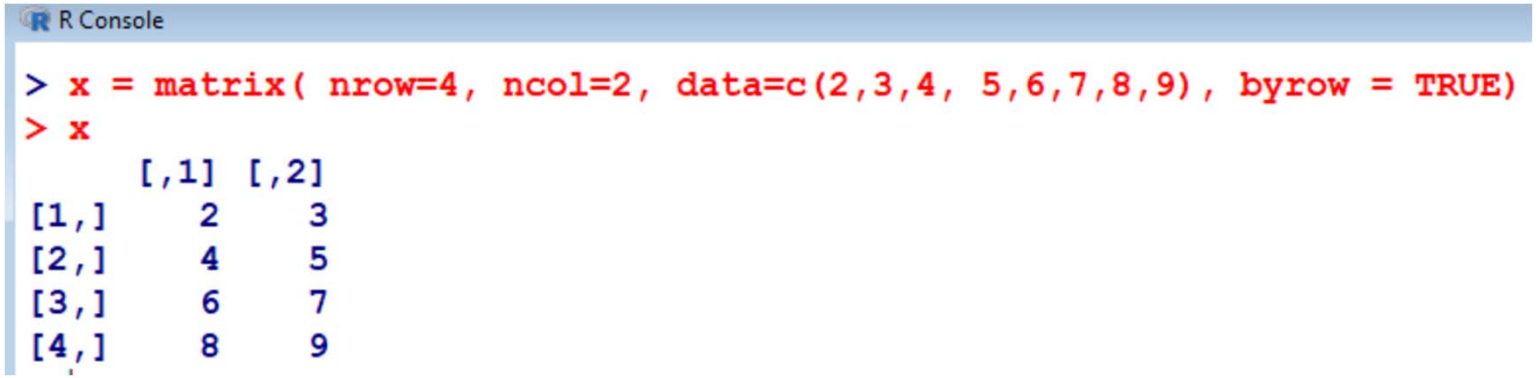
One can access a single element of a matrix with `x[i,j]`:

```
> x[3,2]
[1] 8
```

Matrix

In case, the data has to be entered row wise, then a 4 × 2-matrix *X* can be created with

```
> x = matrix( nrow=4, ncol=2, data=c(2,3,4,
5,6,7,8,9), byrow = TRUE)
> x
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
[4,]    8    9
```

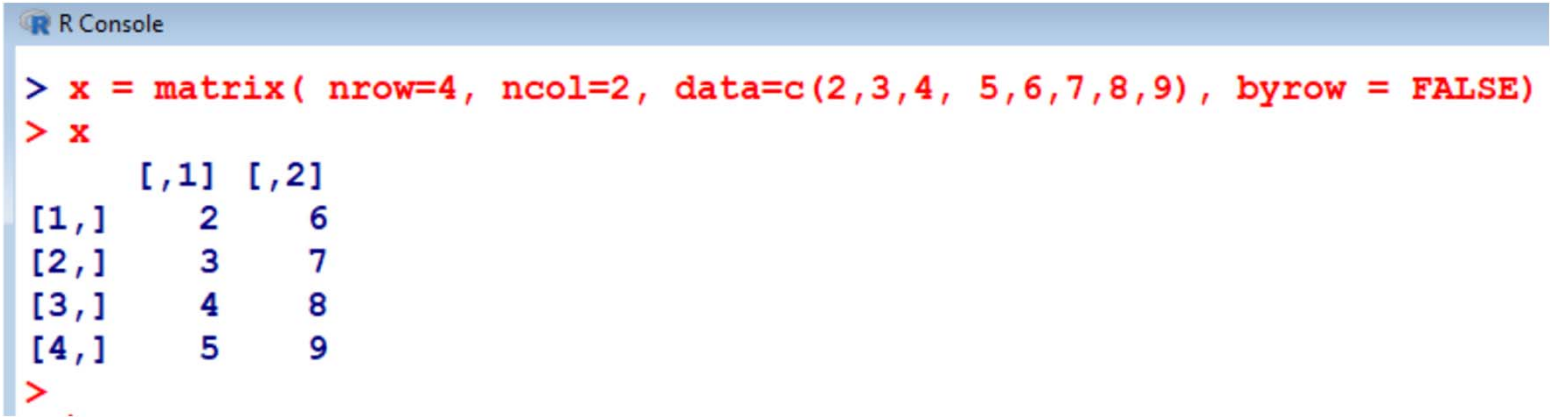


Matrix

In case, the data has to be entered column wise, then a 4 × 2-matrix

***X* can be created with**

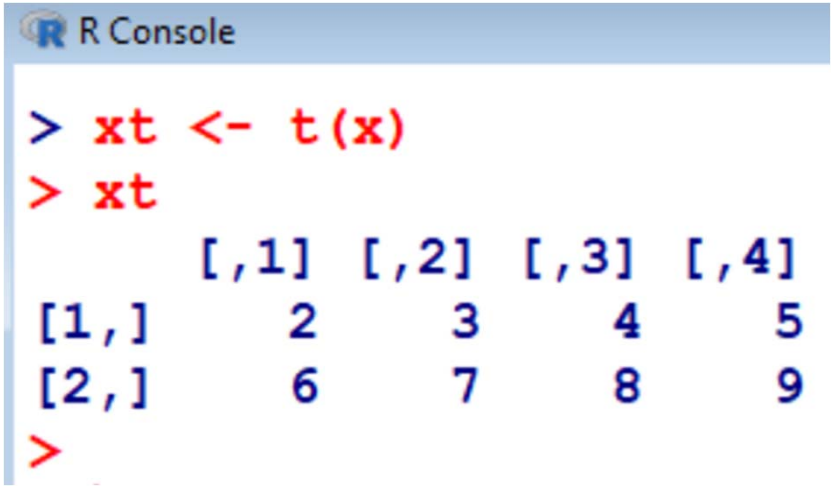
```
> x = matrix( nrow=4, ncol=2, data=c(2,3,4,
5,6,7,8,9), byrow = FALSE)
> x
     [,1] [,2]
[1,]    2    6
[2,]    3    7
[3,]    4    8
[4,]    5    9
```



Matrix: Transpose of a matrix *X*: *X'*

Consider the matrix

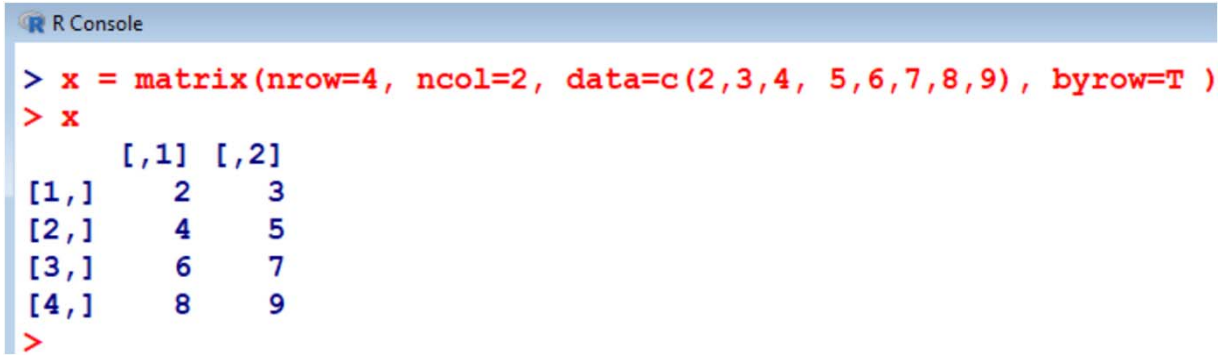
```
> x = matrix( nrow=4, ncol=2, data=c(2,3,4,
5,6,7,8,9), byrow = FALSE)

> x
     [,1] [,2]
[1,]    2    6
[2,]    3    7
[3,]    4    8
[4,]    5    9

> xt <- t(x)
> xt
     [,1] [,2] [,3] [,4]
[1,]    2    3    4    5
[2,]    6    7    8    9
```



Matrix

Multiplication of a matrix with a constant

```
> x = matrix(nrow=4, ncol=2, data=c(2,3,4,
5,6,7,8,9), byrow=T )

> x
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
[4,]    8    9
```



Matrix

Multiplication of a matrix with a constant

```
> x = matrix(nrow=4, ncol=2, data=c(2,3,4,
5,6,7,8,9), byrow=T )
> x
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
[4,]    8    9

> 3*x
     [,1] [,2]
[1,]    6    9
[2,]   12   15
[3,]   18   21
[4,]   24   27
```

Matrix

Multiplication of a matrix with a constant

R Console

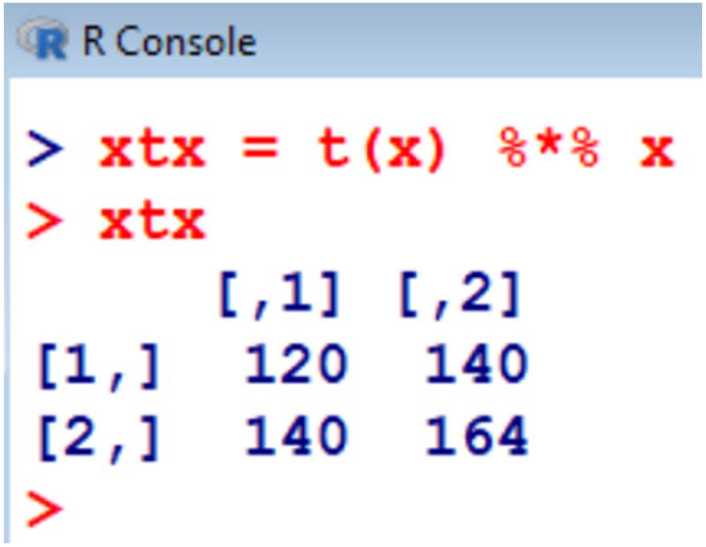
```
> x = matrix(nrow=4, ncol=2, data=c(2,3,4, 5,6,7,8,9), byrow=T )
> x
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
[4,]    8    9
>
> 3*x
     [,1] [,2]
[1,]    6    9
[2,]   12   15
[3,]   18   21
[4,]   24   27
>
```

Matrix

Matrix multiplication: operator %*%

Consider the multiplication of *X'* with *X*

```
> xtx = t(x) %*% x

> xtx
      [,1]   [,2]
[1,]  120    140
[2,]  140    164
```



Matrix

Matrix multiplication: operator %*%

```
> y = matrix(nrow=2, ncol=2, data=c(2,3,4,5),
byrow=T )

> z = matrix(nrow=2, ncol=2, data=c(12,13,
14,15), byrow=T )
```

```
> y
     [,1] [,2]
[1,]    2    3
[2,]    4    5
```

```
> z
     [,1] [,2]
[1,]   12   13
[2,]   14   15
```

Matrix

Matrix multiplication: operator %*%

```
> y%*%z
     [,1] [,2]
[1,]   66   71
[2,]  118  127
```

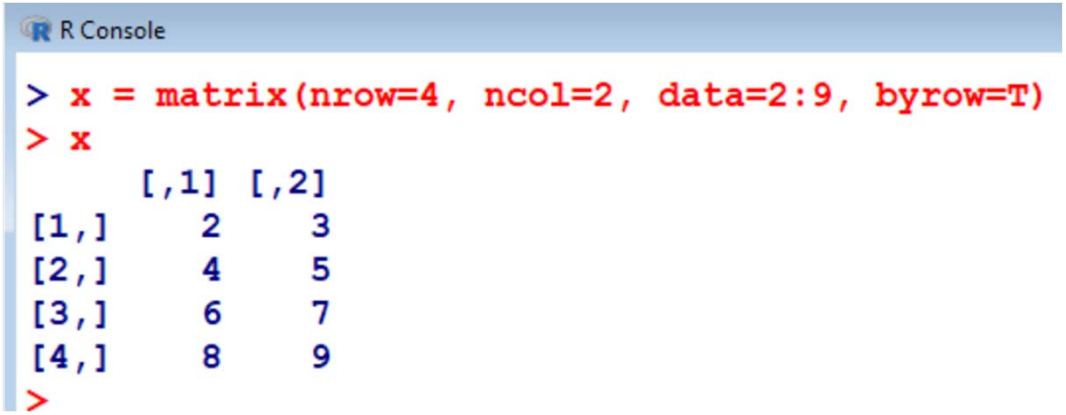
R Console

```
> y = matrix(nrow=2, ncol=2, data=c(2,3,4,5), byrow=T )
> y
     [,1] [,2]
[1,]    2    3
[2,]    4    5
> z = matrix(nrow=2, ncol=2, data=c(12,13, 14,15), byrow=T )
> z
     [,1] [,2]
[1,]   12   13
[2,]   14   15
> y%*%z
     [,1] [,2]
[1,]   66   71
[2,]  118  127
> |
```

Matrix

Addition and subtraction of matrices (of the same dimensions) can be executed with the usual operators + and -

```
> x = matrix(nrow=4, ncol=2, data=2:9, byrow=T)
> x
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
[4,]    8    9
```

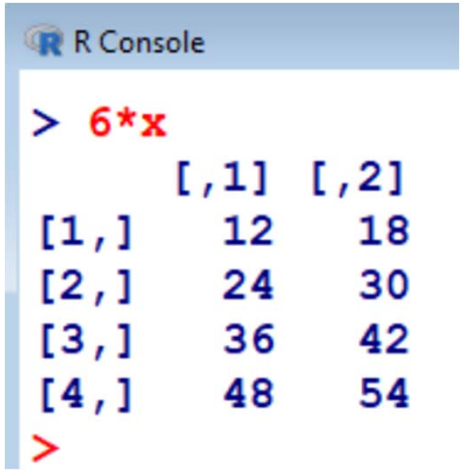


Matrix

Addition and subtraction of matrices (of the same dimensions) can be executed with the usual operators + and -

Create another matrix.

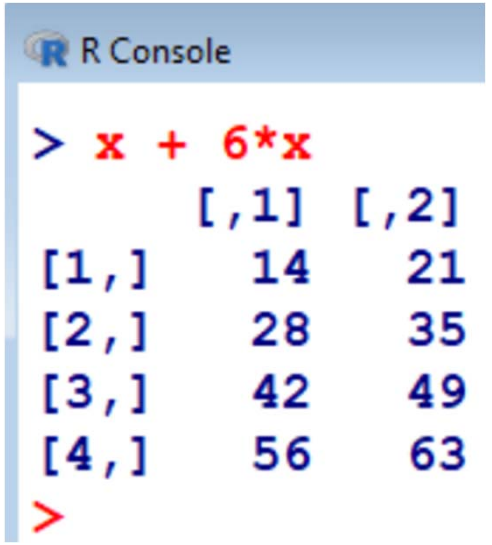
```
> 6*x
     [,1] [,2]
[1,]   12   18
[2,]   24   30
[3,]   36   42
[4,]   48   54
```



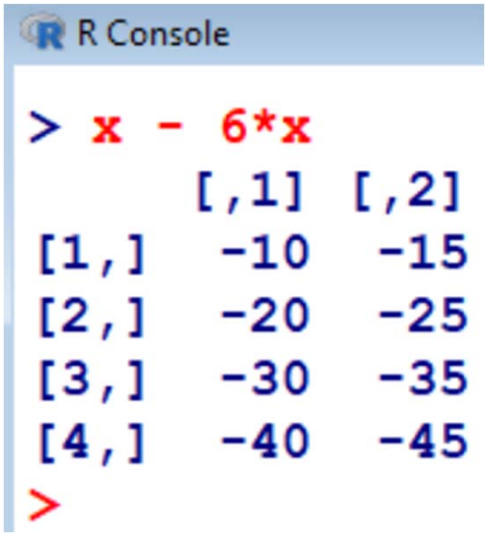
Matrix

Addition and subtraction of matrices (of the same dimensions!) can be executed with the usual operators + and -

```
> x + 6*x
 [,1] [,2]
[1,]   14   21
[2,]   28   35
[3,]   42   49
[4,]   56   63
```




```
> x - 6*x
     [,1] [,2]
[1,]  -10  -15
[2,]  -20  -25
[3,]  -30  -35
[4,]  -40  -45
```



Matrix

Matrix Addition:

```
> y = matrix(nrow=2, ncol=2, data=c(2,3,4,5), byrow=T)

> z = matrix(nrow=2, ncol=2, data=c(12,13, 14,15), byrow=T)
```

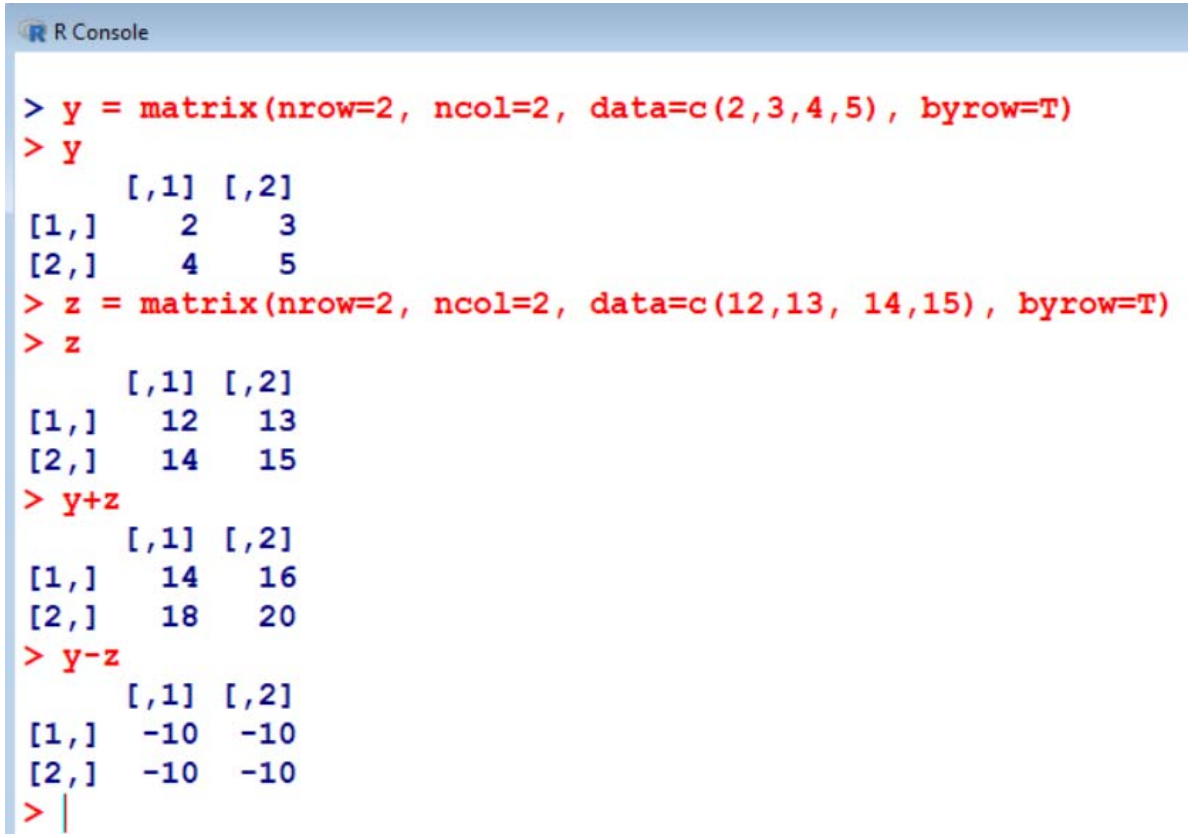
```
> y
     [,1] [,2]
[1,]    2    3
[2,]    4    5
```

```
> z
     [,1] [,2]
[1,]   12   13
[2,]   14   15
```

Matrix

Matrix Addition and Subtraction:

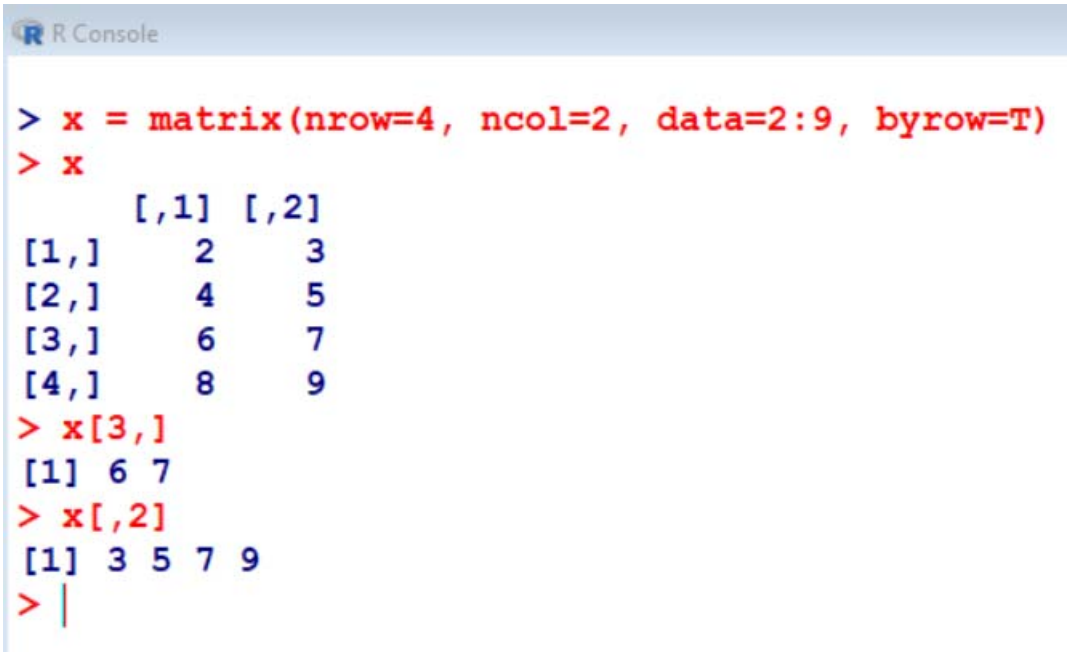
```
> y+z
     [,1] [,2]
[1,]   14   16
[2,]   18   20

> y-z
     [,1] [,2]
[1,]  -10  -10
[2,]  -10  -10
```



Matrix

Access to rows, columns or submatrices:

```
> x[3,]
  [1] 6 7

> x[,2]
  [1] 3 5 7 9
```

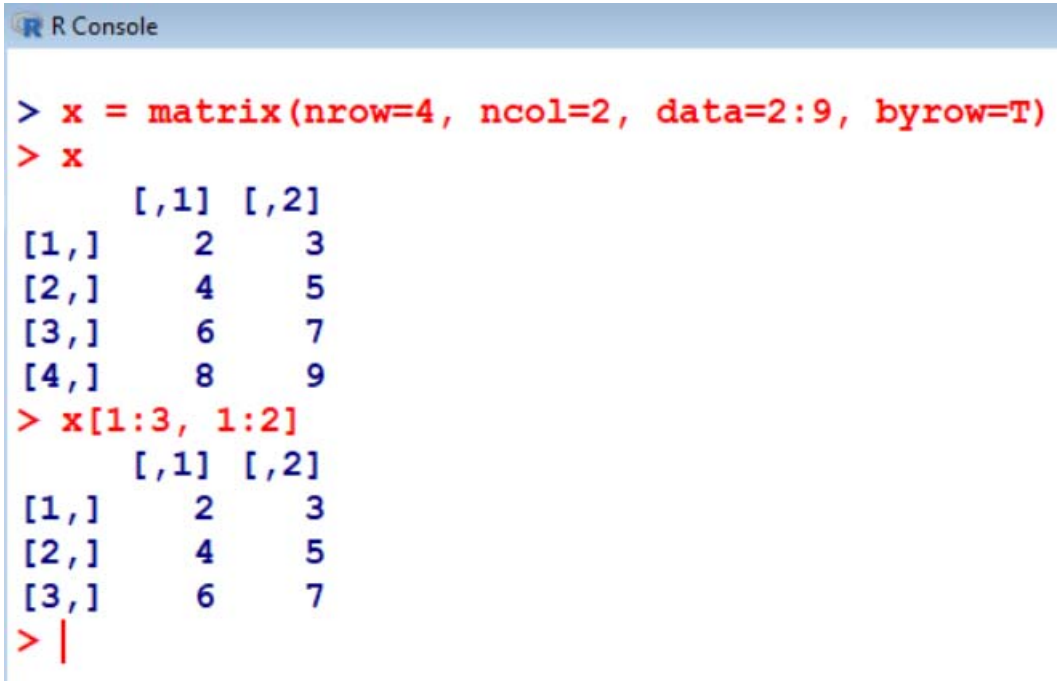


Observe the notations.

Matrix

Access to rows, columns or submatrices:

```
> x[1:3, 1:2]
     [,1] [,2]
[1,]    2    3
[2,]    4    5
[3,]    6    7
```



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 7

Data Handling

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Repeats

Command `rep` is used to replicates the values in a vector.

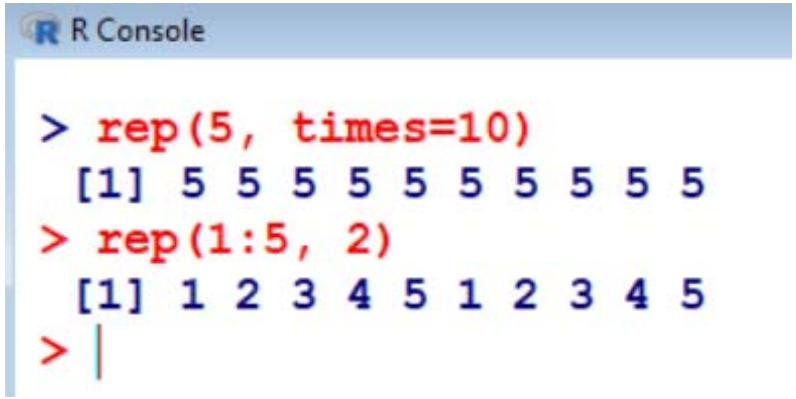
Syntax `rep(x)` replicates the values in a vector `x`.

```
rep(x, times=n) # Repeat x as a whole n times

rep(x, each=n) # Repeat each cell n times
```

Repeats

Repeat an object *n*–times:

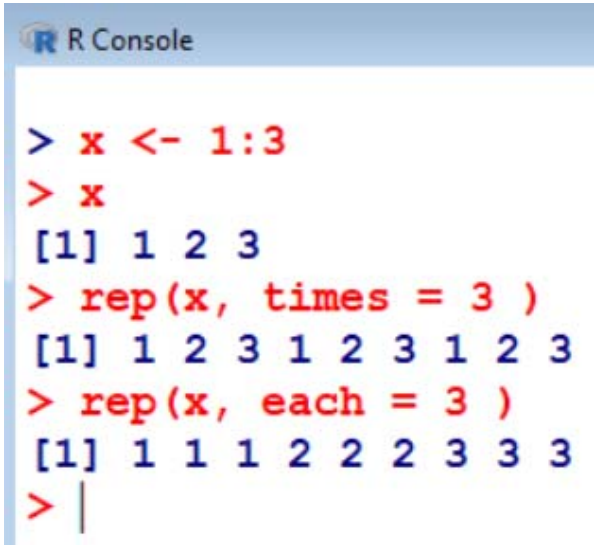
```
> rep(5, times=10)
 [1] 5 5 5 5 5 5 5 5 5 5

> rep(1:5, 2)
 [1] 1 2 3 4 5 1 2 3 4 5
```



Default is `times`.

Repeats

```
> x <- 1:3
> x
[1] 1 2 3

> rep(x, times = 3 )
[1] 1 2 3 1 2 3 1 2 3

> rep(x, each = 3 )
[1] 1 1 1 2 2 2 3 3 3
```



Logical Operators and Comparisons

The following table shows the operations and functions for logical comparisons (TRUE or FALSE).

TRUE and FALSE are reserved words denoting logical constants.

Operator	Executions
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal
==	Exactly equal to
!=	Not equal to
!	Negation (not)

Logical Operators and Comparisons

TRUE and FALSE are reserved words denoting logical constants

Operator	Executions
`xor()`	either... or (exclusive)
`isTRUE(x)`	test if `x` is TRUE
`TRUE`	true
`FALSE`	false

Examples:

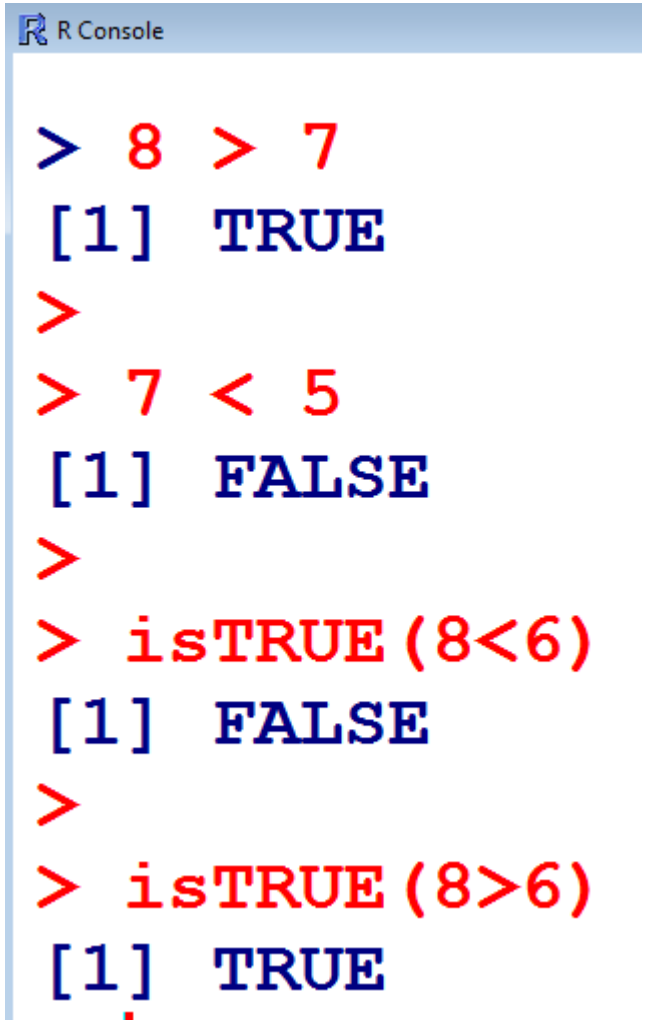
```
> 8 > 7
[1] TRUE

> 7 < 5
[1] FALSE
```

Is 8 less than 6?

```
> isTRUE(8<6)
[1] FALSE
```

Is 8 greater than 6?

```
> isTRUE(8>6)
[1] TRUE
```



Examples:

```
> x <- 5

> (x < 10) & (x > 2)    # & means AND
[1] TRUE
```

Examples:

```
> x = 10
> y = 20
```

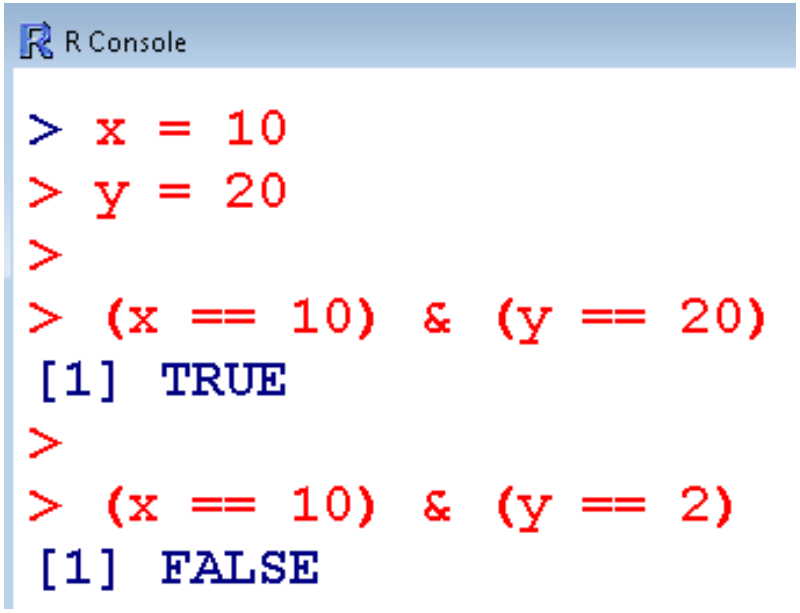
Is x equal to 10 and is y equal to 20?

```
> (x == 10) & (y == 20)
 [1] TRUE
```

== means exactly equal to

Is x equal to 10 and is y equal to 2?

```
> (x == 10) & (y == 2)
 [1] FALSE
```



Examples:

```
> x = 10
> y = 20
```

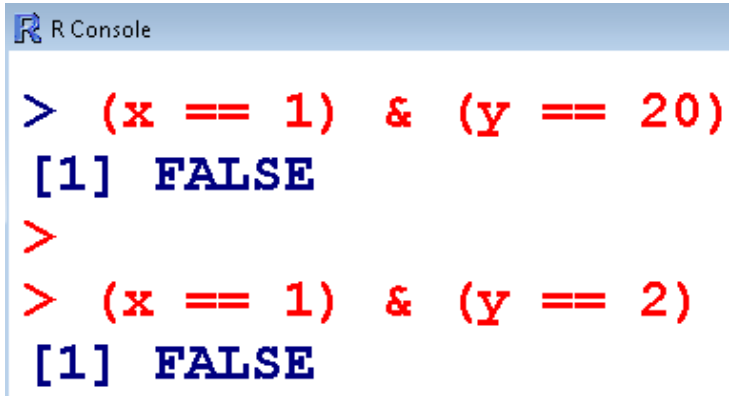
Is x equal to 1 and is y equal to 20?

```
> (x == 1) & (y == 20)
[1] FALSE
```

== means exactly equal to

Is x equal to 1 and is y equal to 2?

```
> (x == 1) & (y == 2)
[1] FALSE
```



Examples:

Find the mean of those observations for which x is less than 5 where x=1,2,…,10.

First find are there any values in dataset for which x is less than 5

```
> x = seq(1:10)
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> x < 5
 [1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE
FALSE FALSE FALSE
```

Find which are those values

```
> x[(x < 5)]
[1] 1 2 3 4
```

Find mean of such values

```
> mean(x[(x < 5)])
[1] 2.5
```

Data Frames

The commands `c` and `matrix` functions combine data.

Another option is the data frame.

In a data frame, we can combine variables of equal length, with each row in the data frame containing observations on the same unit.

Advantage is that one can make changes to the data without affecting the original data.

Data Frames

One can also combine numerical variables, character strings as well as factors in data frame.

Data frames are special types of objects in R designed for data sets.

The data frame format is similar to a spreadsheet, where columns contain variables and observations are contained in rows.

Data Frames

Data frames contain complete data sets that are mostly created with other programs (spreadsheet-files, software SPSS-files, Excel-files etc.).

Variables in a data frame may be numeric (numbers) or categorical (characters or factors).

Data Frames: Creation

❑Creating Data Frames

Use the `data.frame` function to create a data frame by adding column vectors to the data frame.

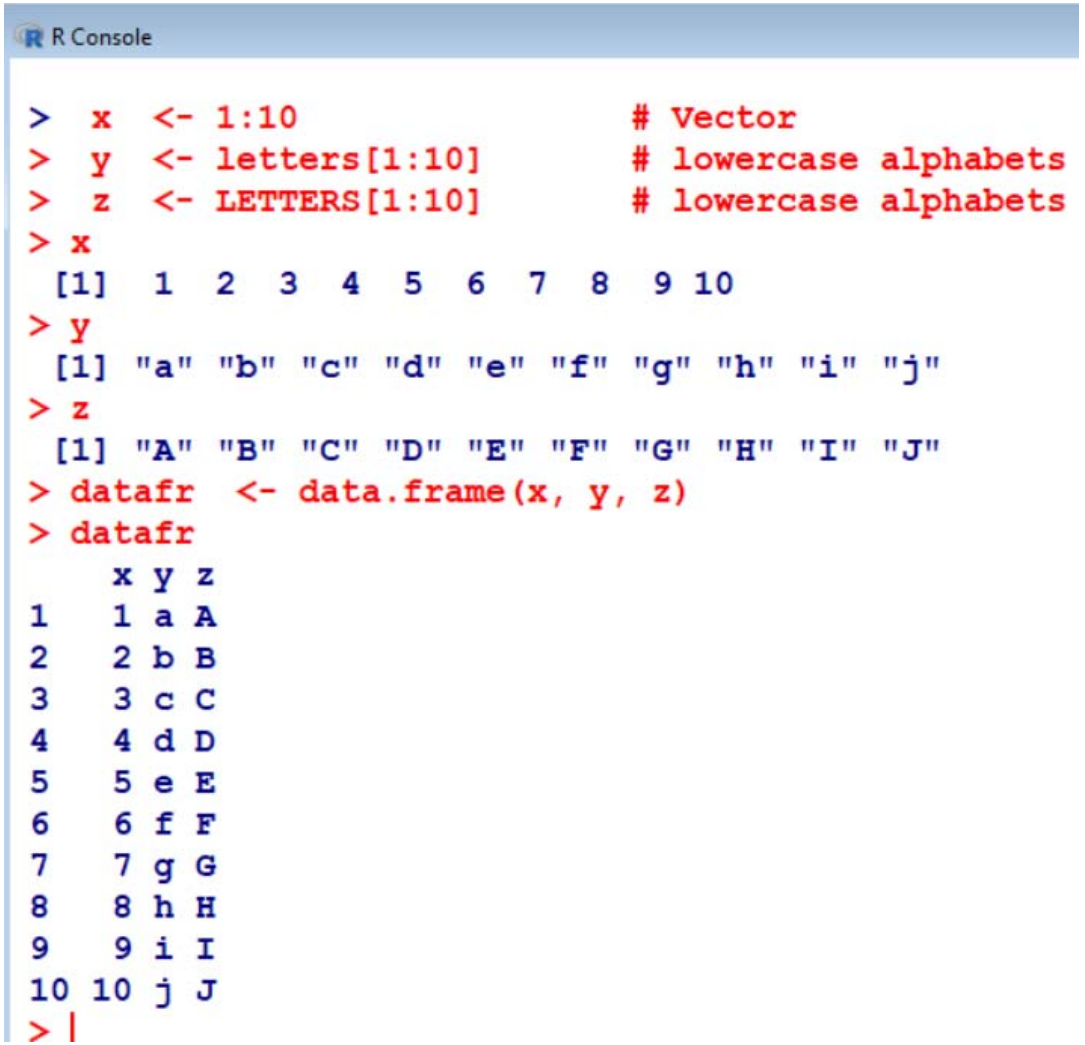
Example:

```
> x  <- 1:10              # Vector
> y  <- letters[1:10]     # lowercase alphabets
> z  <- LETTERS[1:10]     # uppercase alphabets
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> y
 [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
> z
 [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J"
```

Data Frames: Creation

```
> datafr  <- data.frame(x, y, z)
```

```
> datafr
   x y z
1  1 a A
2  2 b B
3  3 c C
4  4 d D
5  5 e E
6  6 f F
7  7 g G
8  8 h H
9  9 i I
10 10 j J
```



Data Frames: Calling a variable

```
> datafr  <- data.frame(x, y, z)

> V1 <- datafr$x
> V1
 [1]  1  2  3  4  5  6  7  8  9 10

> V2 <- datafr$y
> V2
 [1] a b c d e f g h i j
Levels: a b c d e f g h i j

> V3 <- datafr$z
> V3
 [1] A B C D E F G H I J
Levels: A B C D E F G H I J
```

Data Frames: Calling a variable

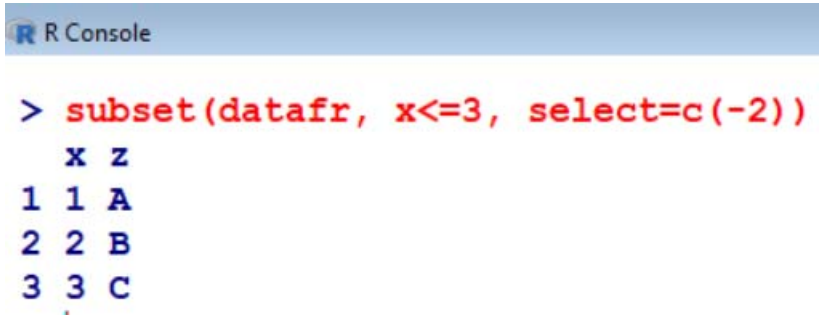
R Console

```
> datafr  <- data.frame(x, y, z)
> datafr
    x y z
1   1 a A
2   2 b B
3   3 c C
4   4 d D
5   5 e E
6   6 f F
7   7 g G
8   8 h H
9   9 i I
10 10 j J
> V1 <- datafr$x
> V1
 [1]  1  2  3  4  5  6  7  8  9 10
> V2 <- datafr$y
> V2
 [1] a b c d e f g h i j
Levels: a b c d e f g h i j
> V3 <- datafr$z
> V3
 [1] A B C D E F G H I J
Levels: A B C D E F G H I J
> |
```

Data Frames: Subset selection

❑ **Uninteresting columns can be eliminated.**

```
> subset(datafr, x<=3, select=c(-2))
  x z
1 1 A
2 2 B
3 3 C
```

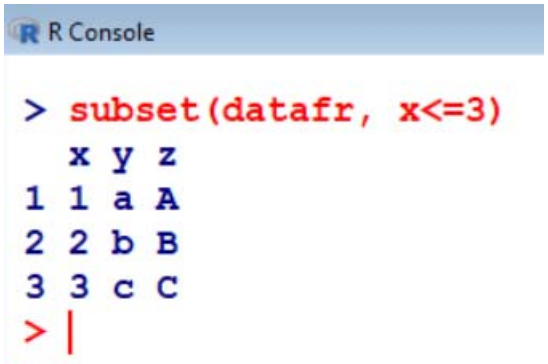


The second column (a,b,…) is not shown.

Data Frames: Subset selection

Subsets of a data frame can be obtained with `subset()` or with the second equivalent command:

```
> subset(datafr, x<=3)
  x y z
1 1 a A
2 2 b B
3 3 c C
```



(# `<=` means logical less than or equal sign)

Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Introduction to R Software

:::

Lecture 8

Graphics and Plots

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Bivariate plots:

Provide first hand visual information about the nature and degree of relationship between two variables.

Relationship can be linear or nonlinear.

We discuss several types of plots through examples.

Bivariate plots: Scatter plot

Plot command:

`x, y:` **Two data vectors**

```
plot(x, y)
```

```
plot(x, y, type)
```

`type`	
"`p`" for points	"`l`" for lines
"`b`" for both	"`c`" for the lines part alone of "`b`"
"`o`" for both 'overplotted'	"`s`" for stair steps.
"`h`" for 'histogram' like (or 'high-density') vertical lines	

Bivariate plots: Scatter plot

Plot command:

`x, y:` **Two data vectors**

```
plot(x, y)
```

```
plot(x, y, type)
```

Get more details from help: `help("type")`

Other options:

`main`	**an overall title for the plot.**
`suba`	**sub title for the plot.**
`xlaba`	**title for the x axis.**
`ylaba`	**title for the y axis.**
`aspthe`	***y/x* aspect ratio.**

Bivariate plots: Example

Number of marks obtained by students depend upon the number of hours of study.

Data on marks out of 500 maximum marks and number of hours per week for 20 students are collected as follows:

Marks out of 500 maximum marks

```
marks <- c(337,316,334,327,340,360, 374,330,352,
353,370,380,384,398,413,428,430,438,439,450)
```

Number of hours per week

```
hours <- c(23,25,25,26,27,28,30,26,29,32,33,34,
35,38,39,42,43,44, 45,45.5)
```

Bivariate plots: Scatter plot

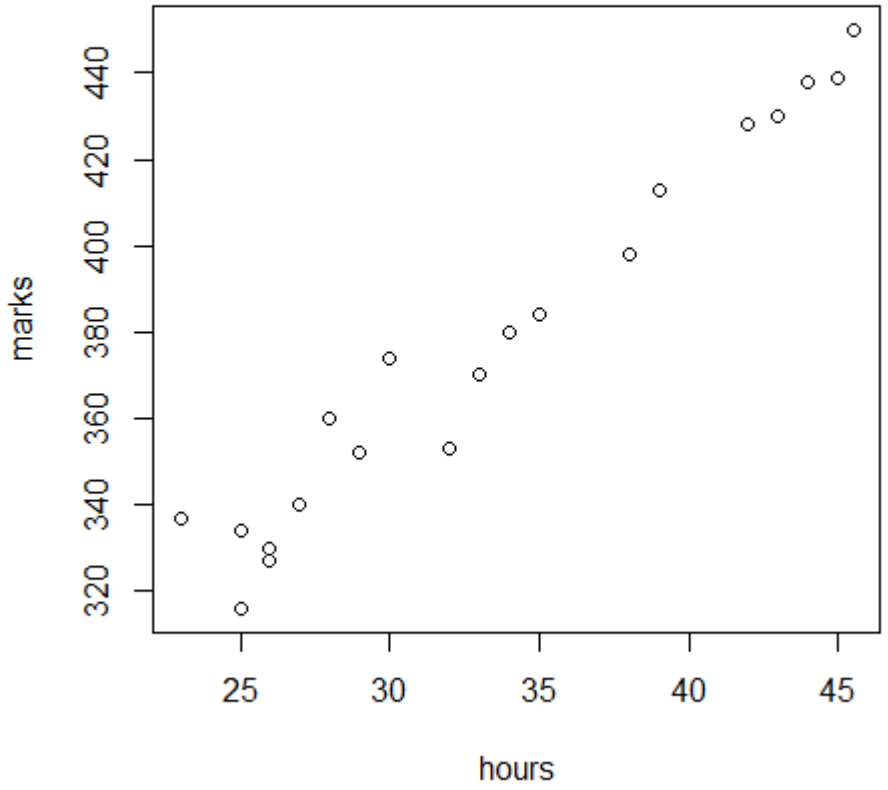
`plot` command:

`x, y:` Two data vectors

Various type of plots are possible to draw.

```
plot(x, y)
```

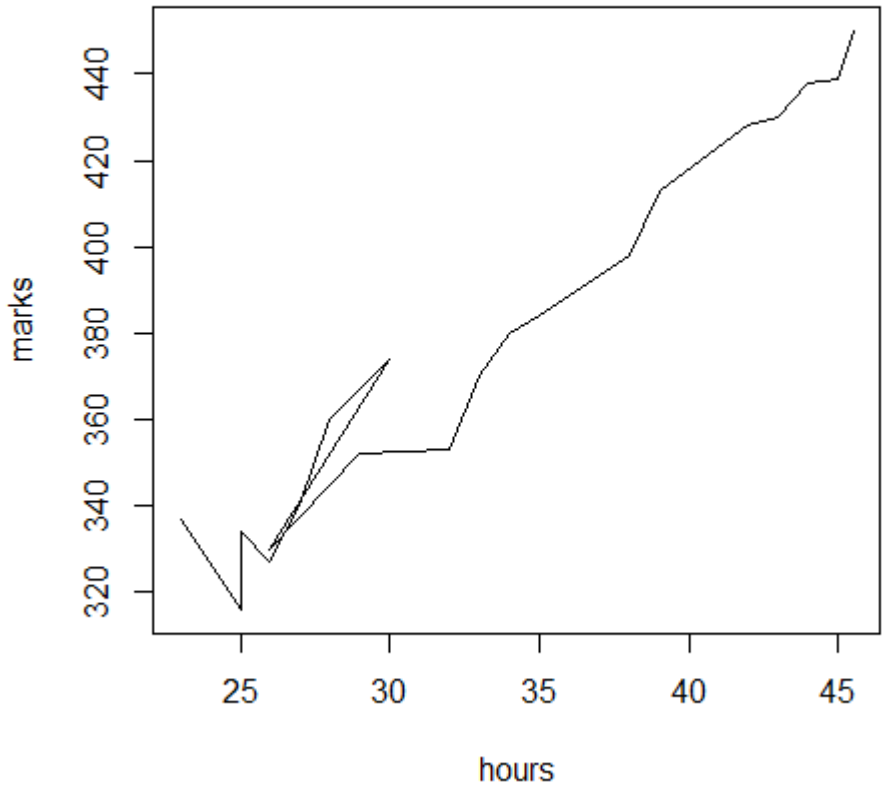
```
plot(hours, marks)
```



Bivariate plots: Scatter plot

```
plot(hours, marks, "l")
```

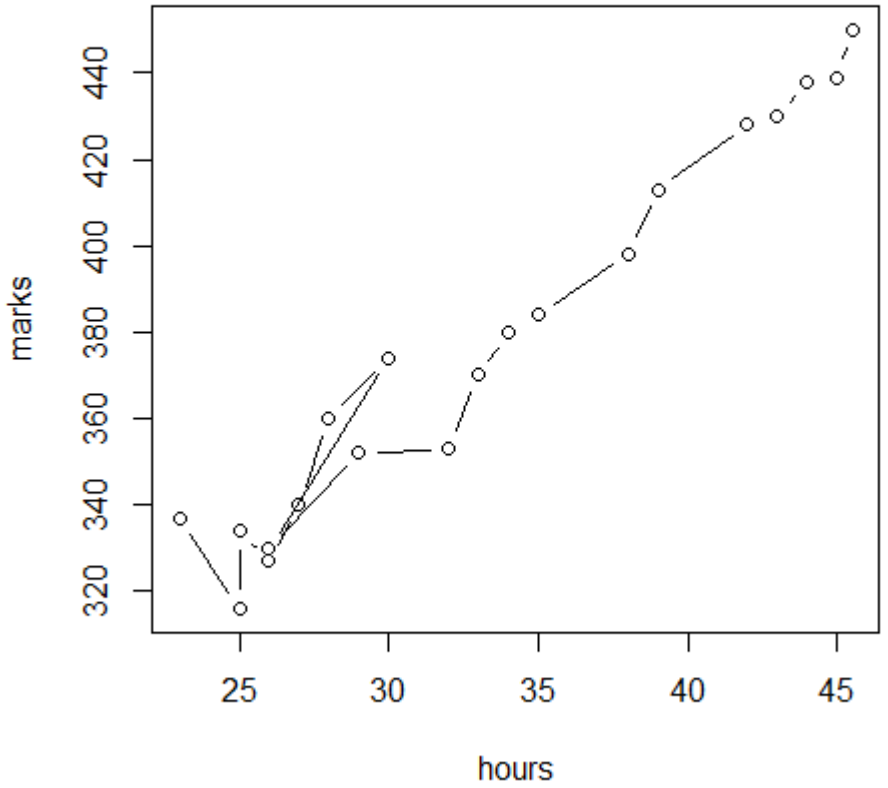
“`l`" for lines,



Bivariate plots: Scatter plot

```
plot(hours, marks, "b")
```

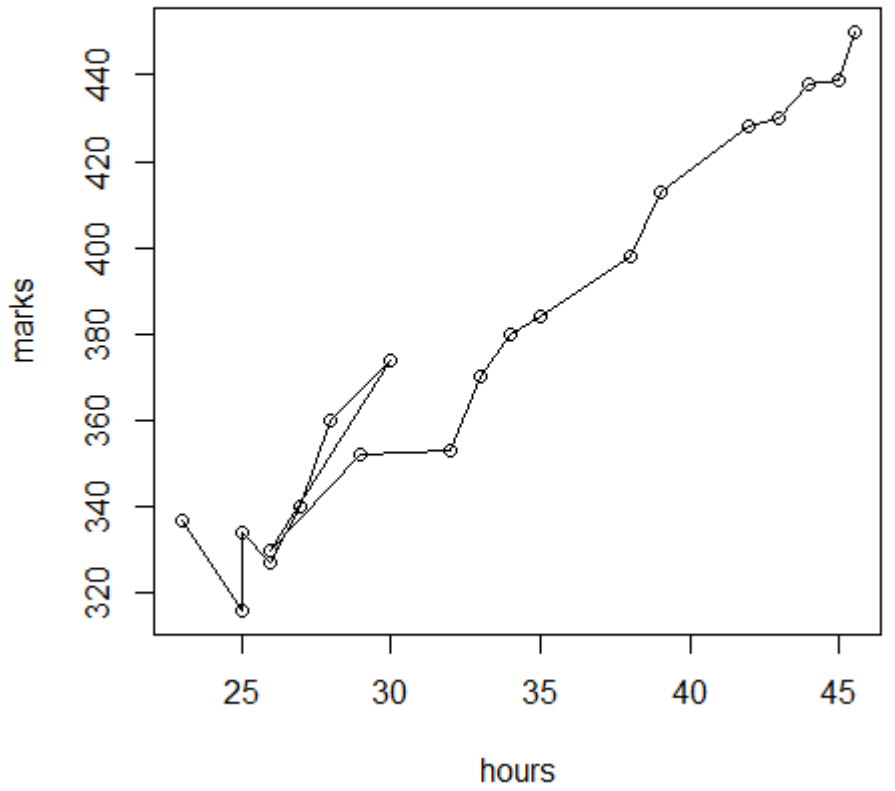
"b" for both – line and point



Bivariate plots: Scatter plot

```
plot(hours, marks, "o")
```

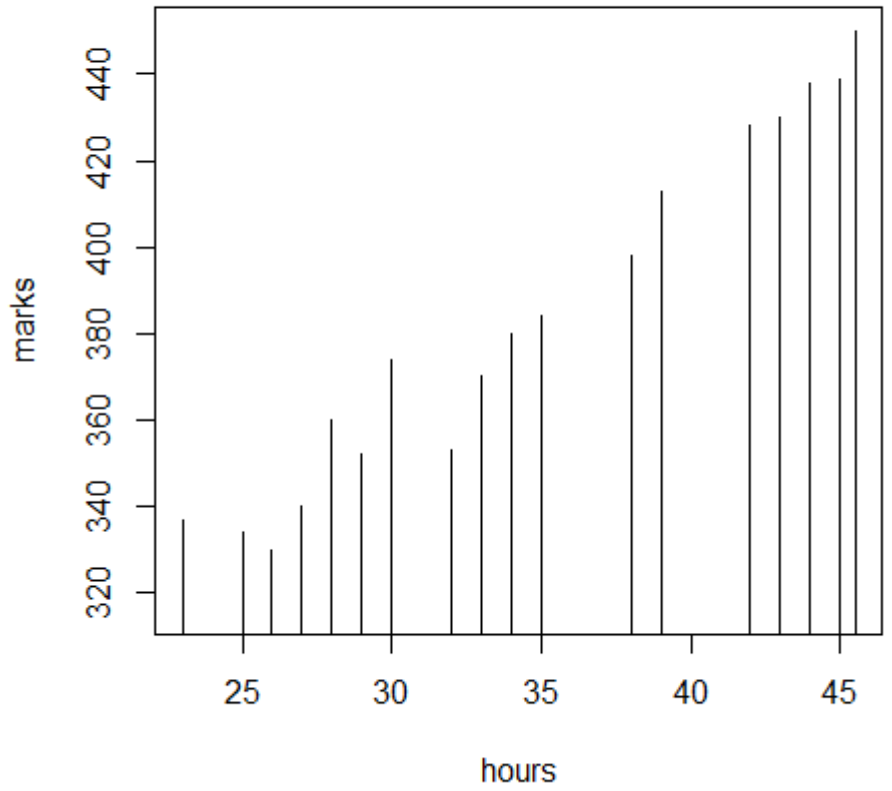
“o" for both ‘overplotted’



Bivariate plots: Scatter plot

```
plot(hours, marks, "h")
```

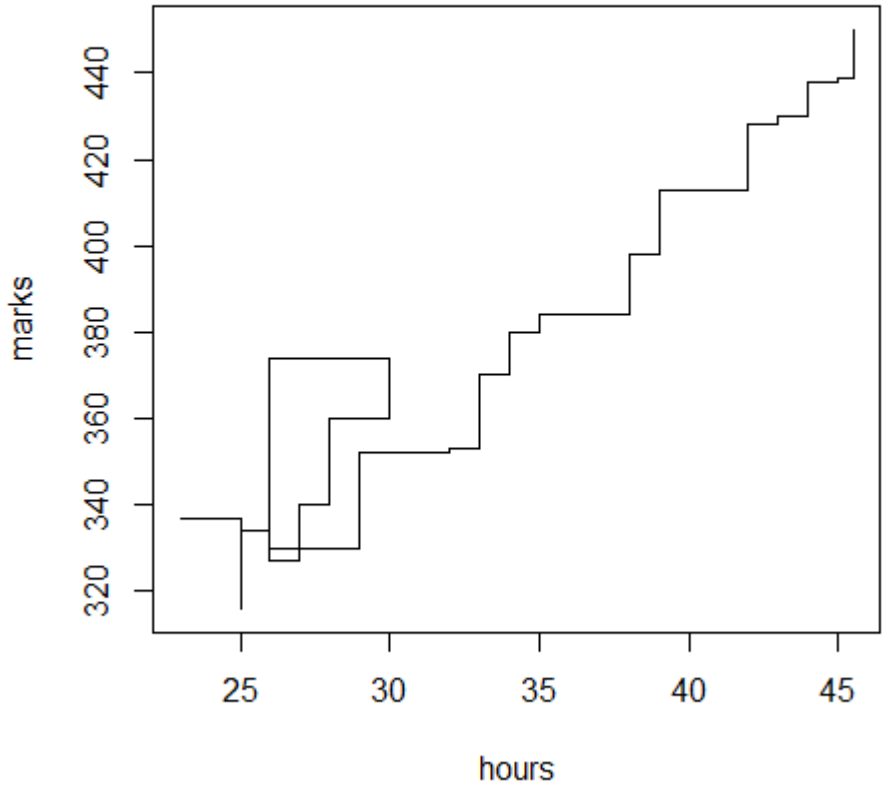
"h" for 'histogram' like (or 'high-density') vertical lines



Bivariate plots: Scatter plot

```
plot(hours, marks, "s")
```

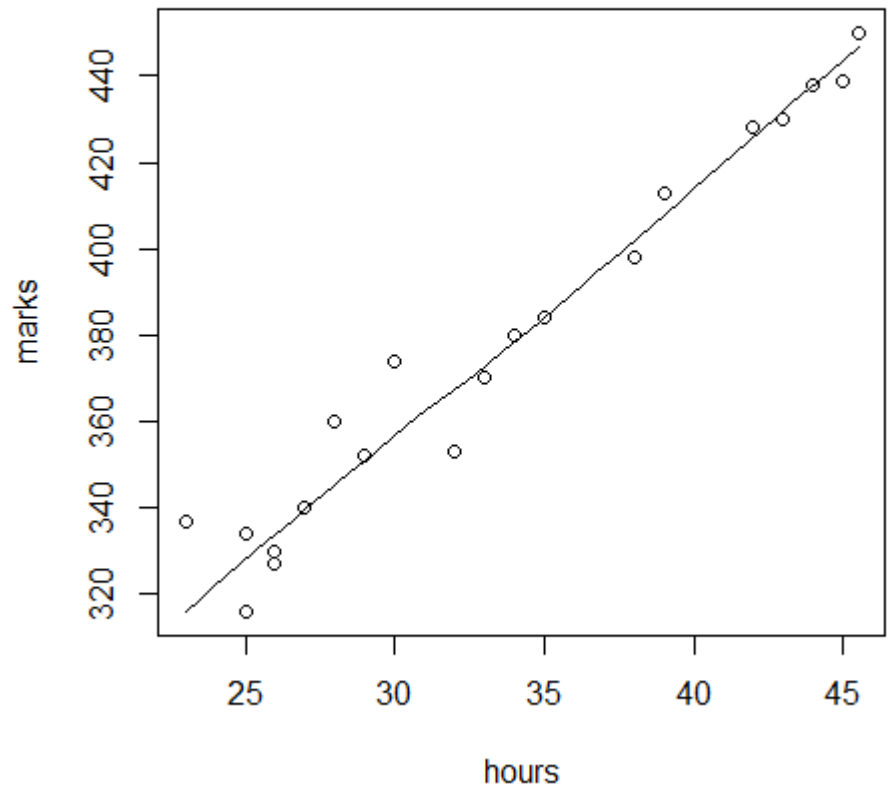
"**s**" for stair **s**teps.



Smooth Scatter plot

`scatter.smooth(x,y)` provides scatter plot with smooth curve

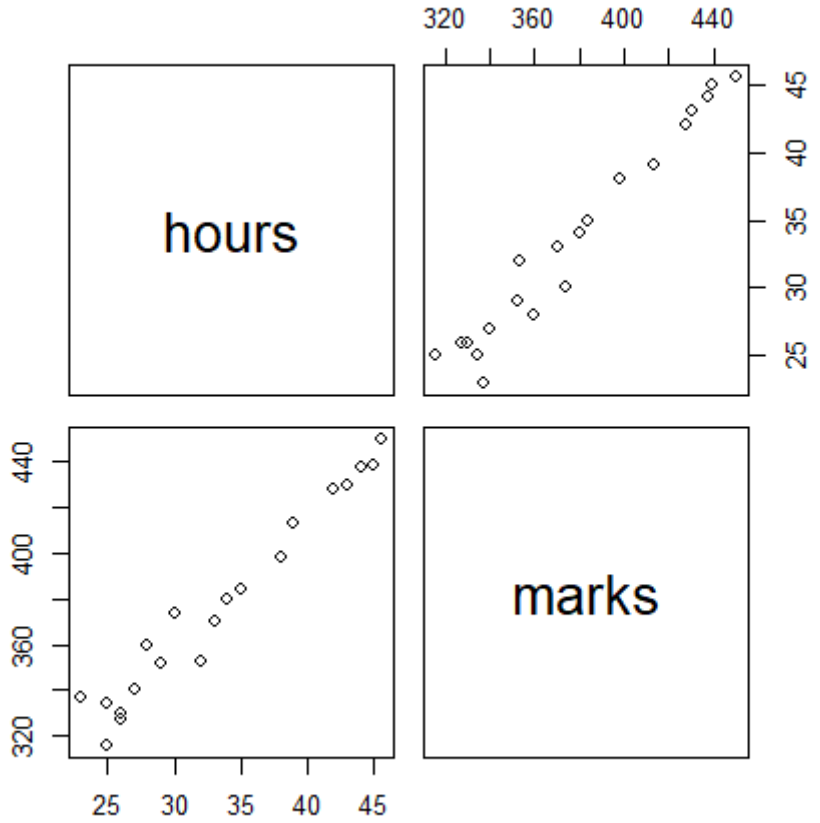
Example: `scatter.smooth(hours, marks)`



Matrix Scatter plot

The command `pairs()` allows the simple creation of a matrix of scatter plots.

```
> pairs(cbind(hours, marks))
```



Correlation plot

`corrplot` is used for the visualization of a correlation matrix.

Usage

Description

A graphical display of a correlation matrix, confidence interval. The details are paid great attention to. It can also visualize a general matrix by setting `is.corr = FALSE`.

Usage

```
corrplot(corr, method = c("circle", "square",
"ellipse", "number", "shade",  "color", "pie"),
type = c("full", "lower", "upper"), col = NULL,
bg = "white", title = "", is.corr = TRUE, …)
```

See Help for more details and options

Correlation plot: Example

Suppose there are 20 observations on 4 variables- y, x_{i1}, x_{i2}, and x_{i3},

i = 1,2,…,20

```
y=c(180,116,118,139,195,152,218,170,179,210,178
,104,145,203,163,216,106,216,191,197)

X1=c(34,12,15,33,31,24,40,31,21,37,29,15,17,38,
17,36,13,39,36,34)

X2=c(3,1,3,1,5,1,5,5,2,3,4,1,1,5,1,3,1,5,5,1)

X3=c(15,13,11,10,17,15,18,13,20,19,16,10,16,16,
19,20,11,18,15,19)
```

Correlation plot

Creating a correlation plot

First install the package `corrplot` and load it.

```
install.packages("corrplot")

library(corrplot)

X123y=data.frame(y,X1,X2,X3) #Data frame creation
```

Creates a correlation matrix

```
X123y_cor = cor(X123y) # Creates correlation
matrix
```

Creates a correlation plot

```
corrplot(X123y_cor, method = "number")
```

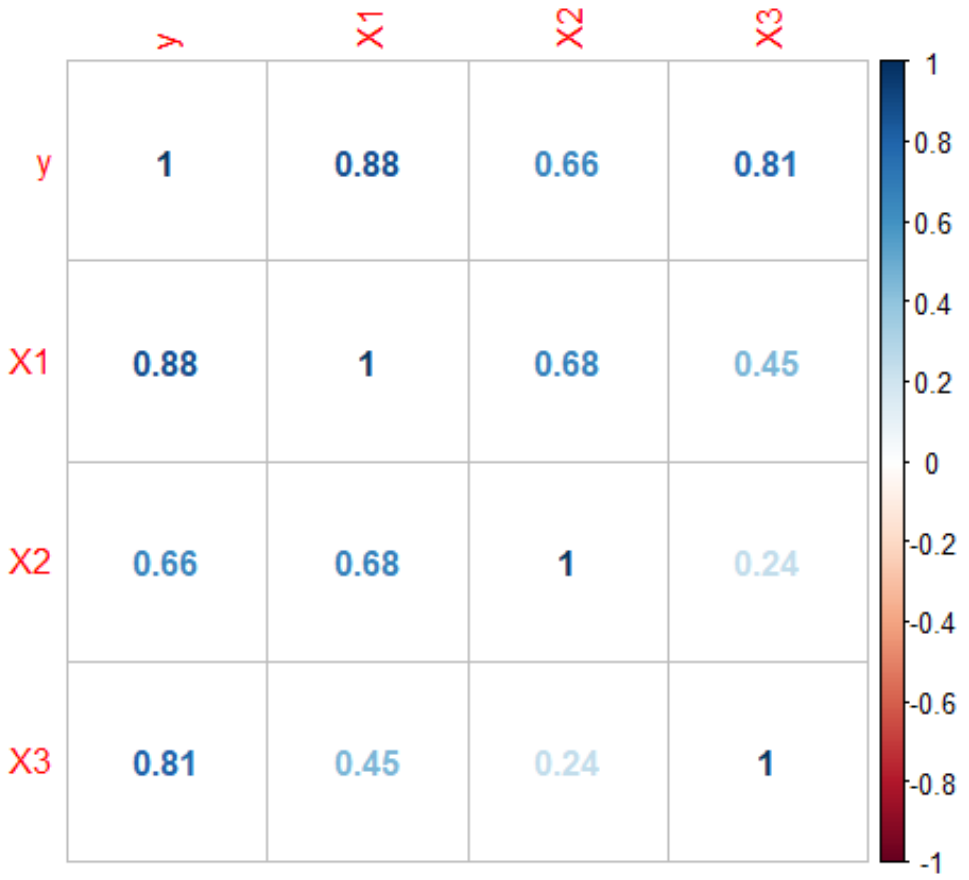
Correlation plot: Example

```
> X123y=data.frame(y,X1,X2,X3) #Data frame creation
> X123y_cor = cor(X123y)


> X123y_cor
           y        X1        X2        X3
y  1.0000000 0.8779040 0.6564152 0.8087657
X1 0.8779040 1.0000000 0.6782342 0.4545290
X2 0.6564152 0.6782342 1.0000000 0.2408078
X3 0.8087657 0.4545290 0.2408078 1.0000000
```

Correlation plot: Example

```
> corrplot(X123y_cor, method = "number")
```

Observe the shades of font of colours in the graphs



Correlation plot: Example

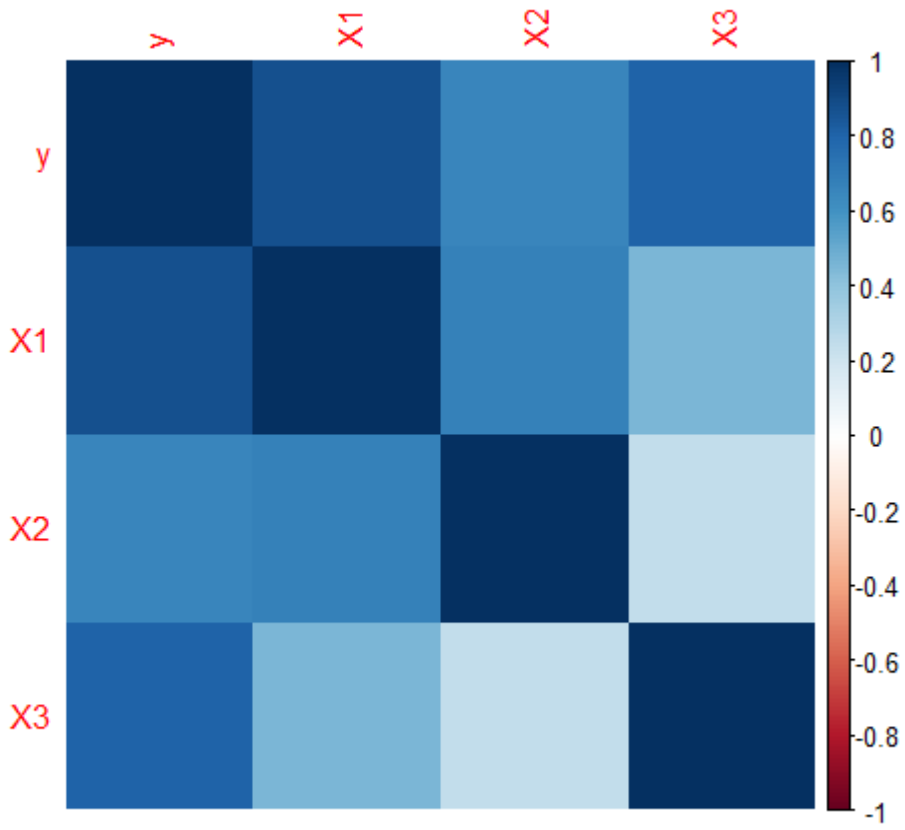
R Console

```
> library(corrplot)
>
> X123y=data.frame(y,X1,X2,X3) #Data frame creation
> X123y_cor = cor(X123y)
> X123y_cor
          y        X1        X2        X3
y  1.0000000 0.8779040 0.6564152 0.8087657
X1 0.8779040 1.0000000 0.6782342 0.4545290
X2 0.6564152 0.6782342 1.0000000 0.2408078
X3 0.8087657 0.4545290 0.2408078 1.0000000
> corrplot(X123y_cor, method = "number")
>
```

Correlation plot: Example

```
> corrplot(X123y_cor, method = "shade")
```

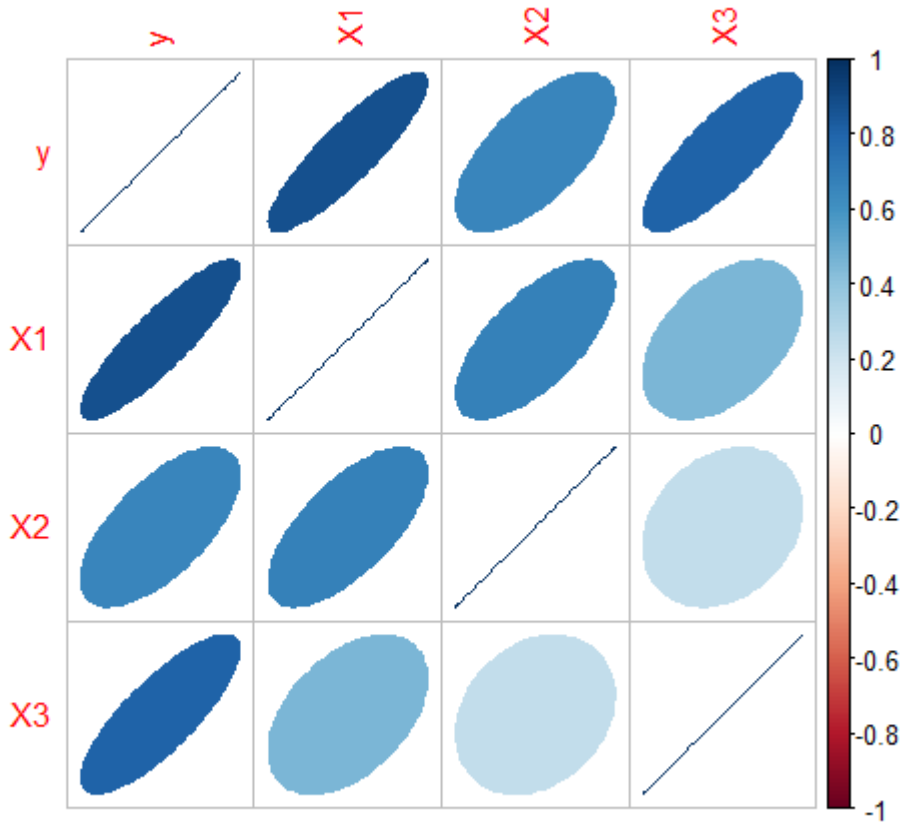
Observe the shades of font of colours in the graphs



Correlation plot: Example

```
> corrplot(X123y_cor, method = "ellipse ")
```

Observe the shades of font of colours and width of ellipse in the graphs



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Sampling Theory with R Software

:::

Lecture 9

Basic Fundamentals

::

Sampling, Sampling Unit, Population and Sample

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Sampling:

Why do you need sampling?

Description of any statistical tool starts with "Let $x_1, x_2,\ldots, x_n$ be a random sample from population…."

Based on this sample, the statistical analysis is conducted.

As a matter of fact, statistics has utility only because it can provide statistical inferences for the entire population using the sample data.

Sampling:

How to obtain these "$x_1, x_2,\ldots, x_n$" ?

If these "$x_1, x_2,\ldots, x_n$" are good, we get good inferences.

If these "$x_1, x_2,\ldots, x_n$" are bad, we get bad inferences.

Entire success of statistical tools depends upon the outcomes and the outcome depends upon the quality of sample used in the analysis.

Sampling:

Sampling theory helps.

It provides methodologies for choosing "$x_1, x_2,\ldots, x_n$" .

The methodologies ensures that the "$x_1, x_2,\ldots, x_n$" are "good" and as per the requirements of the statistical tools to be used.

Sampling:

Sampling theory provides the tools and techniques for data collection keeping in mind the objectives to be fulfilled and nature of population.

These are two ways of obtaining the information

1. Sample surveys

2. Complete enumeration or census

Sampling:

- **Sample surveys collect information on a fraction of total population whereas**
- **the information on whole population is collected in census.**

Some surveys are conducted regularly like economic surveys, agricultural surveys etc.

Some surveys are need based and are conducted when some need arise, e.g., consumer satisfaction surveys at a newly opened shopping mall to see the satisfaction level with the amenities provided in the mall .

Sampling Unit:

An element or a group of elements on which observations can be taken is called a sampling unit.

The objective of the survey helps in determining the definition of sampling unit.

Definition of sampling unit depends and varies as per the objective of the survey.

Sampling Unit:

Example:

Objective: To determine the total income of all the persons in the household.

Sampling unit: Household.

Example:

Objective: To determine the income of any particular person in the household.

Sampling unit: Income of the particular person in the household.

Sampling Unit:

Example:

Objective: To study the health conditions.

Sampling unit: The person on whom the readings on the blood sugar level, blood pressure and other factors will be obtained. These values will together classify the person as healthy or unhealthy.

Sampling Unit:

Example:

Objective: A fish food increases the weight of the fish or not.

Sampling unit: What is sampling unit?

Weight of fish or weight of aquarium?

Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Sampling Theory with R Software

:::

Lecture 10

Basic Fundamentals

::

Terminologies and Concepts

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Population:

Collection of all the sampling units in a given region at a particular point of time or a particular period is called population.

Example:

Objective: Medical facilities in a hospital are to be surveyed through the patients.

Population: Total number of patients registered in the hospital during the time period of survey.

Population:

Example:

Objective: To study the production of wheat in a district.

Population: All the fields cultivating wheat in that district.

Population:

Population size: Total number of sampling units in the population.

Denoted generally by *N*.

The population size can be finite or infinite (*N* is large).

Census:

Complete count of population is called census.

The observations on all the sampling units in the population are collected in a census.

For example, in India, the census is conducted at every tenth year in which observations on all the persons staying in India is collected.

Sample:

Collection of One or more sampling units selected from the population according to some specified procedure.

A sample consists only of a portion of the population units.

Representative Sample:

All salient features of population are present in the sample.

Every sample has to be a representative sample.

For example, if a population has 30% male and 70% female, then we also expect the sample to have nearly 30% male and 70% females.

Representative Sample:

In another example, if we take out a handful of wheat from a 100 Kg. bag of wheat, we expect the same quality of wheat in hand as inside the bag.

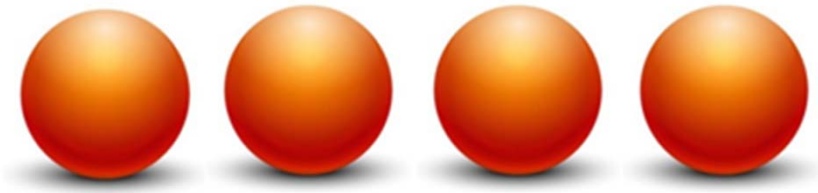
It is expected that a drop of blood will give the same information as all the blood in the body.

Population and Sample:

Population of balls of size 10

Sample 1 of size 3

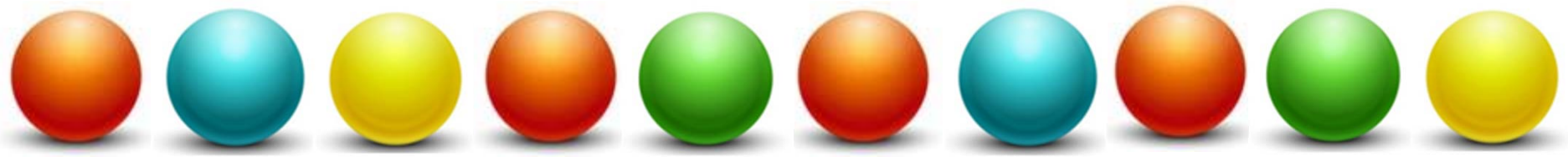
Sample 1 of size 4



Sample 1 of size 5

Population and Sample:

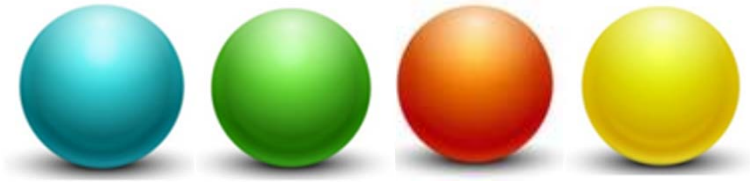
Population of balls of size 10



Sample 1 of size 3
(Green ball is missing)

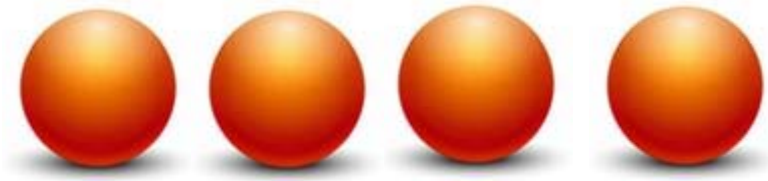
Sample 2 of size 4

(All colour balls are present)



Sample 3 of size 5
(Blue ball is repeated)

Sample 4 of size 4
(Only red balls are selected)

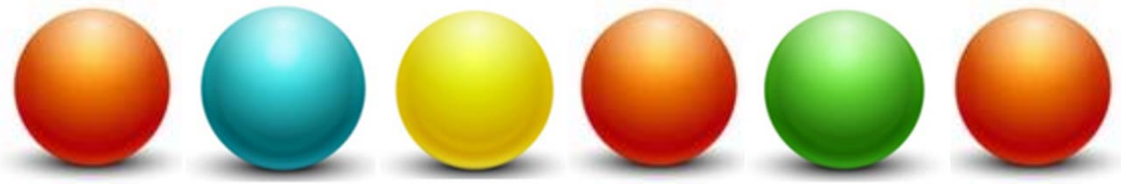


Census and Sample:

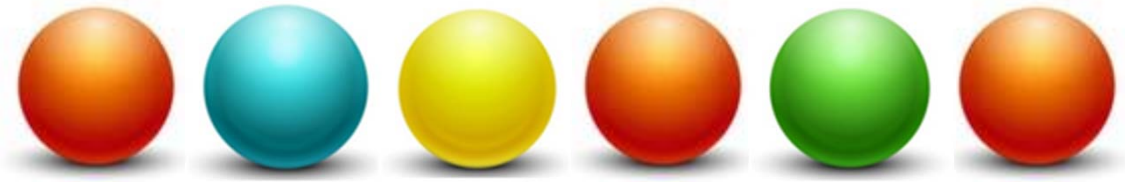
In the context of sample surveys, a collection of units like people, cities, countries etc. is called a finite population.

A census is a 100% sample and it is a complete count of the population.

Population



Sample



This is census.

Sampling Frame:

List of all the units of the population to be surveyed constitutes the sampling frame.

All the sampling units in the sampling frame have identification particulars.

For example, all the students in a particular university listed along with their roll numbers constitutes the sampling frame.

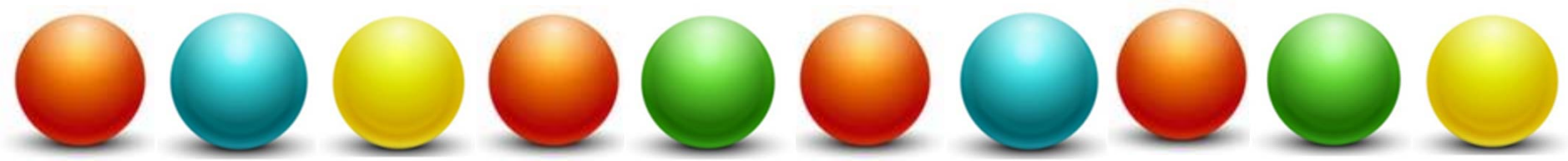
Simple Random Sampling:

Population of balls of size 10

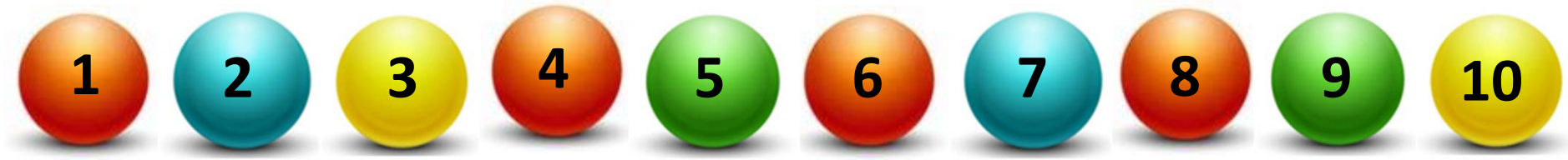
Sampling frame

Simple Random Sampling :

Population of balls of size 10



Sampling frame



Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Sampling Theory with R Software

:::

Lecture 11

Basic Fundamentals

::

Ensuring Representativeness and Type of Surveys

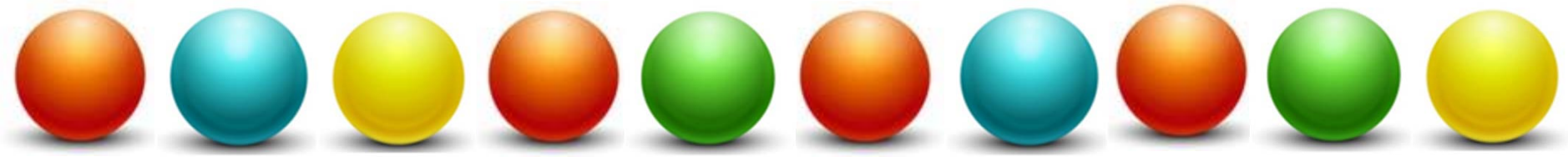
Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Representative Sample:

Population:



Sample 1: Is this representative?
(Green and blues balls are missing)

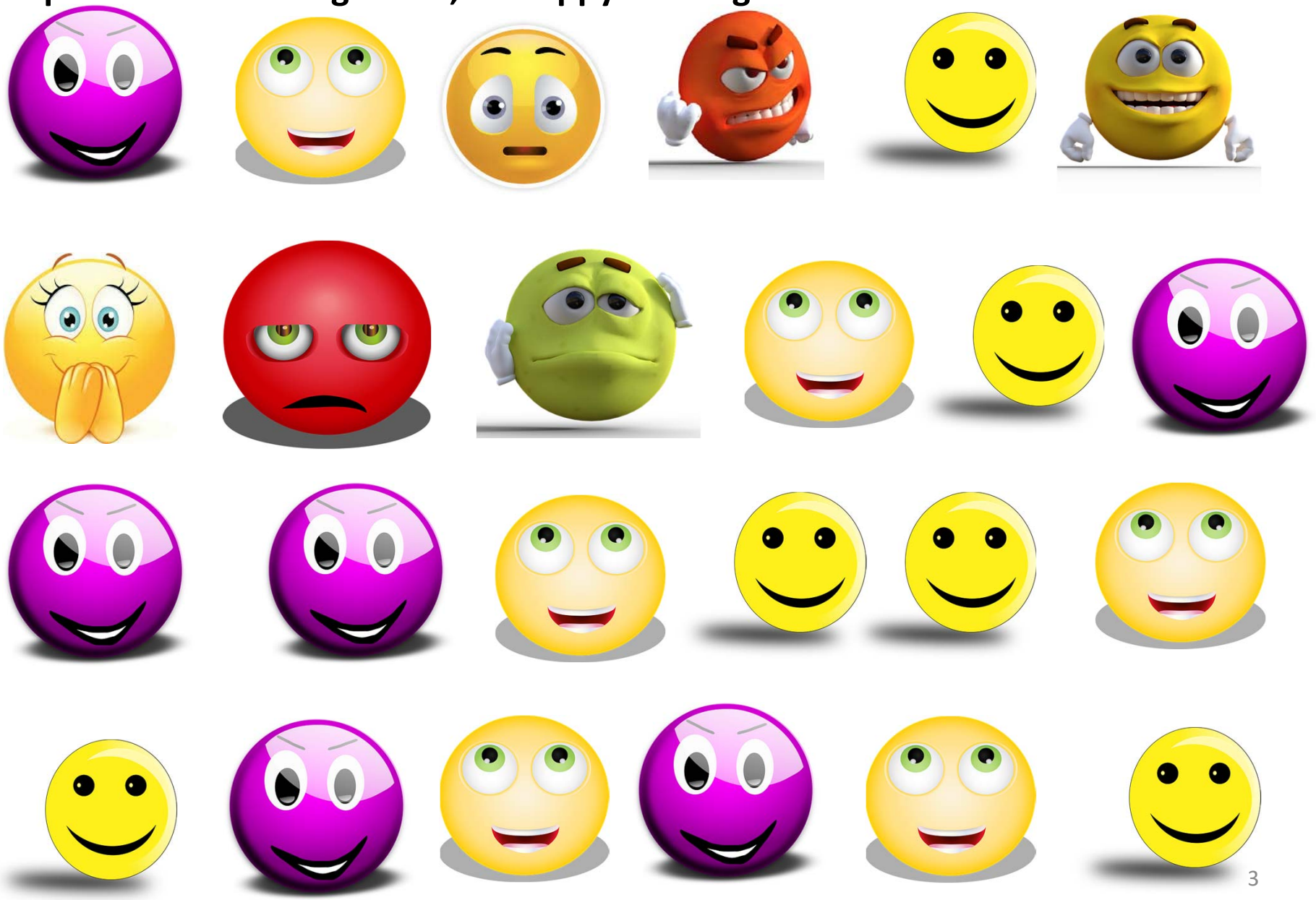
Sample 2: Is this representative?
(Red ball is missing)

Sample 3: Is this representative?
(All balls are proportionally present)

Sample 4: Is this representative?
(Red ball is over represented)

Representative Sample:

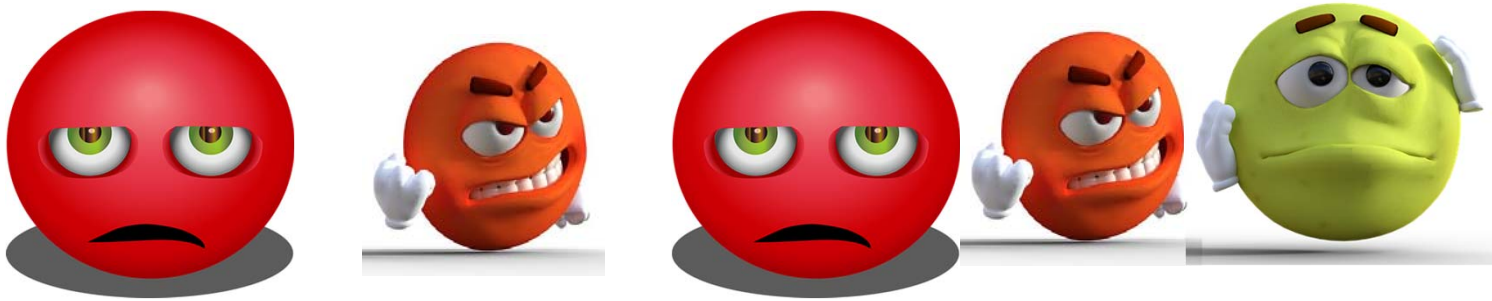
Population: Gives in general, a "Happy“ feeling



Representative Sample:

Is this representative? Appears as if mostly people are not happy.

Is this representative? Appears as if mostly people are Annoyed.



Is this representative? Appears as if mostly people are Happy.

Ensuring representativeness:

There are two possible ways to ensure that the selected sample is representative.

1. **Random sample or Probability sample**
2. **Non-random sample or Purposive sample**
3. **Quota Sample**

Ensuring representativeness:

1. Random Sample or Probability Sample:

The selection of units in the sample from a population is governed by the laws of chance or probability.

The probability of selection of a unit can be equal as well as unequal.

Ensuring representativeness:

2. Non-Random Sample or Purposive Sample:

The selection of units in the sample from population is not governed by the probability laws.

It is the sample based on non-random laws.

Examples:

- **Units are selected on the basis of personal judgment of the surveyor.**
- **The persons volunteering to take some medical test.**
- **The persons volunteering to drink a new type of coffee.**

Ensuring representativeness:

3. Quota Sample:

The survey in this case is continued until a predetermined number of units with the characteristic under study are picked up.

For example, in order to conduct an experiment for rare type of disease, the survey is continued till the required number of patients with disease are collected.

Advantages of Sampling Over Complete Enumeration

- **Reduced cost and enlarged scope**
- **Organization of work**
- **Greater accuracy**
- **Urgent information required**
- **Feasibility (Destructive experiments)**

Type of Surveys:

1. **Demographic Surveys**

2. **Educational Surveys**

3. **Economic Surveys**

4. **Employment Surveys**

5. **Health and Nutrition Surveys**

Type of Surveys:

6. Agricultural Surveys

7. Marketing Surveys

8. Election Surveys

9. Public Polls And Surveys

10. Campus Surveys

Essentials of Data Science With R Software - 2

Sampling Theory and Linear Regression Analysis

Sampling Theory with R Software

:::

Lecture 12

Basic Fundamentals

::

Conducting Surveys and Ensuring Representativeness

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Principal Steps in Conducting a Survey:

1. **Define objective of survey.**
2. **Decide and choose population to be sampled.**
3. **Decide the Data to be collected.**
4. **Decide the degree of precision required.**
5. **Decide the method of measurement.**
6. **Decide the sampling frame.**

Principal Steps in Conducting a Survey:

7. Decide the scheme of selection of sample.

8. Conduct the Pre-test.

9. Organization of the field work.

10. Decide how to present the summary and analysis of data.

11. Decide how to use the information gained for future surveys.

Variability Control in Sample Surveys

The variability control is an important issue any statistical analysis.

A general objective is to draw statistical inferences with minimum variability.

There are various types of sampling schemes which are adopted in different conditions.

These schemes help is controlling the variability at different stages.

Such sampling schemes can be classified in the following way.

Variability Control in Sample Surveys

1. Before selection of sampling units

- **Stratified sampling**
- **Cluster sampling**
- **Two stage sampling**
- **Double sampling etc.**

Variability Control in Sample Surveys

2. At the time of selection of sampling units

- **Systematic sampling**
- **Varying probability sampling**

Variability Control in Sample Surveys

3. After the selection of sampling units

- **Ratio method of estimation**
- **Regression method of estimation**

Note that the ratio and regtresion methods are the methods of estimation and not the methods of drawing samples.

Methods of Data Collection

1. Physical observations and measurements
2. Personal interview
3. Mail enquiry
4. Web based survey
5. Registration
6. Transcription from records
7. Online forms, e.g., Google forms etc.