R Intermediate Short Course
Session 1 - Data Manipulation

Facilitator: Rajesh Lakhan

The University of the West Indies, St Augustine

Monday 18th July 2022
(5:00pm - 7:00pm)
(online)

Opening Items

About Rajesh Lakhan

Friendly Open Environment

Feel free to raise your hand and ask questions

Hopefully open mic as much as possible

More communication is better

Ask away

Getting to know One Another

Start from the participants and go down

- Name
- What you are currently doing: Student, Working...
- Previous Statistical Background?
- Why doing this course/Expected to learn/take away
- Random information about yourself

Acknowledgements

To Ms. Devika Bhagwandin for course material and assistance.

Course Introduction

Course time table:

Dates: Monday 18^{th} July 2022 - Sunday 24^{th} July 2022
Time: 5:00pm - 7:00pm

Course delivery:

Zoom online lectures, Git and Github

To access all files visit:
https://github.com/4Rajesh4/R-intermediate-July-2022
To get updates you need to create an account and "watch" (top right) the repo (repository).
Email notifications and notifications on the home page (when you sign in or click the autocat mascot), will be visible. Select "all activity".

Course Topics

1. Estimation Procedures
2. Hypothesis Testing - one sample and two samples
3. Correlation
4. Analysis of Variance
5. Simple Linear Regression
6. Extra if time allows

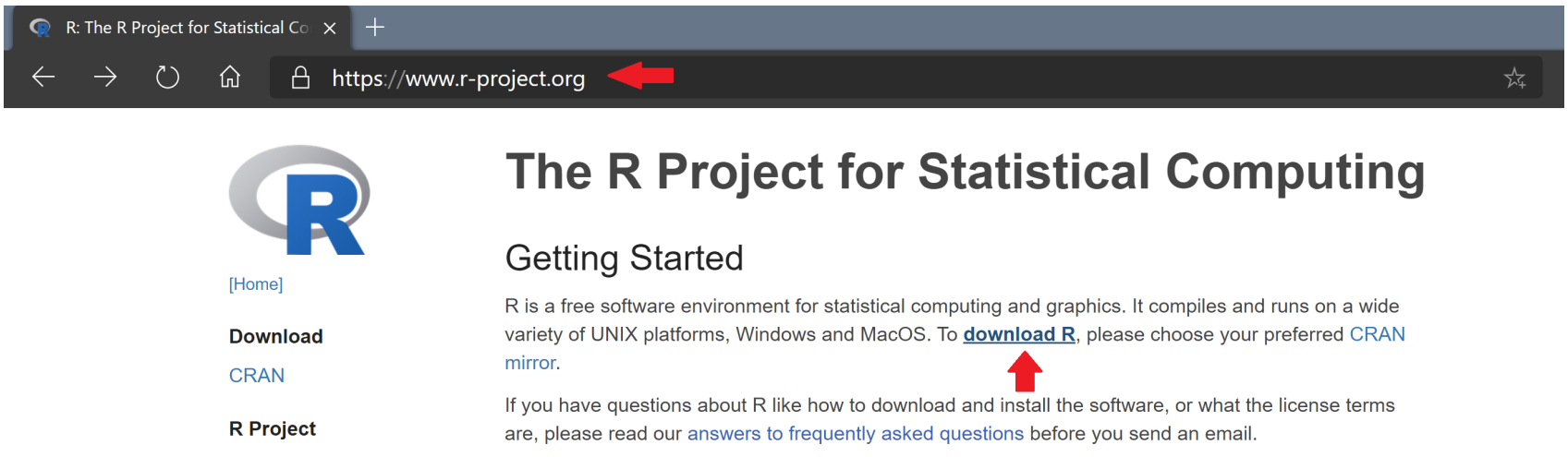
Course Prerequisite:

R Beginner's Course and/or prior statistical knowledge

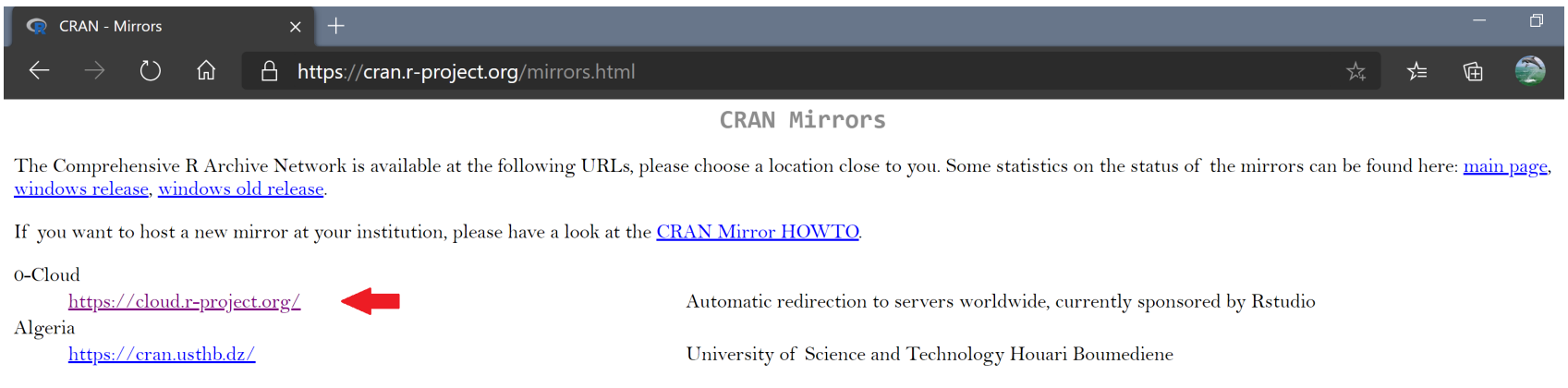
Introduction to R

Steps for downloading R

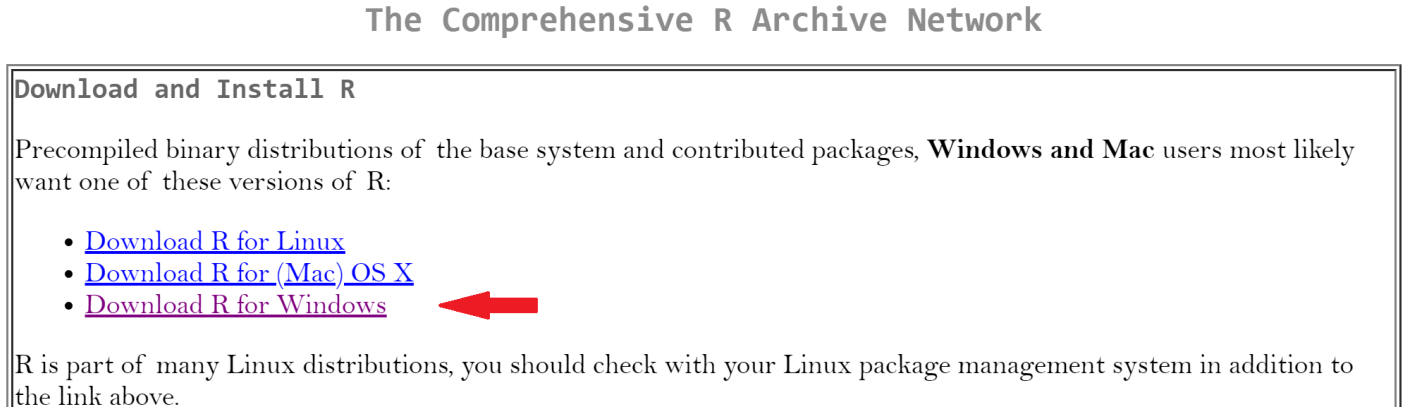
STEP 1:



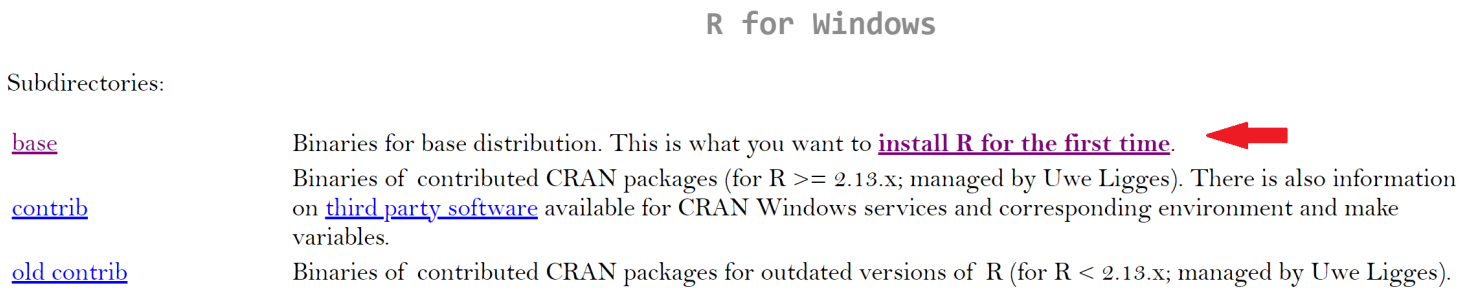
STEP 2:



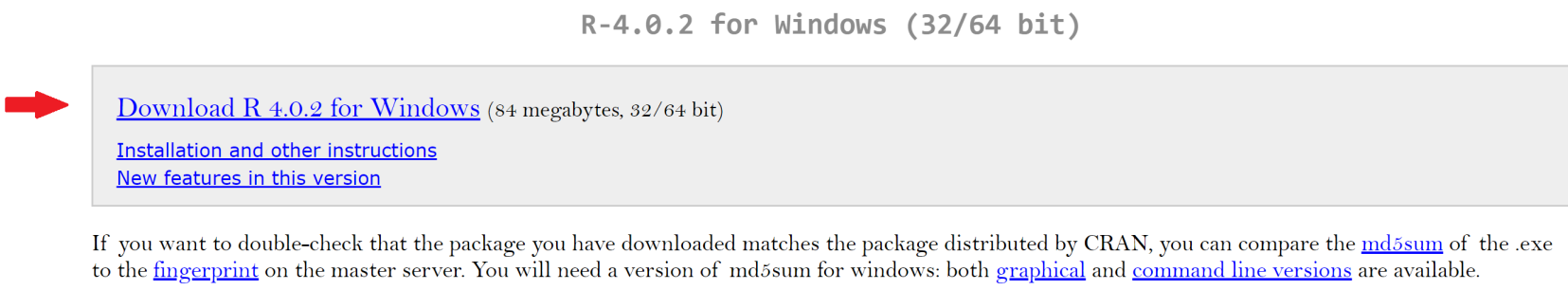
STEP 3:



STEP 4:



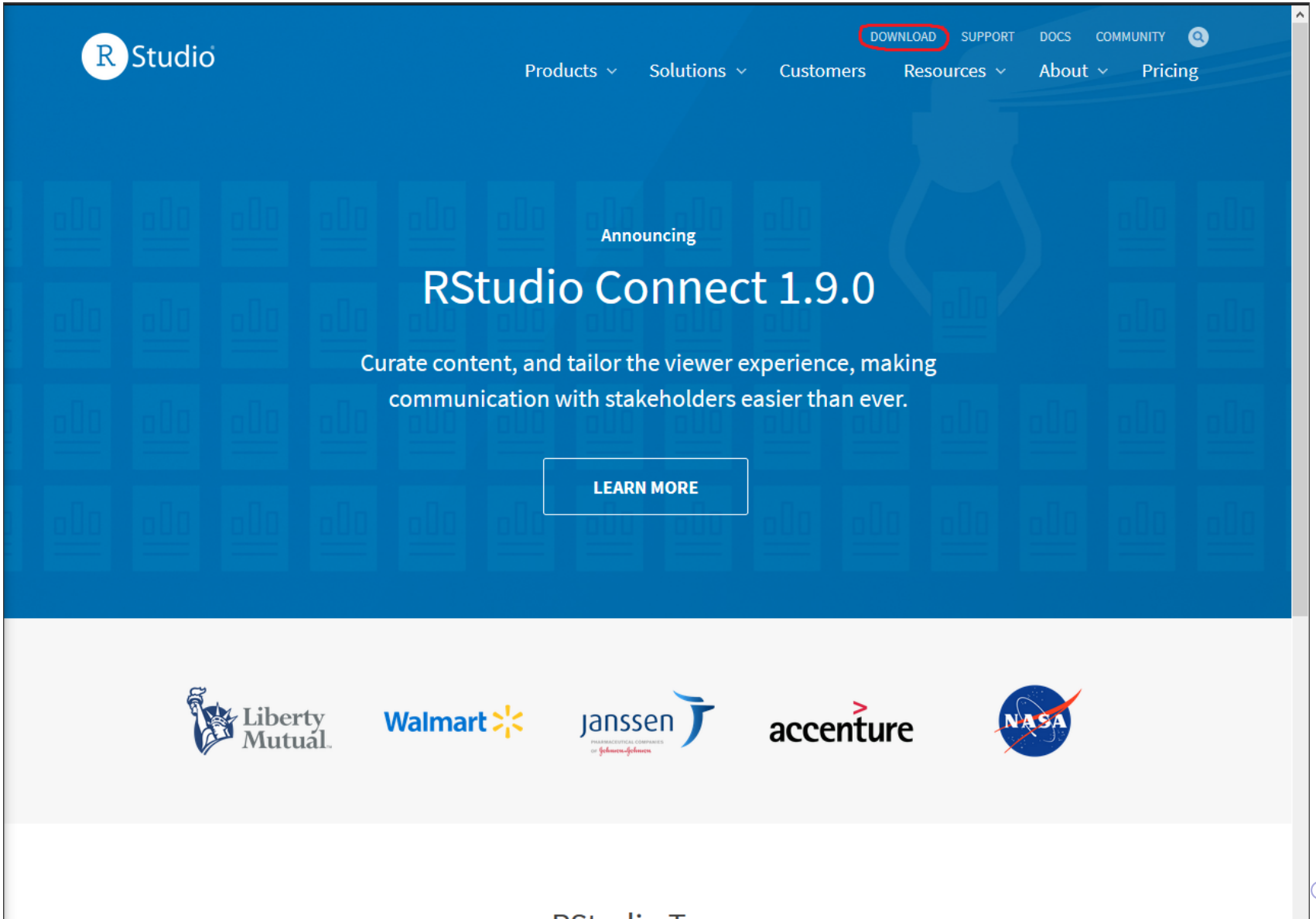
STEP 5:



Introduction to R

Steps for downloading R Studio

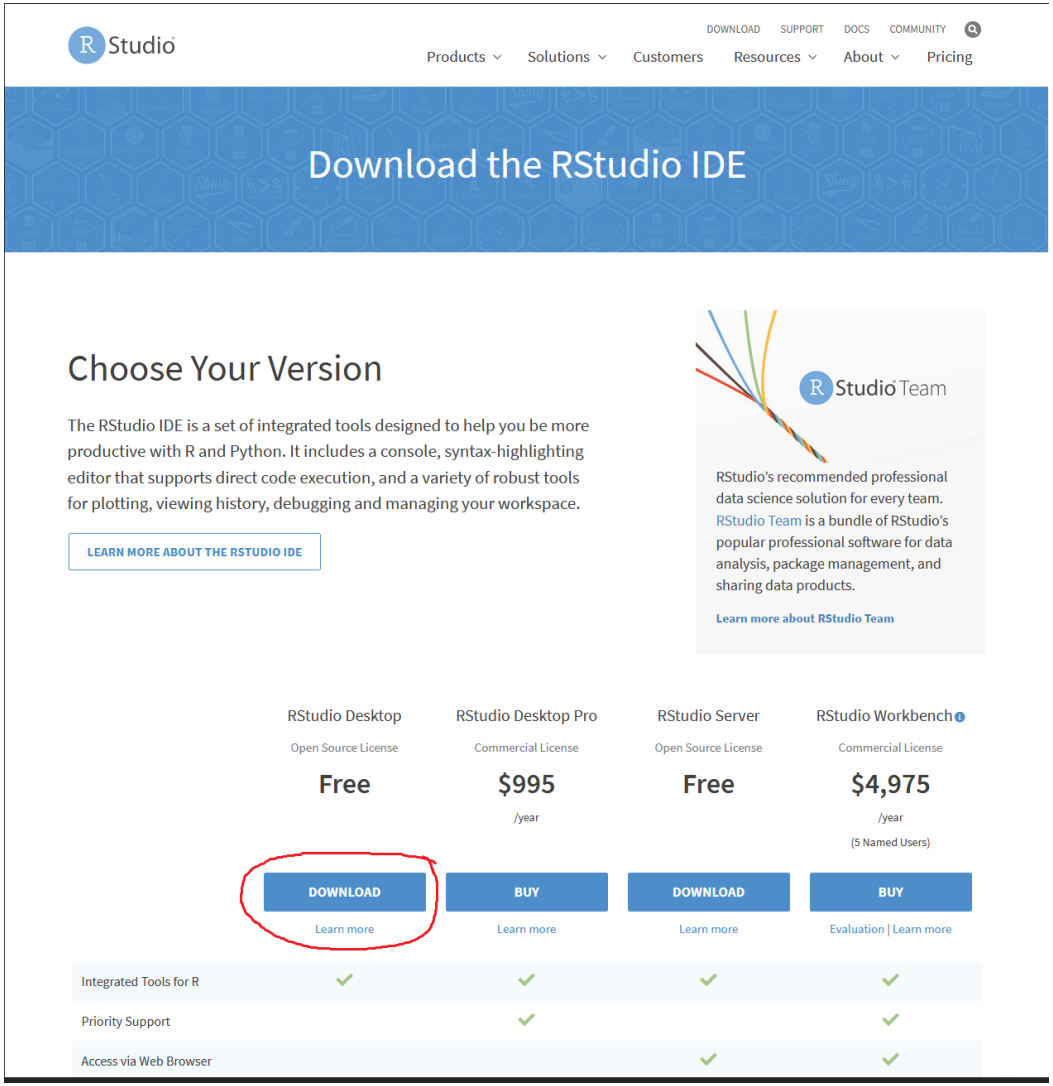
STEP 1:



Installing R Studio

Steps for downloading Rstudio

STEP 2: Selection



Installing R Studio

Steps for downloading Rstudio

STEP 3: Download

RStudio Desktop 1.4.1717 - Release Notes

1. Install R. RStudio requires R 3.0.1+.

2. Download RStudio Desktop. Recommended for your system:

DOWNLOAD RSTUDIO FOR WINDOWS
1.4.1717 | 156.18MB

Requires Windows 10 (64-bit)

All Installers

Linux users may need to import RStudio's public code-signing key prior to installation, depending on the operating system's security policy.

RStudio requires a 64-bit operating system. If you are on a 32 bit system, you can use an older version of RStudio.

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

Zip/Tarballs

Working in RStudio

Change Appearance

Tools -> Global Options -> Appearance

Rearrange Layout

Tools -> Global Options -> Pane Layout

To do:
Click on the Environment, History, Connections... and change to Console.

Working Directory

Written below the Console

getwd() to verify New Method to change working directory:

Navigate in the **Files Pane** to find required folder then:

More -> Set as working Directory

Playing around

Using a script in R

1. Create: File → New Script
2. Save: Save the script created upon closing
3. Re-open: File → Open Script

Installing Packages

Before using a package, the package must first be installed into R then loaded. Packages can be installed one of the three following ways:

1. Using the install command i.e.
   `> install.packages("package name")`
2. Searching for the package and installing within R i.e.
   Package → Install package(s)... → 0-Cloud [https]
3. Downloading the package from an online search then installing into R.
   First download the package as a zipped file from a search engine.
   Then in R: Package → Install package(s) from local files...

Script VS Console (Limited Practical List)

Script:

- Does not run code on "enter"
- You must manually press the 'run button' or 'control + enter key'
- Easy to save work
- Facilitates many lines of code
- ...

Console:

- Runs code on "enter"
- Easy for quick calculations and verification, use case
- Up arrow on keyboard pulls up the previous commands
- Clt + L will clear the console
- ...

Nice Stuff

- The Course Outline is a markdown file
  - Add color
  - Bold
  - Italics
- Anything else
- Website:
- Projects in R Studio

Let's get into code

Simple Operations in R

```
> 1 + 2                 #Addition
> 1 - 2                 #Subtraction
> 5 * 2                 #Multiplication
> 1/2                   #Division
> 2 ^ 4                 #Power
```

Entering data in R

Data can be entered into R in three common ways:

(Also refer to R script)

1. Manually

```
> x = c(1, 2, 3, 4, 5, 6)     #Column vector
> y = t(x)                    #Row vector (transpose
                                 of a column vector)
> z = matrix(x, nrow=2, ncol=3, byrow=T/F)
                                       #2×3 matrix
```

```
> r1 = c(1, 2, 3, 4, 5)
> r2 = c(6, 7, 8, 9, 10)
> table1 = rbind(r1, r2)        #combine by rows
> table2 = cbind(r1, r2)        #combine by columns
                              #manually create table
> colnames(table1) = c("A", "B", "C", "D", "E")
> rownames(table2) = c("A", "B", "C", "D", "E")
```

Exercise:
Create label the matix with the columns as Hi When Five Zoo Final, and the rows as Trinidad, Tobago.

2. Importing a data set

Before reading a data file into R, the working directory must be changed. The working directory should be set to the location of the data set. This can be done by: File $\rightarrow$ Change dir...

Examples of 3 common types of data files:

i. Tab delimited file

```
> data1 = read.table("name.txt",header=T/F)
```

ii. Comma delimited or csv file

```
> data2 = read.csv("name.csv",header=T/F)
```

iii. Excel workbook

R requires the package readxl to read excel files

```
> install.packages("readxl")
> library(readxl)
> data3 = read_excel("name.xlsx")
```

3. Built-in data set

R has several built-in data sets. These are listed in: `> data()`

The data sets need not be loaded or entered manually and can be used directly example `> AirPassengers`

Further information on the data is obtained by: `> ?AirPassengers`

Data Manipulation

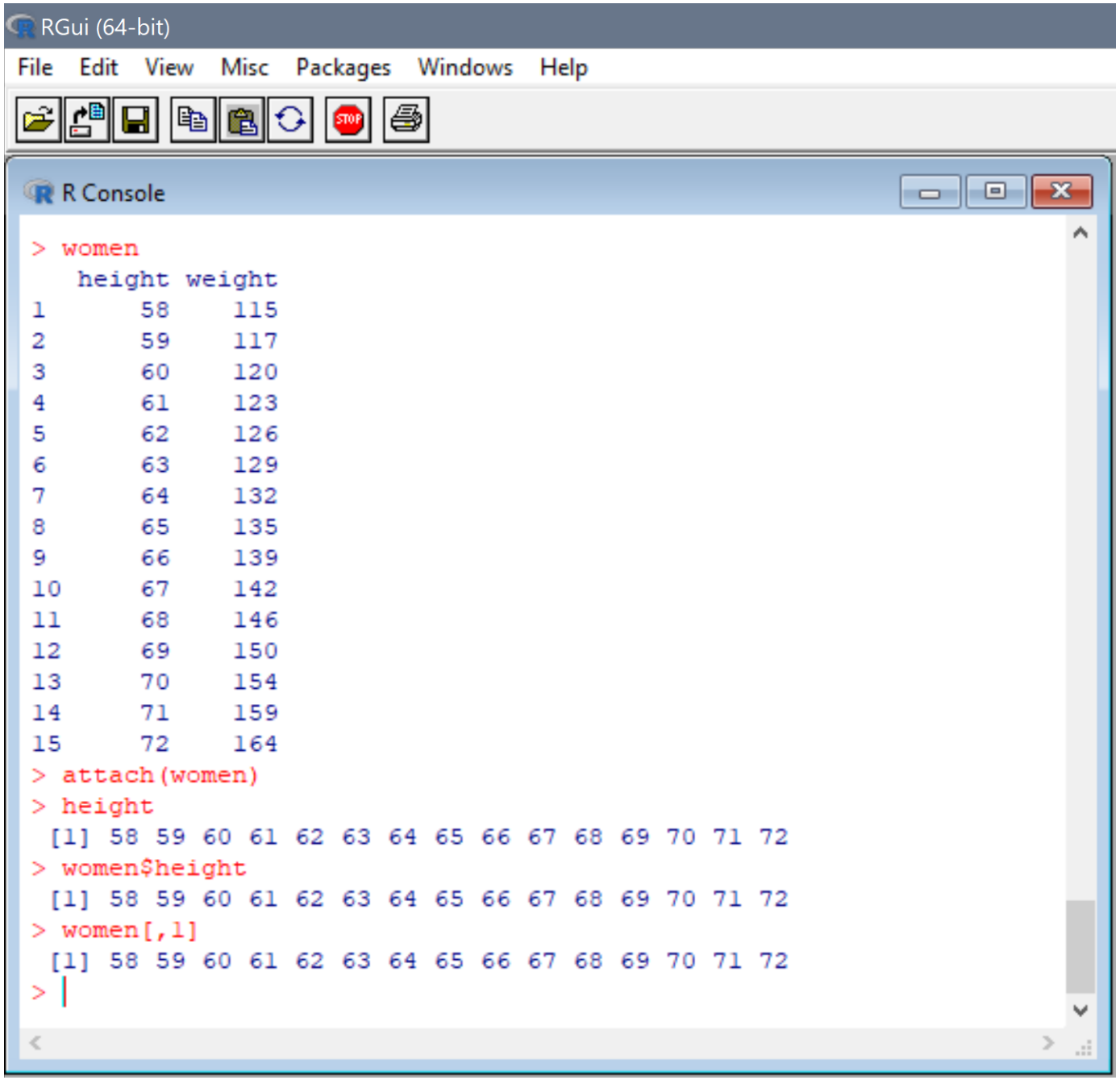
We can extract rows and columns of interest from a data set.

Consider the R built-in data set *women* (also see R output in next slide).

Suppose we are interested in the height variable. This can be extracted in the following ways.

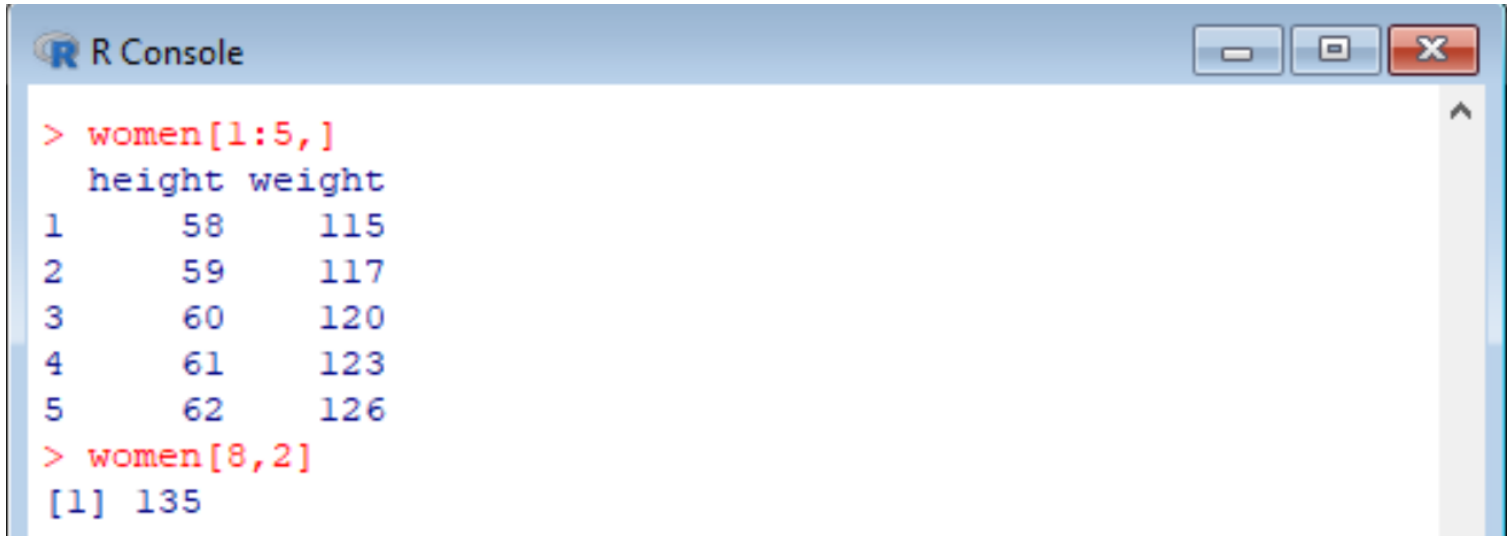
1. Using *attach* command:
   ```
   > attach(women)
   > height
   ```
2. Using $ operator:
   ```
   > women$height
   ```
3. Using square brackets [ ]:

   Data in R is of the form data[row, column].
   Here, height is in column 1 and we require all the rows therefore,
   ```
   > women[,1]
   ```


RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
> women
height weight
1 58 115
2 59 117
3 60 120
4 61 123
5 62 126
6 63 129
7 64 132
8 65 135
9 66 139
10 67 142
11 68 146
12 69 150
13 70 154
14 71 159
15 72 164
> attach(women)
> height
[1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
> women$height
[1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
> women[,1]
[1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
>

Using the square brackets we can also extract rows as well as a range of both rows and columns.

```
> women[1:5,]          #first 5 rows
> women[8, 2]          #row 8 and column 2
```



After the variable of interest is extracted, operations can be performed.

Simple commands

```
> mean(height)                    #mean
> var(height)                     #sample variance
> sd(height)                      #standard deviation
```

```
What does var(women$height) do?
What does var(women[ , 1]) do?
> hist(height, main = "Histogram")    #histogram
```

```
> max(height)                     #maximum
> min(height)                     #minimum
```

Sorting and Ordering data

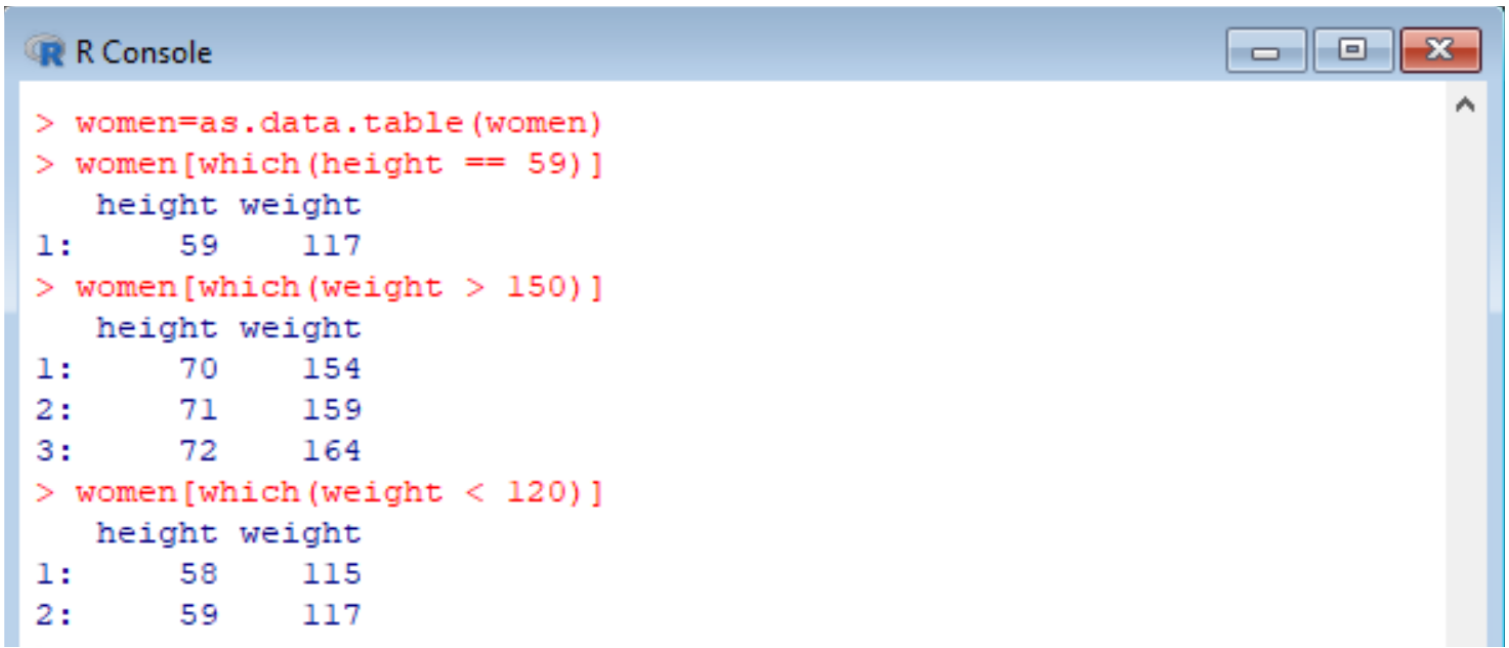
Using the weight variable of the *women* data set,

```
> order(weight)                  #order of magnitude
> weight[order(weight)]              #ascending order
> sort(weight, decreasing=T/F)       #orders data
```

Package *data.table* :

```
> install.packages("data.table")
> library(data.table)

> women = as.data.table(women)
> women[which(height == 59)]
> women[which(weight > 150)]
> women[which(weight < 120)]
```



Worked Example:

Consider the *Diet.csv* data set provided,

a) Suppose we wish to determine the percentage of persons that gained weight after dieting.

Step 1: First read the data file into R.

```
> diet = read.csv("Diet.csv", header = T)
> attach(diet)
> dim(diet)                    #dimension of data set
[1] 78   7
```

Step 2: Find the difference in weight after the diet.

```
> diff = pre.weight - weight6weeks
```

Step 3: Sort the persons that gained weight i.e. those with a negative weight difference.

```
> weightgain = which(diff < 0)
```

Step 4: Determine the number of persons who gained weight, followed by the related percentage.

```
> length(weightgain)
[1] 4
> (4/78) * 100
[1] 5.128205
```

4b) Supposed we want to construct a pie chart to show the percentage of persons that did not gain weight according to the type of diet.

Step 1: Create a data set that includes the difference in weight.

```
> dietnew = cbind(diet, diff)
```

Step 2: Sort the persons which did not gain weight i.e. those with a weight difference ≥ 0.

```
> dietnew = as.data.table(dietnew)
> no.gain = dietnew[which(diff >= 0)]
> attach(no.gain)
```

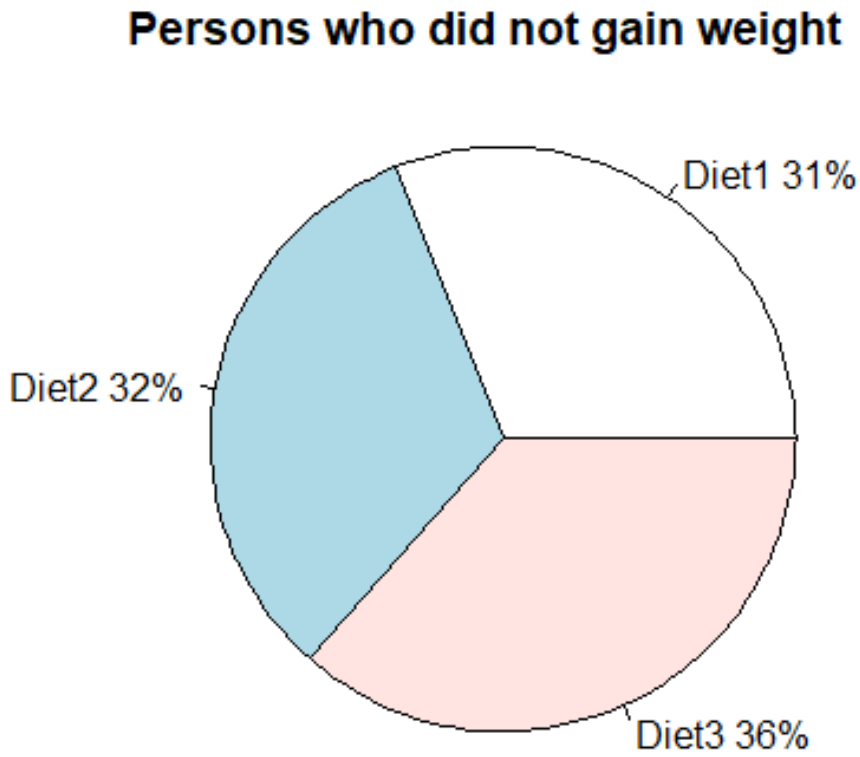
Step 3: Calculate Percentages

```
> diet1 = length(which(Diet == 1))/length(Diet)
> diet2 = length(which(Diet == 2))/length(Diet)
> diet3 = length(which(Diet == 3))/length(Diet)
> pct = c(diet1, diet2, diet3) * 100
> percent = paste(round(pct), "%", sep="")
                        #round off and add % symbol
```

Step 4: Construct Pie Chart

```
> Diet = table(Diet)
> diet.label = c("Diet1", "Diet2", "Diet3")
> labels = paste(diet.label, percent)
> pie(Diet, labels, main = "Persons who did not
  gain weight")
```

Also see R script for colour options and legend.



Note that the percentages can also be found by:

```
> Diet = table(no.gain$Diet)
> pct = (Diet/sum(Diet))*100
```

Closing for Day 1

- See R Code for more tweaks
- Tomorrow we look at estimation and Hypothesis Testing
- Thank you