Creating Script file and Plotting

Zulquar Nain
AMU

2020-04-17

Script file

R script?

While you can run/execute **r code** using `R console` with ease.

It is time consuming.

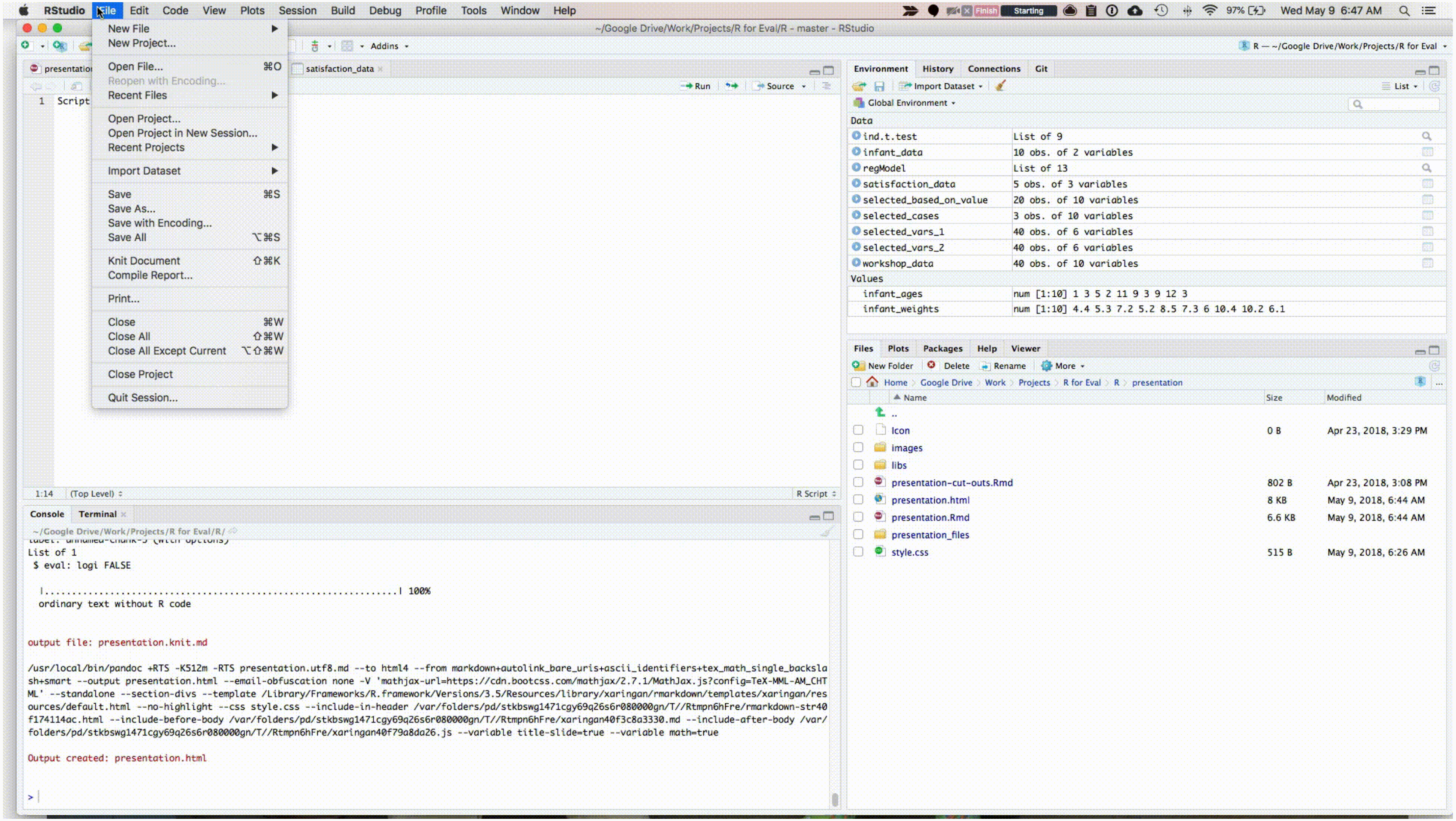
Each time you have to re-enter a command to execute it.

- Be calm, we have solution that is `R script`

A script is simply a text file containing a set of commands and comments. The script can be saved and used later to re-execute the saved commands. The script can also be edited so you can execute a modified version of the commands.

Creating R Script

Create new script file: File -> New File -> R Script



Data

ind.t.test	List of 9
infant_data	10 obs. of 2 variables
regModel	List of 13
satisfaction_data	5 obs. of 3 variables
selected_based_on_value	20 obs. of 10 variables
selected_cases	3 obs. of 10 variables
selected_vars_1	40 obs. of 6 variables
selected_vars_2	40 obs. of 6 variables
workshop_data	40 obs. of 10 variables

Values

infant_ages	num [1:10] 1 3 5 2 11 9 3 9 12 3
infant_weights	num [1:10] 4.4 5.3 7.2 5.2 8.5 7.3 6 10.4 10.2 6.1

Name	Size	Modified
..		
Icon	0 B	Apr 23, 2018, 3:29 PM
images		
libs		
presentation-cut-outs.Rmd	802 B	Apr 23, 2018, 3:08 PM
presentation.html	8 KB	May 9, 2018, 6:44 AM
presentation.Rmd	6.6 KB	May 9, 2018, 6:44 AM
presentation_files		
style.css	515 B	May 9, 2018, 6:26 AM

```
label: unnamed-chunk-5 (with options)
List of 1
 $ eval: logi FALSE

  |.................................................................| 100%
  ordinary text without R code


output file: presentation.knit.md

/usr/local/bin/pandoc +RTS -K512m -RTS presentation.utf8.md --to html4 --from markdown+autolink_bare_uris+ascii_identifiers+tex_math_single_backsla
sh+smart --output presentation.html --email-obfuscation none -V 'mathjax-url=https://cdn.bootcss.com/mathjax/2.7.1/MathJax.js?config=TeX-MML-AM_CHT
ML' --standalone --section-divs --template /Library/Frameworks/R.framework/Versions/3.5/Resources/library/xaringan/rmarkdown/templates/xaringan/res
ources/default.html --no-highlight --css style.css --include-in-header /var/folders/pd/stkbswg1471cgy69q26s6r080000gn/T//Rtmpn6hFre/rmarkdown-str40
f174114ac.html --include-before-body /var/folders/pd/stkbswg1471cgy69q26s6r080000gn/T//Rtmpn6hFre/xaringan40f3c8a3330.md --include-after-body /var/
folders/pd/stkbswg1471cgy69q26s6r080000gn/T//Rtmpn6hFre/xaringan40f79a8da26.js --variable title-slide=true --variable math=true

Output created: presentation.html

>
```

Find detailed description HERE

Plotting

Image:BoldBI

Visualization

Data visualization is the process to transform the information (data) into a visual presentation for example graph.

Visualisation / Plotting is one of greatest strength of `R`

Why `visualisation/Plotting`?

An image speaks louder than words

Data visualizations make data easier for the human brain to understand

visualization also makes it easier to detect patterns, trends, and outliers in groups of data

Good data visualizations should place meaning into complicated datasets so that their message is clear and concise.

According to Tableau, “[data visualization is] one of the most useful professional skills to develop. The better you can convey your points visually, the better you can leverage that information.”

Ingredients for plotting

Data

Materials to visualise that is data. No data no visualisation!

Mapping: Contextual relationship

Mapping depends on what YOU want to show!

Data

Import

See last lecture for importing external data

Mapping

We will learn! A basic graph

```
plot(cars$speed, cars$dist, pch = 19, col = 'red', las = 1, xlab="speed", ylab="Distance",
```

Plotting- Setting

We will use inbuild data sets in `R`

To view available datasets in `R` Type `data()` and execute

We will primarily use `data(cars)`

Most used function for plotting in `R` is `plot()`

Data-Cars

```
data(cars)
```

Examining data

Do you remember? `head()` ; `tail()` ; `nrow()`

```
head(cars, 2)
```

```
##   speed dist
## 1     4    2
## 2     4   10
```

```
tail(cars, 2)
```

```
##    speed dist
## 49    24  120
## 50    25   85
```

```
ncol(cars)
```

```
## [1] 2
```

```
str(cars)
```

```
## 'data.frame':    50 obs. of  2 variables
##  $ speed: num  4 4 7 7 8 9 10 10 10 11
##  $ dist : num  2 10 4 22 16 10 18 26 34
```

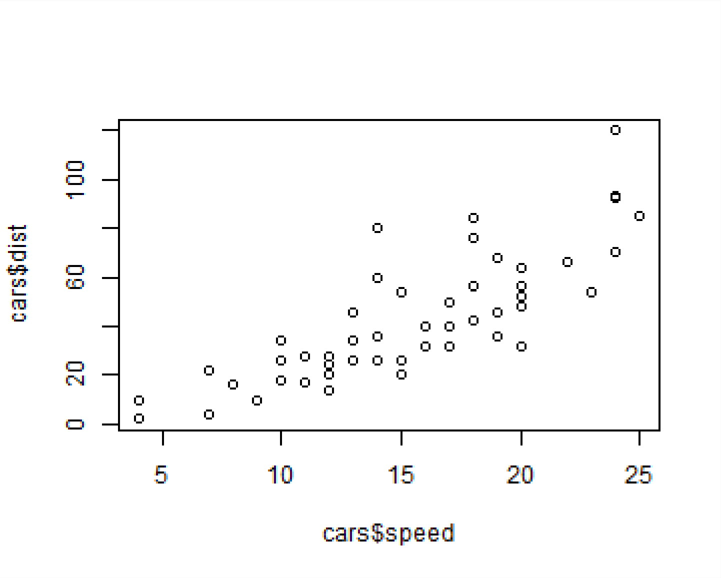
Let's start- `Plot()`

`data(cars)` contains two variables `speed` and `distance`

First plot

Plotting speed and distance

```
plot(cars$speed,cars$dist)
```

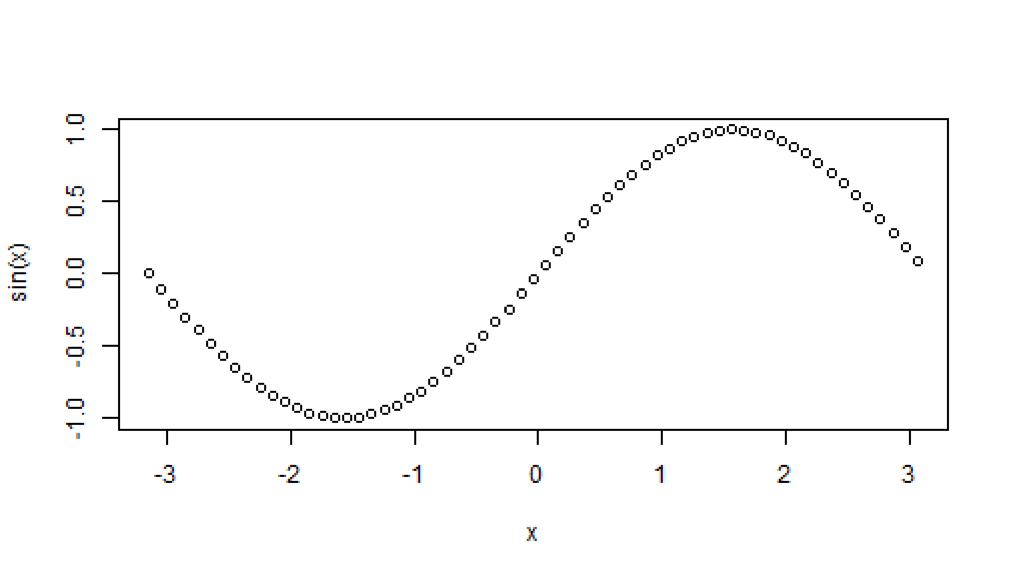


- Here, `cars$speed` is for `x-axis` and `cars$dist` is for `y-axis`
- In `cars$speed`, `cars` is name of the data file and `speed` is variable name
- `plot()` is command to plot

Let's start- `Plot()`

Second plot

```
x <- seq(-pi,pi,0.1)
plot(x, sin(x))
```

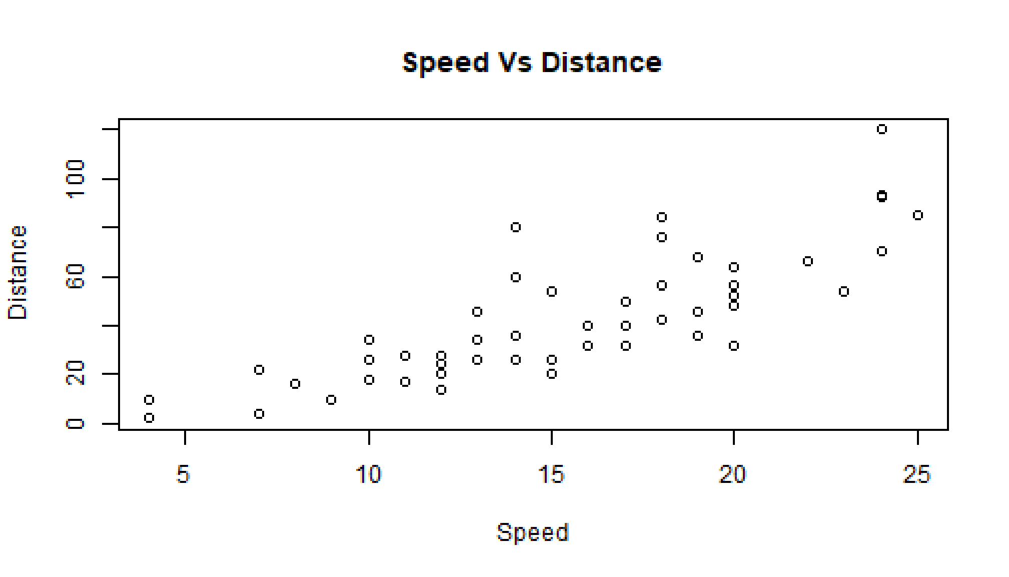


- Here `x` is for `x-axis` (a generated data using `seq` command)
- `sin(x)` is for `y-axis`

Let's start- `Plot()`

Adding label and Title

```
plot(cars$speed, cars$dist,
     xlab = "Speed", ylab = "Distance", main = "Speed Vs Distance" )
```



- Here to add the label, we have added the highlighted codes.
- Names of the label should always be in `" "`

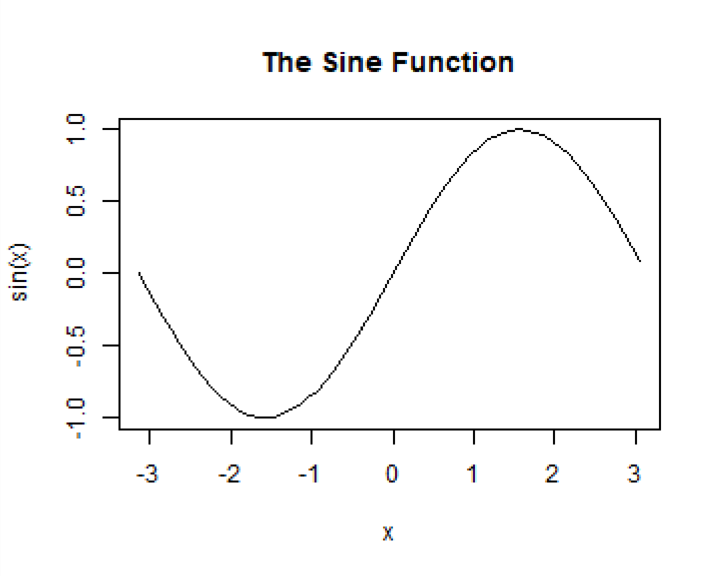
Let's start- `Plot()`

Changing Color and Plot Type

- We can change the plot type with the argument `type`

```
"p" - points
"l" - lines
"b" - both points and lines
"c" - empty points joined by lines
"o" - overplotted points and lines
"s" and "S" - stair steps
"h" - histogram-like vertical lines
"n" - does not produce any points or lines
```

```
plot(x, sin(x),
main="The Sine Function",
ylab="sin(x)",
type="l" )
```

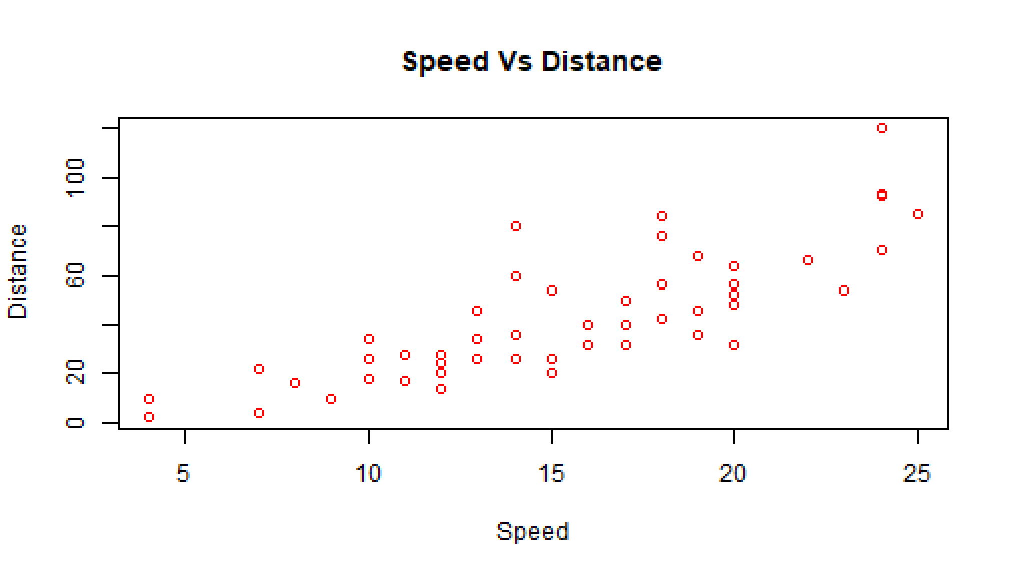


Let's start- `Plot()`

Changing Color and Plot Type

- Similarly, we can define the colors using `col="color name"`

```
plot(cars$speed, cars$dist,
     xlab = "Speed", ylab = "Distance", main = "Speed Vs Distance",
     col="red" )
```



- See the highlighted part of the code

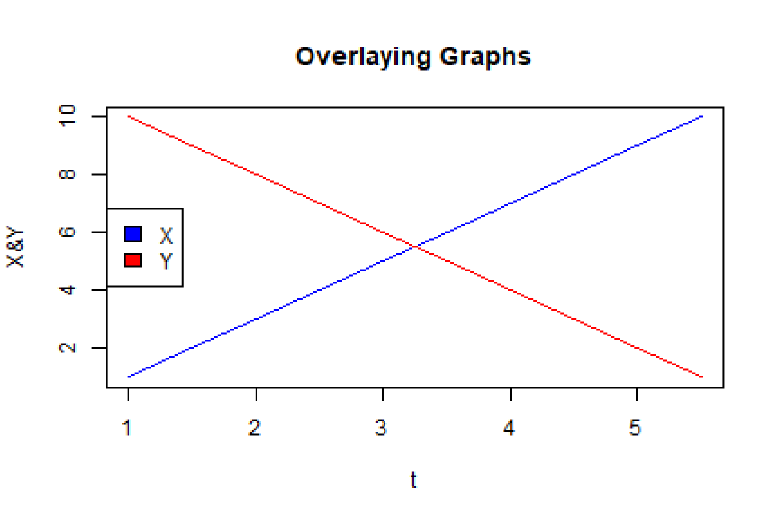
Let's start- `Plot()`

Overlaying Plots

- Do YOU remember the function/command? `seq` -- we will use HERE to generate the variables `X, Y, t`

- plotting

```
plot(t, X,
     main="Overlaying Graphs",
     ylab="",
     type="l",
     col="blue")
lines(t, Y, col="red")
legend("left",
       c("X","Y"),
       fill=c("blue","red")
)
```



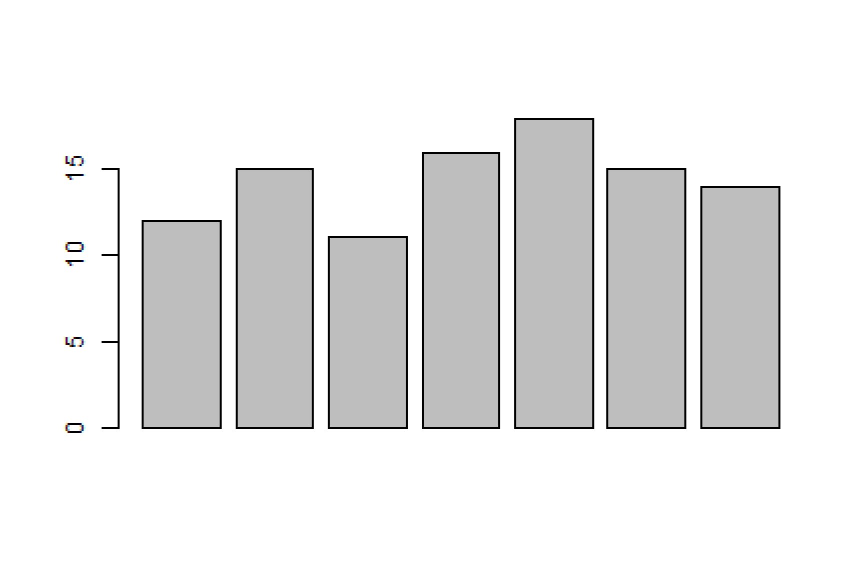
- Checkout `?plot()` and `?legend`

Some Baisc Graphs

R Bar Plot

- Let's assume `AR` contains data of average rainfall in a day of a week.

```
barplot(AR)
```



- There are many other parameters can be added to `barplot()`
- Use `?barplot()` to explore

Some Baisc Graphs

R Bar Plot

- Some of the parameters are added here.

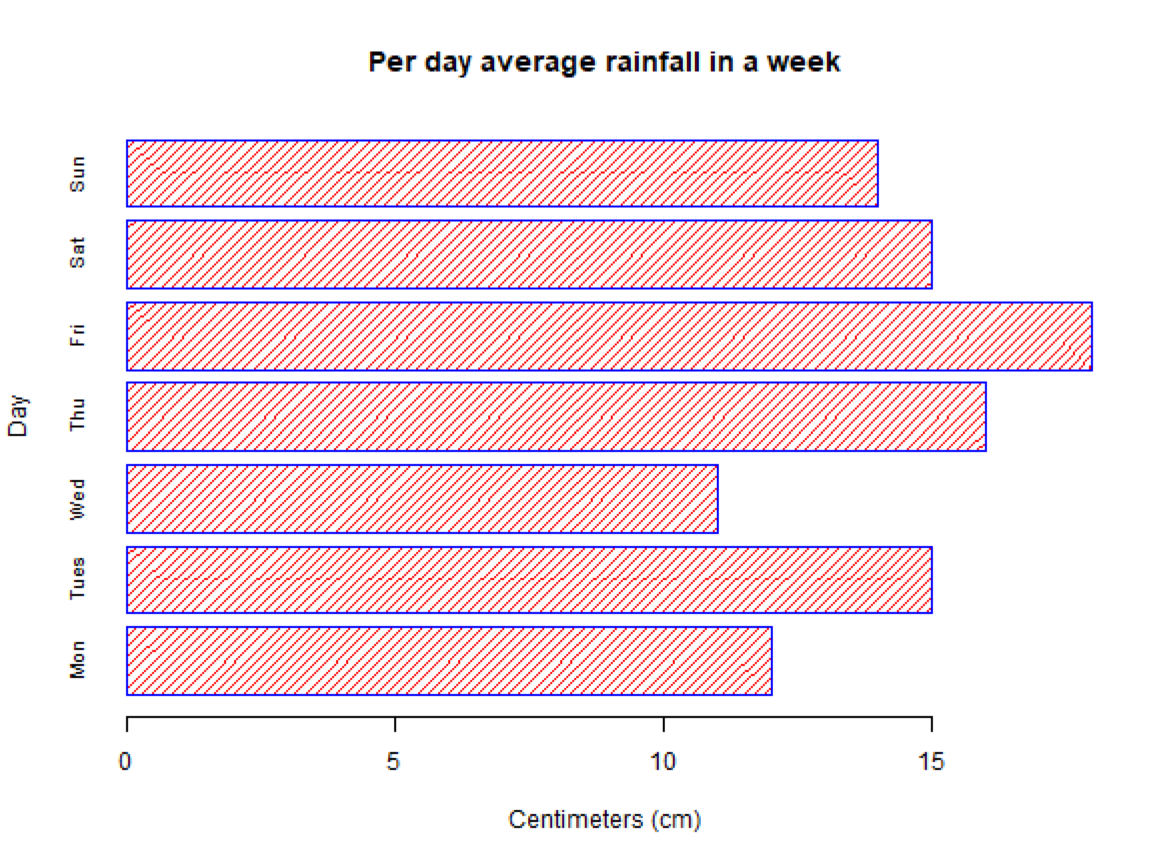
```r
barplot(AR,
main = "Average rainfall in a Day",
xlab = "Centimeters (cm)",
ylab = "Day",
names.arg = c("Mon", "Tues", "Wed", "Thu", "Fri", "Sat", "Sun"),
border="blue",
col="red",
density=20,
horiz = TRUE,
cex.names = .8)#To change the size of label
```

- See the highlighted codes
- Output in next slide

Some Baisc Graphs

R Bar Plot

- Some of the parameters are added here.



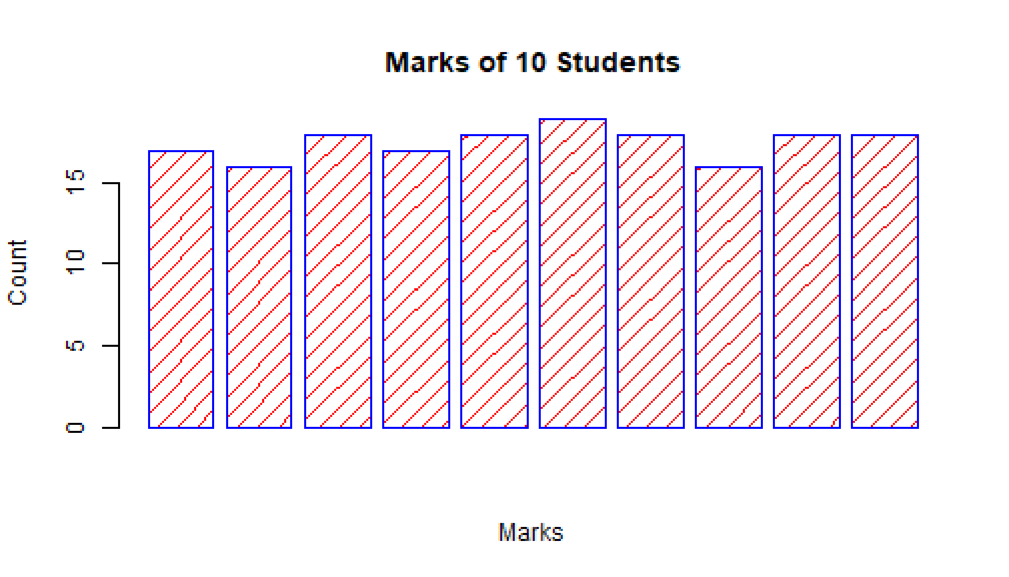
Some Baisc Graphs

Bar Plot of Categorical Data

- For example marks out of 20 of ten students in Math is in vector `MM`

```
##  [1] 17 16 18 17 18 19 18 16 18 18
```

- Simple bar plot



- Does it serve pupose? -- No

Some Baisc Graphs

Bar Plot of Categorical Data

- First convert the data into categorical representation using `table()`
- Check out `?table()`

```
table(MM)
```

```
## MM
## 16 17 18 19
##  2  2  5  1
```

```
barplot(table(MM),
main="Marks of 10 Students",
xlab="Marks",
ylab="Count",
border="blue",
col="red",
density=10
)
```

Some Baisc Graphs

Bar Plot of Categorical Data

Some more Bar plot

```
print(titanic_surv)
```

```
##               train.Pclass
## train.Survived   1   2   3
##              0  80  97 372
##              1 136  87 119
```

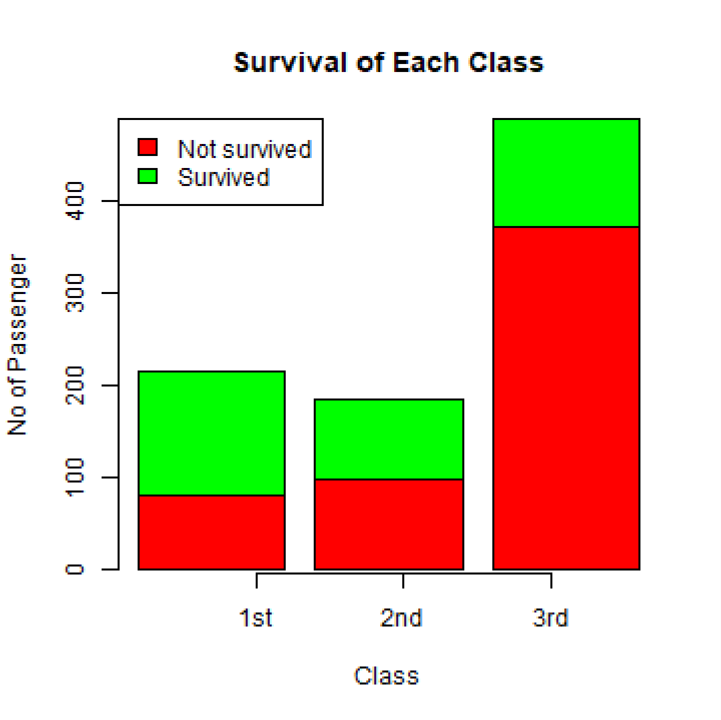
- Here, 1, 2, and 3 represents 1st, 2nd and 3rd class in the train
- 0 and 1 is for the passenger did not survived and survived respectively in the Titanic mishap

```
barplot(titanic_surv,
        main = "Survival of Each Class",
        xlab = "Class",
        ylab = "No of Passenger",
        col = c("red","green")
)
legend("topleft",
       c("Not survived","Survived"),
       fill = c("red","green")
)
```

Some Baisc Graphs

Bar Plot of Categorical Data

Some more Bar plot



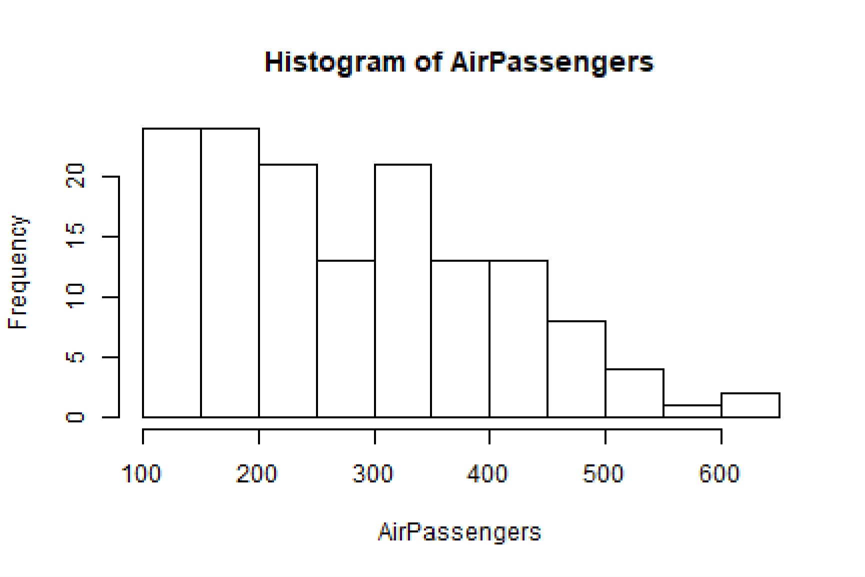
Some Baisc Graphs

Histogram- `hist()`

- Histogram is a visual representation of the distribution of a dataset
- We will use the `data(AirPassengers)` in built in `R`
- Explore the function `hist()` using `?hist()`
- A Basic Histogram

- put the name of your dataset in between the parentheses like `hist(AirPassengers)`
- Histogram for a specific variable can be drawn as `hist(datasetName$VariableName)`

```
hist(AirPassengers)
```

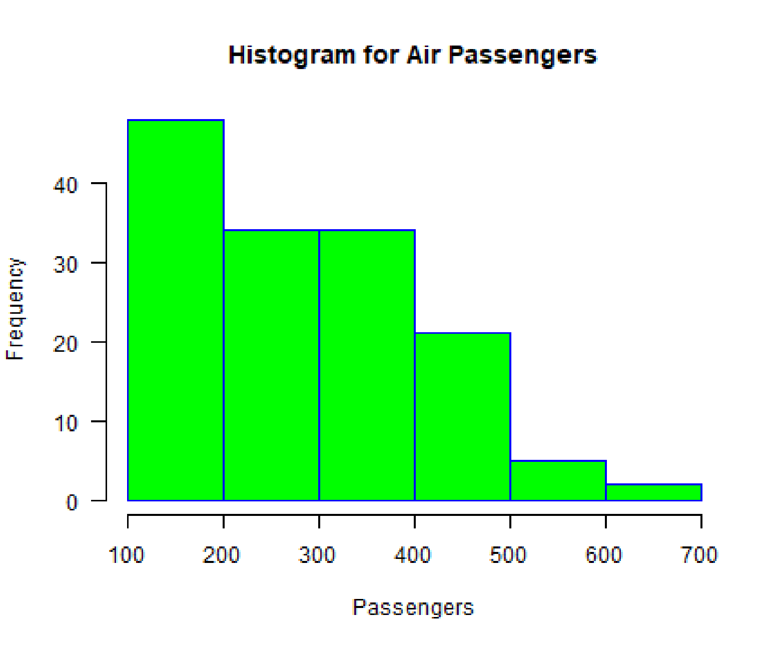


Some Baisc Graphs

Histogram- `hist()`

- Other parameters of `hist()`

```
hist(AirPassengers,
     main="Histogram for Air Passengers",
     xlab="Passengers",
     border="blue",
     col="green",
     xlim=c(100,700),
     las=1,
     breaks=5)
```



- `xlim=c()` & `ylim=c()` fixes the range of `X` and `Y` axes
- Inside `c()` sets starting and ending points
- `las=1` rotates the label of `Y-axis` Checkout `?las`
- `breaks` is for the size/width of `Histogram BINS` Chechout `?breaks`

Some Baisc Graphs

Pie Chart-`Pie Chart`

- Pie chart is drawn using the `pie()` function in `R` programming
- This function takes in a vector of non-negative numbers.
- Basic Syntax of is `pie(x, labels, radius, main, col, clockwise)`

> - x is a vector containing the numeric values used in the pie chart.
> - labels is used to give description to the slices.
> - radius indicates the radius of the circle of the pie chart.(value between –1 and +1).
> - main indicates the title of the chart.
> - col indicates the color palette.
> - clockwise is a logical value indicating if the slices are drawn clockwise or anti clockwise.

- Explore `?pie()`

Some Baisc Graphs

Pie Chart-`Pie Chart`

- In `pie()`, `scores$Obt.Marks` is the vector of positive numbers for which `pie-chart` is drawn
- `scores$Subjects` is the labels
- Note: `scores$Subjects` shows that `Subjects` variable has been selected from `scores` dataset

- `Pie Chart`

```
pie(scores$Obt.Marks, scores$Subjects)
```

- Data

```
print(scores)
```

```
##   Subjects Obt.Marks
## 1     Math        70
## 2      Eng        80
## 3     Urdu        60
## 4       Sc        80
## 5     Soc.        90
```

Some Baisc Graphs

Pie Chart-`Pie Chart`

Other parameters

```r
piepercent<- round(100*(scores$Obt.Marks)/sum((scores$Obt.Marks)), 1) # %age calculation
pie(scores$Obt.Marks, labels = piepercent, # Labels
    main = "Scores pie chart", # Title of chart
    col = rainbow(length(scores$Obt.Marks))) # Color of chart
legend("topright", # legend position
       scores$Subjects, # legend labels
       cex = 0.8, # size of legend texts
   fill = rainbow(length(scores$Obt.Marks))) # legend color
```

Some Baisc Graphs

Scatterplot Matrix

- In case of more than two variables and to find the correlation between one variable versus the remaining ones

- we use scatterplot matrix. `pairs()` function creates matrices of scatterplots.

- `pairs(formula, data)`

- We will use `data(mtcars)` available within `R`; explor `?mtcars`

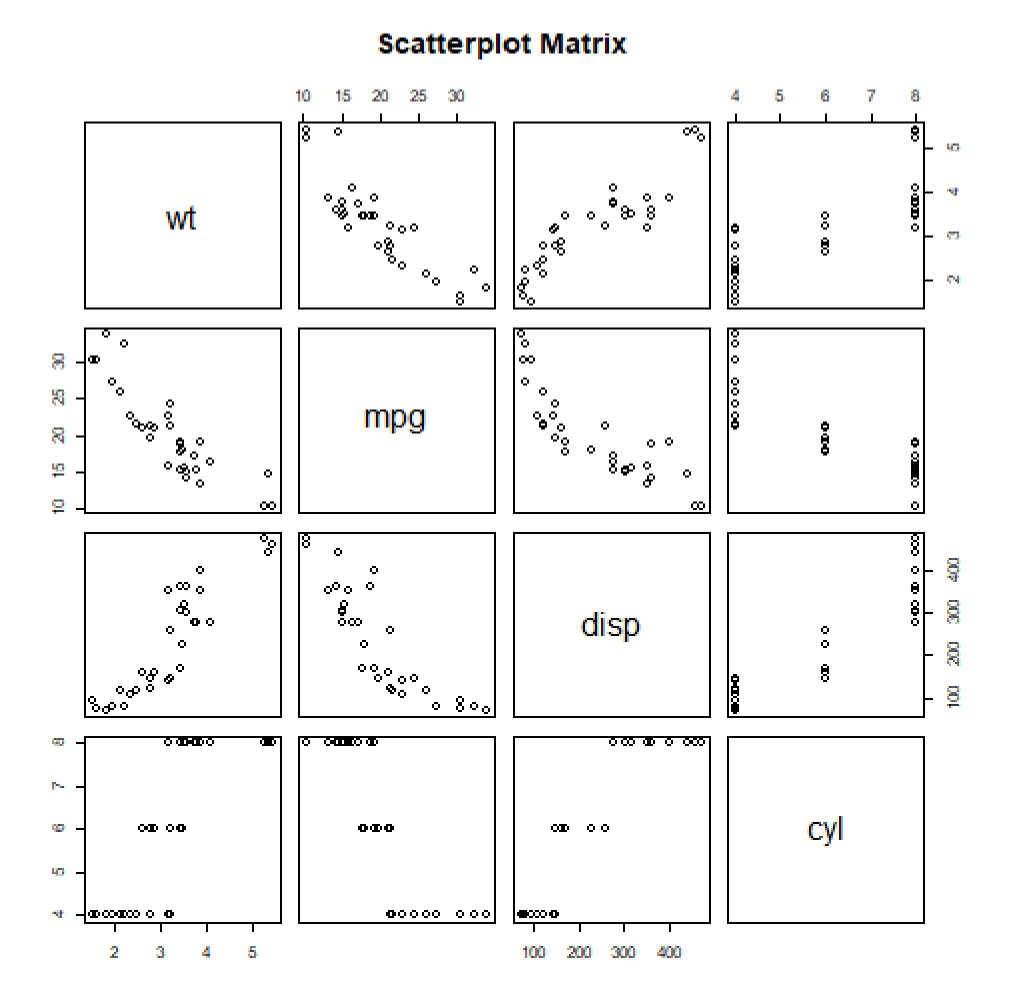
```
##                mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4      21   6  160 110  3.9 2.620 16.46  0  1    4    4
## Mazda RX4 Wag  21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

```
pairs(~wt+mpg+disp+cyl,data = mtcars,
   main = "Scatterplot Matrix")
```

- Plot in the next slide

Some Baisc Graphs

Scatterplot Matrix



Multiple Plots

R Function `par()`

- For drawing multiple graphs in a single plot- use `par()`
- Checkout `?par()`

Let's take an example

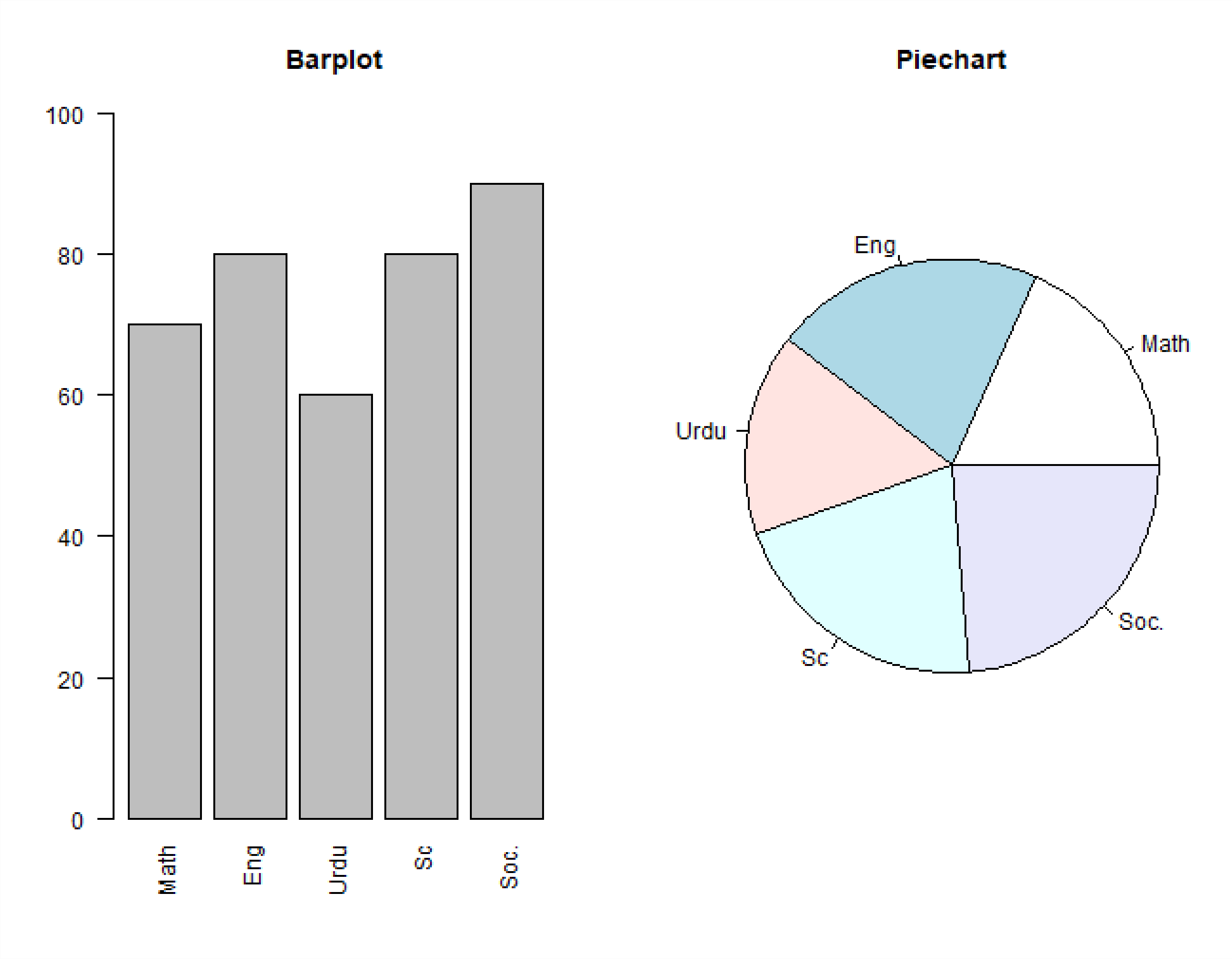
- For drawing two graphs in one plot

```
par(mfrow=c(1,2))    # set the plotting area into a 12 array (1 Row and 2 Col)
barplot(scores$Obt.Marks, names.arg = scores$Subjects, main="Barplot", las=2) # Bar plot
pie(scores$Obt.Marks, scores$Subjects, main="Piechart", radius=1) # Pie Chart
```

- See the graph in next slide
- Here parameter `mfrow` used to specify the number of subplot we need.
- It takes in a vector of form `c(m, n)` which divides the given plot into `m*n` array of subplots.
- For the above example, to plot the two graphs side by side, we have `m=1` and `n=2`.

Multiple Plots

R Function `par()` - Explore it for more control parameters



Saving / Exporting Graph

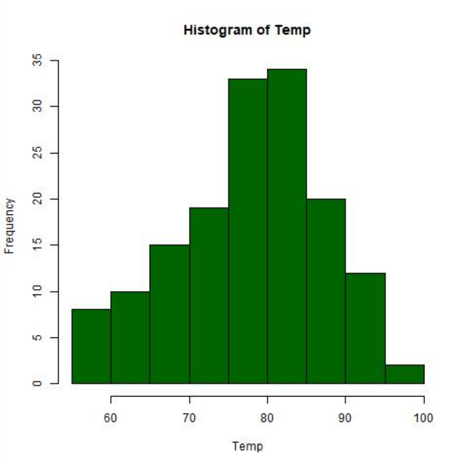
- All types of graphs `(bar plot, pie chart, histogram)` etc. can be saved.
- Graphs can be saved as bitmap image(i.e. .png, jpeg, tiff etc) which are fixed size
- Graphs can be also saved as vector image (.pdf, .eps) which are easily resizable
- We will use the `temperature` column of built-in dataset `airquality`

Saving / Exporting Graph

Saving as `.jpeg`

```
jpeg(file="saving_plot1.jpeg")
# File name
hist(Temp, col="darkgreen")
dev.off() # TO call off
```

Saved Graph



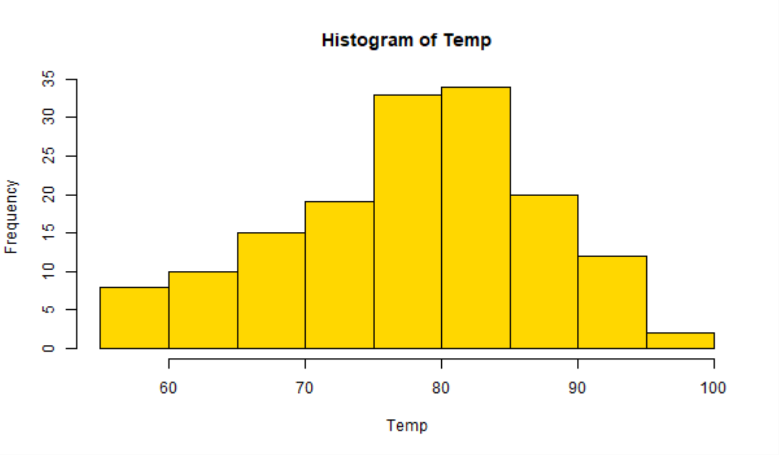
- Image will be saved in `working/default directory`
- we need to call the function `dev.off()` after all the plotting, to save the file and return control to the screen
- The resolution of the image by default will be 480×480 pixel.

Saving / Exporting Graph

Saving as `.png`

```
png(file= "F:/MEGAsync/AMU class Jan-May 2
width=600, height=350)
hist(Temp, col="gold")
dev.off()
```

Saved Graph



- You can specify the full path tp save the image at desired plcae (as above)
- You can also specify the resolution at desired level using arguments `width` and `height`

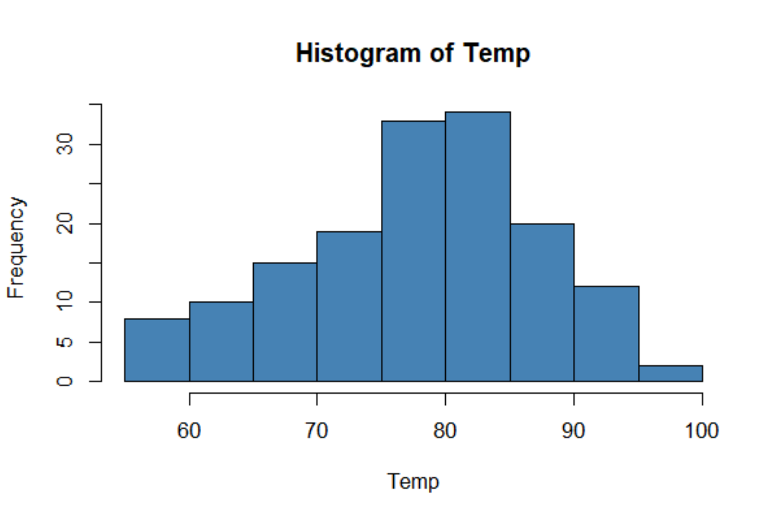
Saving / Exporting Graph

Saving as `.bmp`

- Size of the plot can be specified in different units such as in inch, cm or mm with the argument `units` and ppi with `res`.
- The following code saves a bmp file of size `6x4 inch` and `100 ppi`.

```
bmp(file="saving_plot3.bmp",
width=6, height=4, units="in", res=
hist(Temp, col="steelblue")
dev.off()
```

Saved Graph



Saving / Exporting Graph

Saving as `.pdf`

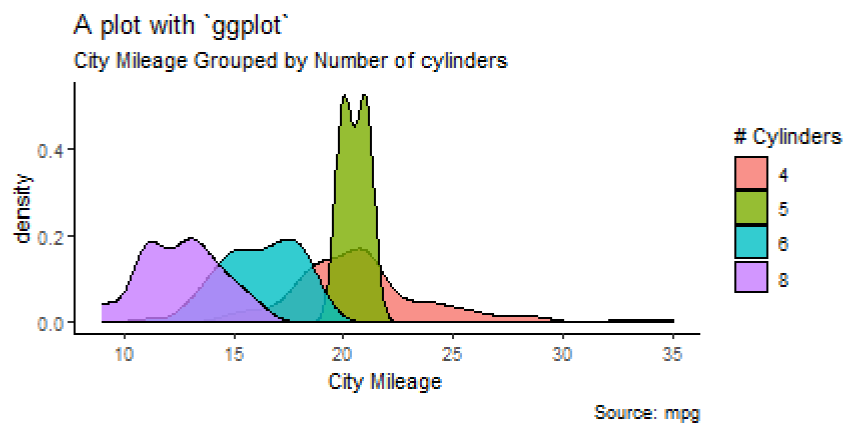
```
bmp(file="saving_plot4.pdf",
width=6, height=4, units="in", res=100)
hist(Temp, col="violet")
dev.off()
```

Saved Graph

Plotting in R

- This presentation is not exhaustive.
- Adopt learning by doing approach
- Make use of `Google` and `R Documentation`
- It was about basic `R` plotting
- Plotting has become more exciting and easy using `package-ggplot2` in `R`
- What is `package`? See the next slide!

--



A plot with `ggplot`

City Mileage Grouped by Number of cylinders

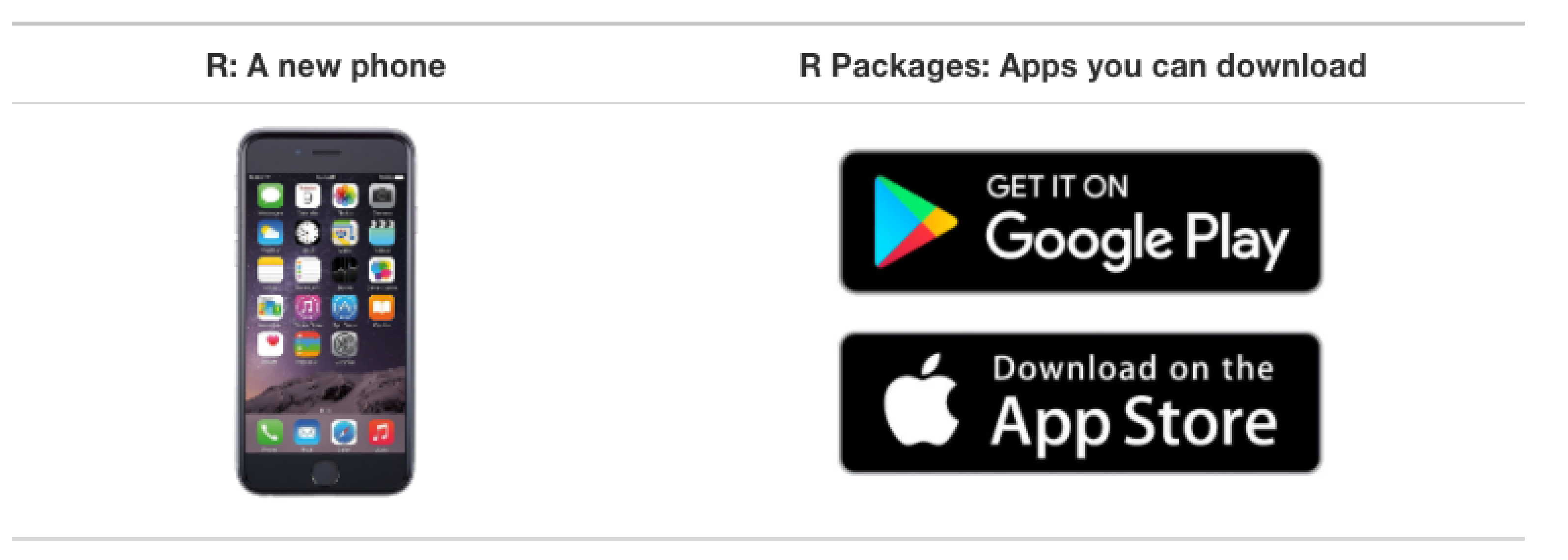
Source: mpg

What is `packages`

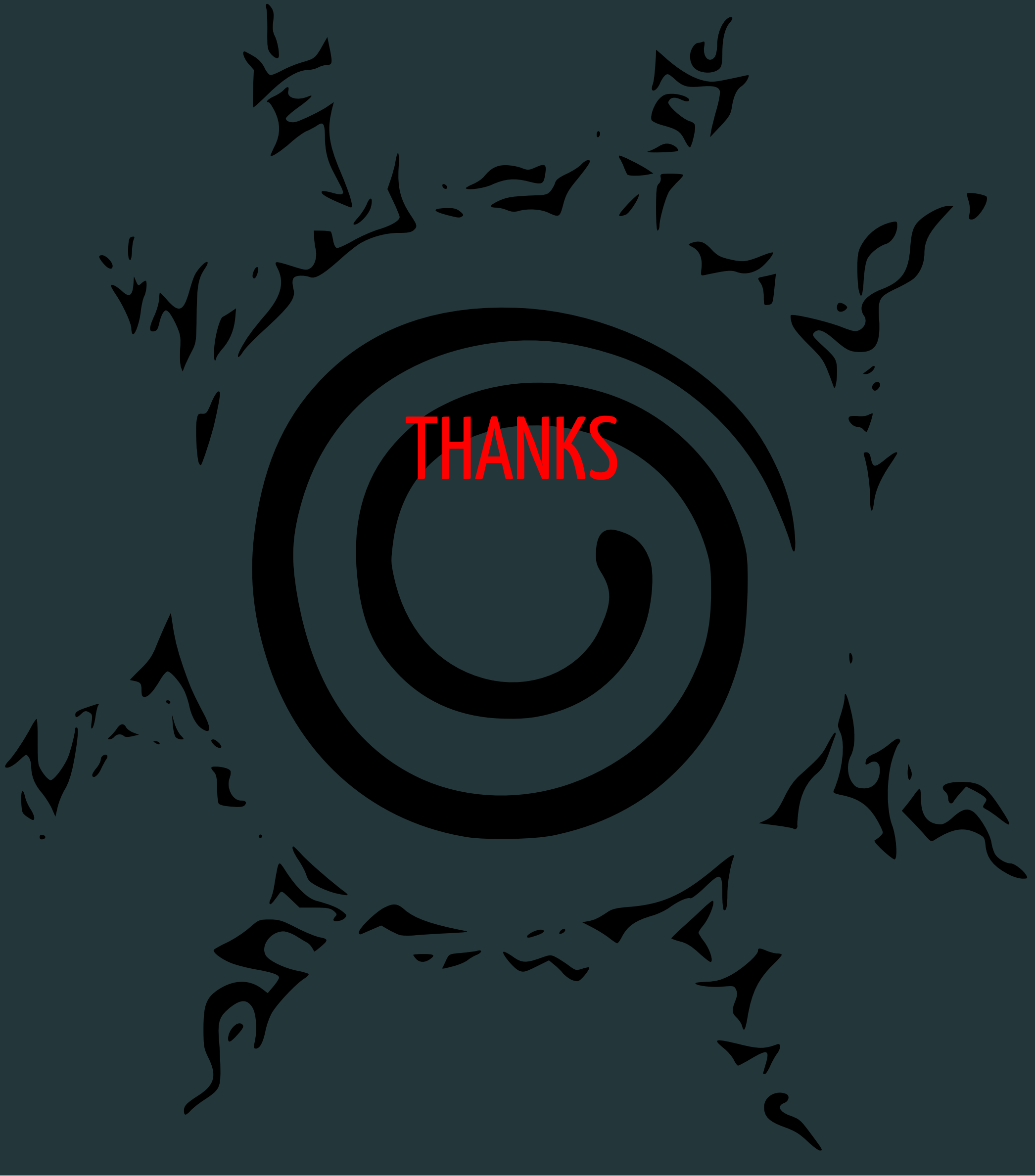
Packages

Packages add functionality that is not present in base R.

Strength `R` lies in packages



Courtesy Modern Dive



THANKS