Exploratory Data Analysis I

# Exploratory Data Analysis (EDA) I

## R with R Studio and/or R Studio.cloud

### Course Contents

1. 2022-12-07: Introduction: About the course　[lead by TK] - An introduction to open and public data, and data science
2. **2022-12-14: Exploratory Data Analysis (EDA) 1 [lead by hs]**  
   **- R Basics with RStudio and/or RStudio.cloud; Toy Data**
3. 2022-12-21: Exploratory Data Analysis (EDA) 2 [lead by hs]  
   - R Markdown; Introduction to tidyverse I; Public Data, WDI
4. 2023-01-11: Exploratory Data Analysis (EDA) 3 [lead by hs]  
   - Introduction to tidyverseII; WDI, WIR, etc
5. 2023-01-18: Exploratory Data Analysis (EDA) 4 [lead by hs]  
   - Introduction to tidyverse III; WDI, WIR, etc
6. 2023-01-25: Exploratory Data Analysis (EDA) 5 [lead by hs]  
   - Introduction to tidyverse III; WDI, WIR, etc
7. 2023-02-01: Introduction to PPDAC (Problem-Plan-Data-Analysis-Conclusion) Cycle: [lead by TK]
8. 2023-02-08: Model building I [lead by TK] -Collecting and visualizing data and Introduction to WDI  
   (World Development Indicators by World Bank)
9. 2023-02-15: Model building II [lead by TK] -Analyzing data and communications
10. 2023-02-22: Project Presentation

### Learning Resources

#### Textbooks and References

* “R for Data Science” by Hadley Wickham and Garrett Grolemund:
  + Free Online Book: <https://r4ds.had.co.nz>
* Visit bookdown site: <https://bookdown.org>
  + Many more on the [archive page](https://bookdown.org/home/archive/).

### Interactive Exercises

* Posit Primers:<https://posit.cloud/learn/primers>:
  + The Basics, Work with Data, Visualize Data, Tidy Your Data, Report Reproducibly
* {swirl} Learn R, in R: <https://swirlstats.com>
  + Designed and developed by a team at Johns Hopkins University for coursera courses

### Posit Primers created by learnr

* [learnr Interactive Tutorials for R](https://rstudio.github.io/learnr/index.html)

#### Posit Primers <https://posit.cloud/learn/primers>

1. The Basics – [r4ds: Explore, I](https://r4ds.had.co.nz/explore-intro.html#explore-intro)

* [Visualization Basics](https://rstudio.cloud/learn/primers/1.1)
* [Programming Basics](https://rstudio.cloud/learn/primers/1.2)

1. Work with Data – [r4ds: Wrangle, I](https://r4ds.had.co.nz/wrangle-intro.html#wrangle-intro)

* Working with Tibbles
* Isolating Data with dplyr
* Deriving Information with dplyr

1. Visualize Data – [r4ds: Explore, II](https://r4ds.had.co.nz/explore-intro.html#explore-intro)
2. Tidy Your Data – [r4ds: Wrangle, II](https://r4ds.had.co.nz/wrangle-intro.html#wrangle-intro)
3. Iterate – [r4ds: Program](https://r4ds.had.co.nz/program-intro.html#program-intro)
4. Write Functions – [r4ds: Program](https://r4ds.had.co.nz/program-intro.html#program-intro)

### Data Science and EDA

#### Wikipedia <https://en.wikipedia.org/wiki/Data_science>

An inter-disciplinary field that uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data.

* Create Insights
* Impact Decision Making
* Maintain & Improve Overtime

### What is R?

#### R (programming language), [Wikipedia](https://en.wikipedia.org/wiki/R_(programming_language))

* **R is a programming language** and **free software** environment for **statistical computing and graphics** supported by the R Foundation for Statistical Computing.
* The R language is widely used among statisticians and data miners for developing statistical software and data analysis.
* A **GNU package**, the official R software environment is written primarily in C, Fortran, and R itself (thus, it is partially self-hosting) and is freely available under the GNU General Public License.

#### History of R and more

“R Programming for Data Science” by Roger Peng

* [Chapter 2. History and Overview of R](https://bookdown.org/rdpeng/rprogdatascience/history-and-overview-of-r.html)
* [Overview and History of R: Youtube video](https://www.youtube.com/watch?v=STihTnVSZnI&feature=youtu.be)

### Why R? – Responses by Hadley Wickham

#### [r4ds](https://r4ds.had.co.nz/introduction.html" \l "python-julia-and-friends): R is a great place to start your data science journey because

* R is an environment designed from the ground up to support data science.
* R is not just a programming language, but it is also an interactive environment for doing data science.
* To support interaction, R is a much more flexible language than many of its peers.

#### Why R today?

When you talk about choosing programming languages, I always say you shouldn’t pick them based on technical merits, but rather pick them based on the community. And I think the R community is like really, really strong, vibrant, free, welcoming, and embraces a wide range of domains. So, if there are like people like you using R, then your life is going to be much easier. That’s the first reason.

**Interview**: [“Advice to Young (and Old) Programmers, H. Wickham”](https://www.r-bloggers.com/2018/08/advice-to-young-and-old-programmers-a-conversation-with-hadley-wickham/)

### What is RStudio? <https://posit.com>

RStudio is an integrated development environment, or IDE, for R programming.

#### R Studio (Wikipedia)

RStudio is an integrated development environment (IDE) for R, a programming language for statistical computing and graphics. It is available in two formats: RStudio Desktop is a regular desktop application while RStudio Server runs on a remote server and allows accessing RStudio using a web browser.

### Installation of R and R Studio

#### R Installation

To download R, go to CRAN, the comprehensive R archive network. CRAN is composed of a set of mirror servers distributed around the world and is used to distribute R and R packages. Don’t try and pick a mirror that’s close to you: instead use the cloud mirror, <https://cloud.r-project.org>, which automatically figures it out for you.

A new major version of R comes out once a year, and there are 2-3 minor releases each year. It’s a good idea to update regularly.

#### R Studio Installation

Download and install it from <http://www.rstudio.com/download>.

RStudio is updated a couple of times a year. When a new version is available, RStudio will let you know.

### R Studio

#### The First Step

1. Start R Studio Application
2. Top Menu: File > New Project > New Directory > New Project > *Directory name or Browse the directory and choose the parent directory you want to create the directory*

#### When You Start the Project

1. Go to the directory you created
2. Double click \_‘Directory Name’.Rproj

Or,

1. Start R Studio
2. File > Open Project (or choose from Recent Project)

*In this way the working directory of the session is set to the project directory and R can search releted files without difficulty* (getwd(), setwd())

### Posit Cloud

RStudio Cloud is a lightweight, cloud-based solution that allows anyone to do, share, teach and learn data science online.

#### Cloud Free

* Up to 15 projects total
* 1 shared space (5 members and 10 projects max)
* 15 project hours per month
* Up to 1 GB RAM per project
* Up to 1 CPU per project
* Up to 1 hour background execution time

#### How to Start Posit Cloud

1. Go to <https://posit.cloud/>
2. Sign Up: *top right*

* Email address or Google account

1. New Project: *Project Name*
2. R Console

## Let’s Get Started

Start RStudio and create a project, or login to Posit Cloud and create a project.

### The First Examples

Input the following codes into Console in the left bottom pane.

* The first two:

head(cars)

## speed dist  
## 1 4 2  
## 2 4 10  
## 3 7 4  
## 4 7 22  
## 5 8 16  
## 6 9 10

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 17 ...

* Two more:

summary(cars)

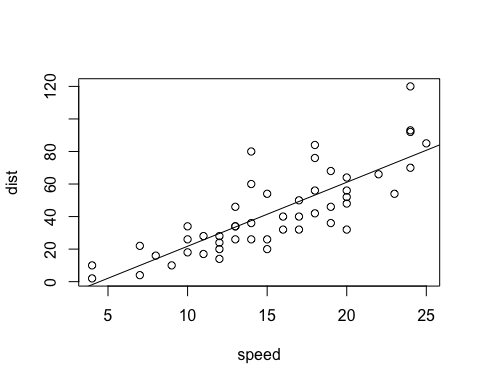
## speed dist   
## Min. : 4.0 Min. : 2.00   
## 1st Qu.:12.0 1st Qu.: 26.00   
## Median :15.0 Median : 36.00   
## Mean :15.4 Mean : 42.98   
## 3rd Qu.:19.0 3rd Qu.: 56.00   
## Max. :25.0 Max. :120.00

plot(cars)



* And three more:

plot(cars) # cars: Speed and Stopping Distances of Cars  
abline(lm(cars$dist~cars$speed))



lm(cars$dist~cars$speed)

##   
## Call:  
## lm(formula = cars$dist ~ cars$speed)  
##   
## Coefficients:  
## (Intercept) cars$speed   
## -17.579 3.932

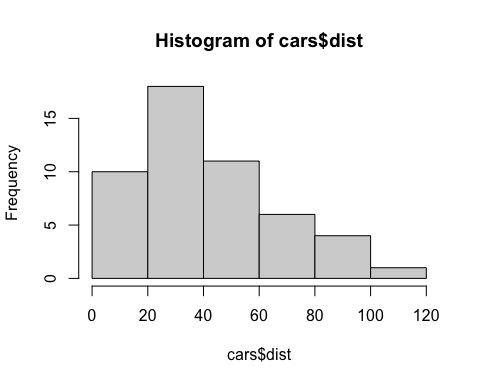
summary(lm(cars$dist~cars$speed))

##   
## Call:  
## lm(formula = cars$dist ~ cars$speed)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -29.069 -9.525 -2.272 9.215 43.201   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -17.5791 6.7584 -2.601 0.0123 \*   
## cars$speed 3.9324 0.4155 9.464 1.49e-12 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 15.38 on 48 degrees of freedom  
## Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438   
## F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

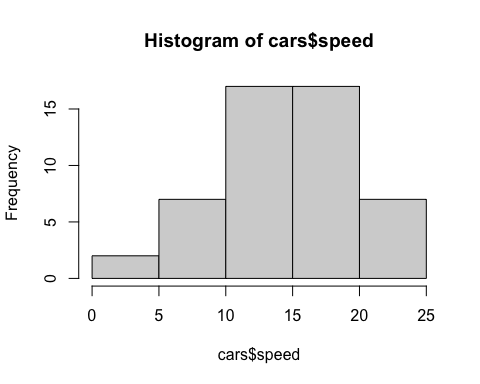
#### Brief Explanation

* head(cars): The first 6 rows of the pre-installed data cars.
* str(cars): The data structure of the pre-installed data cars.
* summary(cars): The summary of the pre-installed data cars.
* plot(cars): A scatter plot of the pre-installed data cars.
  + plot(cars$dist~cars$speed)
  + cars$dist, cars$[[2]], cars[,2] are same
* abline(lm(cars$dist~cars$speed)): Add a regression line of a linear model
* lm(cars$dist~cars$speed): The equation of the regression line
* summary(lm(cars$dist~cars$speed): The summary of the linear regression model

hist(cars$dist)



hist(cars$speed)



#### View and help

* View(cars)
* ?cars: same as help(cars)
* ??cars: same as `help.search(“cars”)

#### datasets

* ?datasets
* library(help = "datasets")
* data() shows all data already attached and available.

### Practicum

Pick a data in the datasets package and try

* head()
* str()
* summary()

and some more.

#### iris

head(iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa

str(iris)

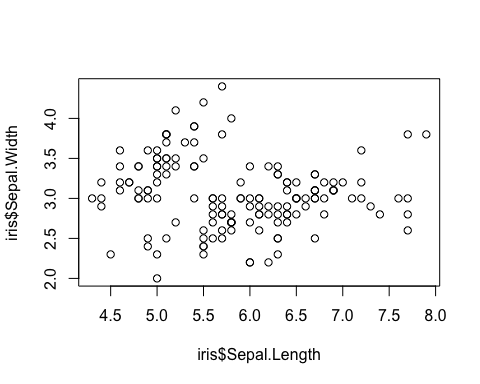
## 'data.frame': 150 obs. of 5 variables:  
## $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
## $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
## $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
## $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
## $ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

summary(iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width   
## Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100   
## 1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300   
## Median :5.800 Median :3.000 Median :4.350 Median :1.300   
## Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199   
## 3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800   
## Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500   
## Species   
## setosa :50   
## versicolor:50   
## virginica :50   
##   
##   
##

Can you plot?

plot(iris$Sepal.Length, iris$Sepal.Width)



## tidyverse Packages

### Brief Introduction to R on RStudio

#### Four Panes and Tabs

1. Top Left: Source Editor
2. Top Right: Environment, History, etc.
3. Bottom Left: Console, Terminal, Render, Background Jobs
4. Bottom Right: Files, Plots, Packages, Help, Viewer, Presentation

#### Set up

* Highly recommend to set the language to be “English”.
* Create “data” directory.

Sys.setenv(LANG = "en")  
dir.create("data")

#### Three Ways to Run Codes

1. Console - Bottom Left Pane
2. R Script - pull down menu under File
3. R Notebook, R Markdown - pull down menu under File

### Second Way: R Script

#### Examples: R Scripts in Moodle

* basics.R
* coronavirus.R

1. Copy a script in Moodle: *{file name}.R*
2. In RStudio (create Project in RStudio) choose File > New File > R Script and paste it.
3. Choose File > Save with a name; e.g. *{file names}* (.R will be added automatically)

To run a code: at the cursor press *Ctrl+Shift+Enter* (Win) or *Cmd+Shift+Enter* (Mac).

### Packages

R packages are extensions to the R statistical programming language. R packages contain code, data, and documentation in a standardised collection format that can be installed by users of R, typically via a centralised software repository such as CRAN (the Comprehensive R Archive Network).

#### Installation and attachement

You can install packages by “Install Packages…” under “Tool” in the top menu.

* install.packages("tidyverse")
* install.packages("rmarkdown")

### Third Way: R Notebook

Choose R Notebook from the pull down File menu in the top bar.

### Edit YAML

**Default\* is as follows**

---  
title: "R Notebook"  
output: html\_notebook  
---

**Template**

---  
title: "Title of R Notebook"  
author: "ID and Your Name"  
date: "2022-12-17"   
output:   
 html\_notebook:  
# number\_sections: yes  
# toc: true  
# toc\_float: true  
---

* Don’t change the format. Indention matters.
* The statement after # is ignored.
* Date is automatically inserted when you compile the file.
* You can replace “2022-12-17” by “2022-12-14” or in any date format surrounded by double quotation marks.
* Section numbers: - default is number\_sections: no.
* Table of contents, toc: true - default is toc: false.
* Floating table of contents in HTML output, toc\_float: true - default is toc\_float: false

### Create a Code Chunk to Attach Packages

Insert Chunk in Code pull down menu in the top bar, or use the C button on top. You can use shortcut keys listed under Tools in the top bar.

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.0 ✔ purrr 0.3.5   
## ✔ tibble 3.1.8 ✔ dplyr 1.0.10  
## ✔ tidyr 1.2.1 ✔ stringr 1.4.1   
## ✔ readr 2.1.3 ✔ forcats 0.5.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

## First Example

### Importing data

Let us assign the iris data in the pre-installed package datasets to df\_iris. You can give any name starting from an alphabet, though there are some rules.

df\_iris <- datasets::iris  
class(df\_iris)

## [1] "data.frame"

The class of data iris is data.frame, the basic data class of R. You can assign the same data as a tibble, the data class of tidyverse as follows.

tbl\_iris <- as\_tibble(datasets::iris)  
class(tbl\_iris)

## [1] "tbl\_df" "tbl" "data.frame"

* df\_iris <- iris can replace df\_iris <- datasets::iris because the package datasets is installed and attached as default. Since you may have other data called iris included in a different package or you may have changed iris before, it is safer to specify the name of the package with the name of the data.
* Within R Notebook or in Console, you may get different output, and tf\_iris and tbl\_iris behave differently. It is because of the default settings of R Markdown.

### Look at the data

#### Several ways to view the data.

The View command open up a window to show the contents of the data and you can use the filter as well.

View(df\_iris)

The following simple command also shows the data.

df\_iris

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa  
## 7 4.6 3.4 1.4 0.3 setosa  
## 8 5.0 3.4 1.5 0.2 setosa  
## 9 4.4 2.9 1.4 0.2 setosa  
## 10 4.9 3.1 1.5 0.1 setosa  
## 11 5.4 3.7 1.5 0.2 setosa  
## 12 4.8 3.4 1.6 0.2 setosa  
## 13 4.8 3.0 1.4 0.1 setosa  
## 14 4.3 3.0 1.1 0.1 setosa  
## 15 5.8 4.0 1.2 0.2 setosa  
## 16 5.7 4.4 1.5 0.4 setosa  
## 17 5.4 3.9 1.3 0.4 setosa  
## 18 5.1 3.5 1.4 0.3 setosa  
## 19 5.7 3.8 1.7 0.3 setosa  
## 20 5.1 3.8 1.5 0.3 setosa  
## 21 5.4 3.4 1.7 0.2 setosa  
## 22 5.1 3.7 1.5 0.4 setosa  
## 23 4.6 3.6 1.0 0.2 setosa  
## 24 5.1 3.3 1.7 0.5 setosa  
## 25 4.8 3.4 1.9 0.2 setosa  
## 26 5.0 3.0 1.6 0.2 setosa  
## 27 5.0 3.4 1.6 0.4 setosa  
## 28 5.2 3.5 1.5 0.2 setosa  
## 29 5.2 3.4 1.4 0.2 setosa  
## 30 4.7 3.2 1.6 0.2 setosa  
## 31 4.8 3.1 1.6 0.2 setosa  
## 32 5.4 3.4 1.5 0.4 setosa  
## 33 5.2 4.1 1.5 0.1 setosa  
## 34 5.5 4.2 1.4 0.2 setosa  
## 35 4.9 3.1 1.5 0.2 setosa  
## 36 5.0 3.2 1.2 0.2 setosa  
## 37 5.5 3.5 1.3 0.2 setosa  
## 38 4.9 3.6 1.4 0.1 setosa  
## 39 4.4 3.0 1.3 0.2 setosa  
## 40 5.1 3.4 1.5 0.2 setosa  
## 41 5.0 3.5 1.3 0.3 setosa  
## 42 4.5 2.3 1.3 0.3 setosa  
## 43 4.4 3.2 1.3 0.2 setosa  
## 44 5.0 3.5 1.6 0.6 setosa  
## 45 5.1 3.8 1.9 0.4 setosa  
## 46 4.8 3.0 1.4 0.3 setosa  
## 47 5.1 3.8 1.6 0.2 setosa  
## 48 4.6 3.2 1.4 0.2 setosa  
## 49 5.3 3.7 1.5 0.2 setosa  
## 50 5.0 3.3 1.4 0.2 setosa  
## 51 7.0 3.2 4.7 1.4 versicolor  
## 52 6.4 3.2 4.5 1.5 versicolor  
## 53 6.9 3.1 4.9 1.5 versicolor  
## 54 5.5 2.3 4.0 1.3 versicolor  
## 55 6.5 2.8 4.6 1.5 versicolor  
## 56 5.7 2.8 4.5 1.3 versicolor  
## 57 6.3 3.3 4.7 1.6 versicolor  
## 58 4.9 2.4 3.3 1.0 versicolor  
## 59 6.6 2.9 4.6 1.3 versicolor  
## 60 5.2 2.7 3.9 1.4 versicolor  
## 61 5.0 2.0 3.5 1.0 versicolor  
## 62 5.9 3.0 4.2 1.5 versicolor  
## 63 6.0 2.2 4.0 1.0 versicolor  
## 64 6.1 2.9 4.7 1.4 versicolor  
## 65 5.6 2.9 3.6 1.3 versicolor  
## 66 6.7 3.1 4.4 1.4 versicolor  
## 67 5.6 3.0 4.5 1.5 versicolor  
## 68 5.8 2.7 4.1 1.0 versicolor  
## 69 6.2 2.2 4.5 1.5 versicolor  
## 70 5.6 2.5 3.9 1.1 versicolor  
## 71 5.9 3.2 4.8 1.8 versicolor  
## 72 6.1 2.8 4.0 1.3 versicolor  
## 73 6.3 2.5 4.9 1.5 versicolor  
## 74 6.1 2.8 4.7 1.2 versicolor  
## 75 6.4 2.9 4.3 1.3 versicolor  
## 76 6.6 3.0 4.4 1.4 versicolor  
## 77 6.8 2.8 4.8 1.4 versicolor  
## 78 6.7 3.0 5.0 1.7 versicolor  
## 79 6.0 2.9 4.5 1.5 versicolor  
## 80 5.7 2.6 3.5 1.0 versicolor  
## 81 5.5 2.4 3.8 1.1 versicolor  
## 82 5.5 2.4 3.7 1.0 versicolor  
## 83 5.8 2.7 3.9 1.2 versicolor  
## 84 6.0 2.7 5.1 1.6 versicolor  
## 85 5.4 3.0 4.5 1.5 versicolor  
## 86 6.0 3.4 4.5 1.6 versicolor  
## 87 6.7 3.1 4.7 1.5 versicolor  
## 88 6.3 2.3 4.4 1.3 versicolor  
## 89 5.6 3.0 4.1 1.3 versicolor  
## 90 5.5 2.5 4.0 1.3 versicolor  
## 91 5.5 2.6 4.4 1.2 versicolor  
## 92 6.1 3.0 4.6 1.4 versicolor  
## 93 5.8 2.6 4.0 1.2 versicolor  
## 94 5.0 2.3 3.3 1.0 versicolor  
## 95 5.6 2.7 4.2 1.3 versicolor  
## 96 5.7 3.0 4.2 1.2 versicolor  
## 97 5.7 2.9 4.2 1.3 versicolor  
## 98 6.2 2.9 4.3 1.3 versicolor  
## 99 5.1 2.5 3.0 1.1 versicolor  
## 100 5.7 2.8 4.1 1.3 versicolor  
## 101 6.3 3.3 6.0 2.5 virginica  
## 102 5.8 2.7 5.1 1.9 virginica  
## 103 7.1 3.0 5.9 2.1 virginica  
## 104 6.3 2.9 5.6 1.8 virginica  
## 105 6.5 3.0 5.8 2.2 virginica  
## 106 7.6 3.0 6.6 2.1 virginica  
## 107 4.9 2.5 4.5 1.7 virginica  
## 108 7.3 2.9 6.3 1.8 virginica  
## 109 6.7 2.5 5.8 1.8 virginica  
## 110 7.2 3.6 6.1 2.5 virginica  
## 111 6.5 3.2 5.1 2.0 virginica  
## 112 6.4 2.7 5.3 1.9 virginica  
## 113 6.8 3.0 5.5 2.1 virginica  
## 114 5.7 2.5 5.0 2.0 virginica  
## 115 5.8 2.8 5.1 2.4 virginica  
## 116 6.4 3.2 5.3 2.3 virginica  
## 117 6.5 3.0 5.5 1.8 virginica  
## 118 7.7 3.8 6.7 2.2 virginica  
## 119 7.7 2.6 6.9 2.3 virginica  
## 120 6.0 2.2 5.0 1.5 virginica  
## 121 6.9 3.2 5.7 2.3 virginica  
## 122 5.6 2.8 4.9 2.0 virginica  
## 123 7.7 2.8 6.7 2.0 virginica  
## 124 6.3 2.7 4.9 1.8 virginica  
## 125 6.7 3.3 5.7 2.1 virginica  
## 126 7.2 3.2 6.0 1.8 virginica  
## 127 6.2 2.8 4.8 1.8 virginica  
## 128 6.1 3.0 4.9 1.8 virginica  
## 129 6.4 2.8 5.6 2.1 virginica  
## 130 7.2 3.0 5.8 1.6 virginica  
## 131 7.4 2.8 6.1 1.9 virginica  
## 132 7.9 3.8 6.4 2.0 virginica  
## 133 6.4 2.8 5.6 2.2 virginica  
## 134 6.3 2.8 5.1 1.5 virginica  
## 135 6.1 2.6 5.6 1.4 virginica  
## 136 7.7 3.0 6.1 2.3 virginica  
## 137 6.3 3.4 5.6 2.4 virginica  
## 138 6.4 3.1 5.5 1.8 virginica  
## 139 6.0 3.0 4.8 1.8 virginica  
## 140 6.9 3.1 5.4 2.1 virginica  
## 141 6.7 3.1 5.6 2.4 virginica  
## 142 6.9 3.1 5.1 2.3 virginica  
## 143 5.8 2.7 5.1 1.9 virginica  
## 144 6.8 3.2 5.9 2.3 virginica  
## 145 6.7 3.3 5.7 2.5 virginica  
## 146 6.7 3.0 5.2 2.3 virginica  
## 147 6.3 2.5 5.0 1.9 virginica  
## 148 6.5 3.0 5.2 2.0 virginica  
## 149 6.2 3.4 5.4 2.3 virginica  
## 150 5.9 3.0 5.1 1.8 virginica

The output within R Notebook is a tibble style. Try the same command in Console.

slice(df\_iris, 1:10)

## Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
## 1 5.1 3.5 1.4 0.2 setosa  
## 2 4.9 3.0 1.4 0.2 setosa  
## 3 4.7 3.2 1.3 0.2 setosa  
## 4 4.6 3.1 1.5 0.2 setosa  
## 5 5.0 3.6 1.4 0.2 setosa  
## 6 5.4 3.9 1.7 0.4 setosa  
## 7 4.6 3.4 1.4 0.3 setosa  
## 8 5.0 3.4 1.5 0.2 setosa  
## 9 4.4 2.9 1.4 0.2 setosa  
## 10 4.9 3.1 1.5 0.1 setosa

1:10

## [1] 1 2 3 4 5 6 7 8 9 10

## `

#### Data Structure

Let us look at the structure of the data. You can try str(df\_iris) on Console or by adding a code chunk in R Notebook introducing later.

glimpse(df\_iris)

## Rows: 150  
## Columns: 5  
## $ Sepal.Length <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.…  
## $ Sepal.Width <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7, 3.…  
## $ Petal.Length <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.…  
## $ Petal.Width <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.…  
## $ Species <fct> setosa, setosa, setosa, setosa, setosa, setosa, setosa, s…

There are six types of data in R; Double, Integer, Character, Logical, Raw, Complex.

The names after $ are column names. If you call df\_iris$Species, you have the Species column. Species is in the 5th collumn, typeof(df\_iris[[5]]) does the same as the next.

df\_iris[2,4] =0.2 is the fourth entry of Sepal.Width.

typeof(df\_iris$Species)

## [1] "integer"

class(df\_iris$Species)

## [1] "factor"

For factors = fct see [the R Document](https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/factor) or an explanation in [Factor in R: Categorical Variable & Continuous Variables](https://www.guru99.com/r-factor-categorical-continuous.html).

typeof(df\_iris$Sepal.Length)

## [1] "double"

class(df\_iris$Sepal.Length)

## [1] "numeric"

**Q1.** What are the differences ofdf\_iris, slice(df\_iris, 1:10) and glimpse(df\_iris) above?

**Q2.** What are the differences ofdf\_iris, slice(df\_iris, 1:10) and glimpse(df\_iris) in the console?

#### Summary of the Data

The following is very convenient to get the summary information of a data.

summary(df\_iris)

## Sepal.Length Sepal.Width Petal.Length Petal.Width   
## Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100   
## 1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300   
## Median :5.800 Median :3.000 Median :4.350 Median :1.300   
## Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199   
## 3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800   
## Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500   
## Species   
## setosa :50   
## versicolor:50   
## virginica :50   
##   
##   
##

Minimum, 1st Quadrant (25%), Median, Mean, 3rd Quadrant (75%), Maximum, and the count of each factor.

### Visualizing Data

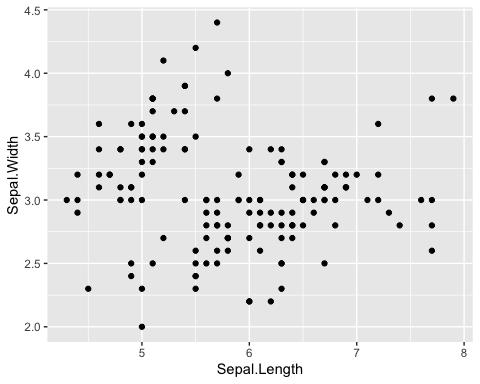
#### Scatter Plot

We use ggplot to draw graphs. The scatter plot is a projection of data with two variables and .

ggplot(data = <data>, aes(x = <column name for x>, y = <column name for y>)) +  
 geom\_point()

ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width)) +  
 geom\_point()

ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width)) +  
 geom\_point()

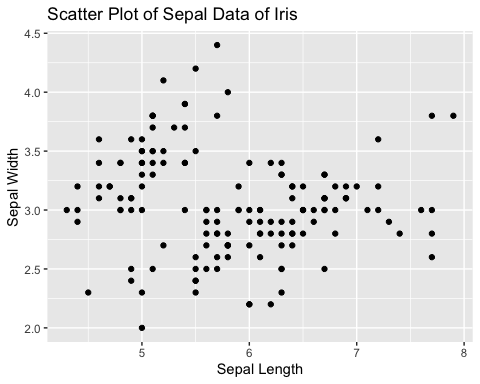


#### Scatter Plot with [Labels](https://ggplot2.tidyverse.org/reference/labs.html)

Add title and labels adding labs().

ggplot(data = <data>, aes(x = <column name for x>, y = <column name for y>)) +  
 geom\_point() +  
 labs(title = "Title", x = "Label for x", y = "Label for y")

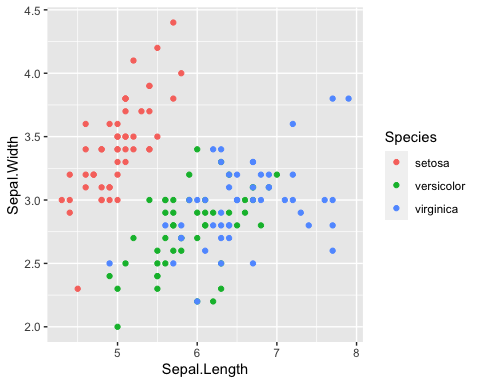
ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width)) +  
 geom\_point() +   
 labs(title = "Scatter Plot of Sepal Data of Iris", x = "Sepal Length", y = "Sepal Width")



#### Scatter Plot with [Colors](https://ggplot2.tidyverse.org/reference/aes_colour_fill_alpha.html)

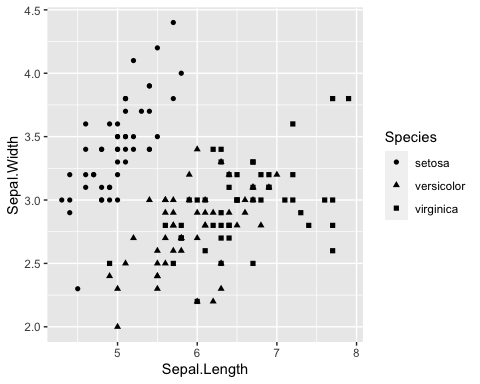
Add different colors automatically to each species. Can you see each group?

ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +  
 geom\_point()



#### Scatter Plot with Shapes

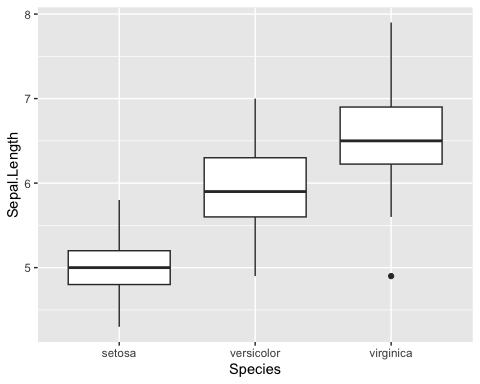
ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width, shape = Species)) +  
 geom\_point()



#### [Boxplot](https://ggplot2.tidyverse.org/reference/geom_boxplot.html)

The boxplot compactly displays the distribution of a continuous variable.

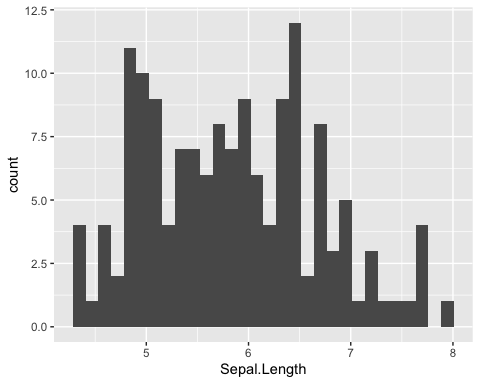
ggplot(data = df\_iris, aes(x = Species, y = Sepal.Length)) +  
 geom\_boxplot()



#### [Histogram](https://ggplot2.tidyverse.org/reference/geom_histogram.html)

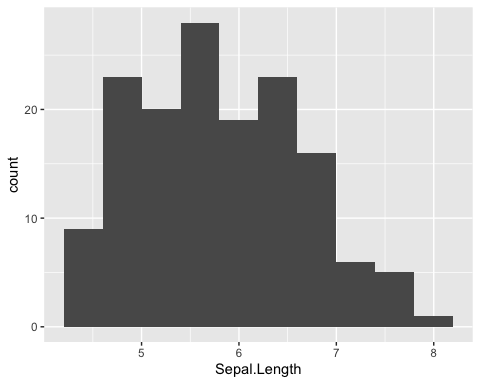
Visualize the distribution of a single continuous variable by dividing the x axis into bins and counting the number of observations in each bin. Histograms (geom\_histogram()) display the counts with bars.

ggplot(data = df\_iris, aes(x = Sepal.Length)) +  
 geom\_histogram()



Change the number of bins by bins = <number>.

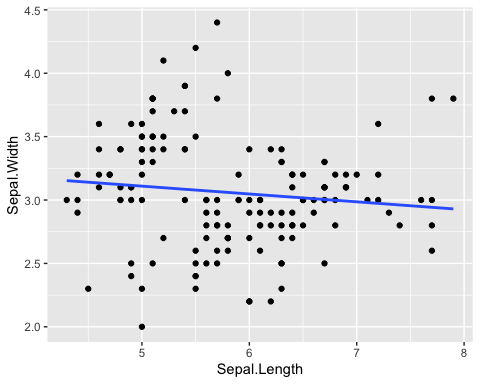
ggplot(data = df\_iris, aes(x = Sepal.Length)) +  
 geom\_histogram(bins = 10)



### Data Modeling

Professor Kaizoji will cover the mathematical models and hypothesis testings.

ggplot(data = df\_iris, aes(x = Sepal.Length, y = Sepal.Width)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE)



## Comments on Week 2

#### Helpful Resources

* Cheat Sheet in RStudio: <https://www.rstudio.com/resources/cheatsheets/>
  + [RStudio IED](https://raw.githubusercontent.com/rstudio/cheatsheets/main/rstudio-ide.pdf)
  + [Base R Cheat Sheet](https://github.com/rstudio/cheatsheets/raw/main/base-r.pdf)
* ‘Quick R’ by DataCamp: <https://www.statmethods.net/management>
* [An Introduction to R](https://cran.rstudio.com)

#### Practicum

* Posit Primers: The Basics: <https://posit.cloud/learn/primers/1>
  + Complete Visualization Basics and Programming Basics

#### Assignments - See Moodle

1. Assignment Week 2-1: Introduction Plus Forum

* Due: Tuesday, 20 December 2022, 11:59 PM

1. Assignment Week 2-2: Quiz 1 on R Basics

* Due: Tuesday, 20 December 2022, 11:59 PM

## Swirl: An interactive learning environment for R and statistics

* {swirl} website: <https://swirlstats.com>
* JHU Data Science in coursera uses swirl for exercises.

### Swirl Courses

1. R Programming: The basics of programming in R
2. Regression Models: The basics of regression modeling in R
3. Statistical Inference: The basics of statistical inference in R
4. Exploratory Data Analysis: The basics of exploring data in R

You can install other swirl courses as well

* [Swirl Courses Organized by Title](http://swirlstats.com/scn/title.html)
* [Swirl Courses Organized by Author’s Name](http://swirlstats.com/scn/surname.html)
* [Github: swirl courses](https://github.com/swirldev/swirl_courses#swirl-courses)
  + install\_course("Course Name Here")

### Install and Start Swirl Courses

#### Three Steps to Start Swirl

install.packages("swirl") # Only the first time.  
library(swirl) # Everytime you start swirl  
swirl() # Everytime you start or resume swirl

#### R Programming: The basics of programming in R

1: Basic Building Blocks 2: Workspace and Files 3: Sequences of Numbers   
 4: Vectors 5: Missing Values 6: Subsetting Vectors   
 7: Matrices and Data Frames 8: Logic 9: Functions   
10: lapply and sapply 11: vapply and tapply 12: Looking at Data   
13: Simulation 14: Dates and Times 15: Base Graphics

#### Recommended Sections in Order

1, 3, 4, 5, 6, 7, 12, 15, 14, 8, 9, 10, 11, 13, 2

* Section 2 discusses the directories and file systems of a computer
* Sections 9, 10, 11 are for programming

#### Controling a swirl Session

* … <– That’s your cue to press Enter to continue
* You can exit swirl and return to the R prompt (>) at any time by pressing the Esc key.
* If you are already at the prompt, type bye() to exit and save your progress. When you exit properly, you’ll see a short message letting you know you’ve done so.

When you are at the R prompt (>):

1. Typing skip() allows you to skip the current question.
2. Typing play() lets you experiment with R on your own; swirl will ignore what you do…
3. UNTIL you type nxt() which will regain swirl’s attention.
4. Typing bye() causes swirl to exit. Your progress will be saved.
5. Typing main() returns you to swirl’s main menu.
6. Typing info() displays these options again.

#### Final Remark

You will encounter the message like ‘Would you like to receive credit for completing this course on Coursera.org?’ at the end of each course. This is for coursera courses. Select ‘NO’.

## More on R Script: Examples

### R Scripts in Moodle

* basics.R
* coronavirus.R

1. Copy a script in Moodle: *{file name}.R*
2. In RStudio (Workspace in RStudio.cloud, Project in RStudio) choose File > New File > R Script and paste it.
3. Choose File > Save with a name; e.g. *{file names}* (.R will be added automatically)

### basics.R

The script with the outputs.

#################  
#  
# basics.R  
#  
################  
# 'Quick R' by DataCamp may be a handy reference:   
# https://www.statmethods.net/management/index.html  
# Cheat Sheet at RStudio: https://www.rstudio.com/resources/cheatsheets/  
# Base R Cheat Sheet: https://github.com/rstudio/cheatsheets/raw/main/base-r.pdf  
# To execute the line: Control + Enter (Window and Linux), Command + Enter (Mac)  
## try your experiments on the console  
  
## calculator  
  
3 + 7  
  
### +, -, \*, /, ^ (or \*\*), %%, %/%  
  
3 + 10 / 2  
  
3^2  
  
2^3  
  
2\*2\*2  
  
### assignment: <-, (=, ->, assign())   
  
x <- 5  
  
x   
  
#### object\_name <- value, '<-' shortcut: Alt (option) + '-' (hyphen or minus)   
#### Object names must start with a letter and can only contain letter, numbers, \_ and .  
  
this\_is\_a\_long\_name <- 5^3  
  
this\_is\_a\_long\_name  
  
char\_name <- "What is your name?"  
  
char\_name  
  
#### Use 'tab completion' and 'up arrow'  
  
### ls(): list of all assignments  
  
ls()  
ls.str()  
  
#### check Environment in the upper right pane  
  
### (atomic) vectors  
  
5:10  
  
a <- seq(5,10)  
  
a  
  
b <- 5:10  
  
identical(a,b)  
  
seq(5,10,2) # same as seq(from = 5, to = 10, by = 2)  
  
c1 <- seq(0,100, by = 10)  
  
c2 <- seq(0,100, length.out = 10)  
  
c1  
  
c2  
  
length(c1)  
  
#### ? seq ? length ? identical  
  
(die <- 1:6)  
  
zero\_one <- c(0,1) # same as 0:1  
  
die + zero\_one # c(1,2,3,4,5,6) + c(0,1). re-use  
  
d1 <- rep(1:3,2) # repeat  
  
  
d1  
  
die == d1  
  
d2 <- as.character(die == d1)  
  
d2  
  
d3 <- as.numeric(die == d1)  
  
d3  
  
### class() for class and typeof() for mode  
### class of vectors: numeric, charcters, logical  
### types of vectors: doubles, integers, characters, logicals (complex and raw)  
  
typeof(d1); class(d1)  
  
typeof(d2); class(d2)  
  
typeof(d3); class(d3)  
  
sqrt(2)  
  
sqrt(2)^2  
  
sqrt(2)^2 - 2  
  
typeof(sqrt(2))  
  
typeof(2)  
  
typeof(2L)  
  
5 == c(5)  
  
length(5)  
  
### Subsetting  
  
(A\_Z <- LETTERS)  
  
A\_F <- A\_Z[1:6]  
  
A\_F  
  
A\_F[3]  
  
A\_F[c(3,5)]  
  
large <- die > 3  
  
large  
  
even <- die %in% c(2,4,6)  
  
even  
  
A\_F[large]  
  
A\_F[even]  
  
A\_F[die < 4]  
  
### Compare df with df1 <- data.frame(number = die, alphabet = A\_F)  
df <- data.frame(number = die, alphabet = A\_F, stringsAsFactors = FALSE)  
  
df  
  
df$number  
  
df$alphabet  
  
df[3,2]  
  
df[4,1]  
  
df[1]  
  
class(df[1])  
  
class(df[[1]])  
  
identical(df[[1]], die)  
  
identical(df[1],die)  
  
####################  
# The First Example  
####################  
  
plot(cars)  
  
# Help  
  
? cars  
  
# cars is in the 'datasets' package  
  
data()  
  
# help(cars) does the same as ? cars  
# You can use Help tab in the right bottom pane  
  
help(plot)  
? par  
  
head(cars)  
  
str(cars)  
  
summary(cars)  
  
x <- cars$speed  
y <- cars$dist  
  
min(x)  
mean(x)  
quantile(x)  
  
plot(cars)  
  
abline(lm(cars$dist ~ cars$speed))  
  
summary(lm(cars$dist ~ cars$speed))  
  
boxplot(cars)  
  
hist(cars$speed)  
hist(cars$dist)  
hist(cars$dist, breaks = seq(0,120, 10))

### coronavirus.R

The script and its outputs. **coronavirus.csv** is very large

# https://coronavirus.jhu.edu/map.html  
# JHU Covid-19 global time series data  
# See R pakage coronavirus at: https://github.com/RamiKrispin/coronavirus  
# Data taken from: https://github.com/RamiKrispin/coronavirus/tree/master/csv  
# Last Updated  
Sys.Date()  
  
## Download and read csv (comma separated value) file  
coronavirus <- read.csv("https://github.com/RamiKrispin/coronavirus/raw/master/csv/coronavirus.csv")  
# write.csv(coronavirus, "data/coronavirus.csv")  
  
## Summaries and structures of the data  
head(coronavirus)  
str(coronavirus)  
coronavirus$date <- as.Date(coronavirus$date)  
str(coronavirus)  
  
range(coronavirus$date)  
unique(coronavirus$country)  
unique(coronavirus$type)  
  
## Set Country  
COUNTRY <- "Japan"  
df0 <- coronavirus[coronavirus$country == COUNTRY,]  
head(df0)  
tail(df0)  
(pop <- df0$population[1])  
df <- df0[c(1,6,7,13)]  
str(df)  
head(df)  
### alternatively,  
head(df0[c("date", "type", "cases", "population")])  
###  
  
## Set types  
df\_confirmed <- df[df$type == "confirmed",]  
df\_death <- df[df$type == "death",]  
df\_recovery <- df[df$data\_type == "recovery",]  
head(df\_confirmed)  
head(df\_death)  
head(df\_recovery)  
  
## Histogram  
plot(df\_confirmed$date, df\_confirmed$cases, type = "h")  
plot(df\_death$date, df\_death$cases, type = "h")  
# plot(df\_recovered$date, df\_recovered$cases, type = "h") # no data for recovery  
  
## Scatter plot and correlation  
plot(df\_confirmed$cases, df\_death$cases, type = "p")  
cor(df\_confirmed$cases, df\_death$cases)  
  
  
## In addition set a period  
start\_date <- as.Date("2021-07-01")  
end\_date <- Sys.Date()   
df\_date <- df[df$date >=start\_date & df$date <= end\_date,]  
##  
  
## Set types  
df\_date\_confirmed <- df\_date[df\_date$type == "confirmed",]  
df\_date\_death <- df\_date[df\_date$type == "death",]  
df\_date\_recovery <- df\_date[df\_date$data\_type == "recovery",]  
head(df\_date\_confirmed)  
head(df\_date\_death)  
head(df\_date\_recovery)  
  
## Histogram  
plot(df\_date\_confirmed$date, df\_date\_confirmed$cases, type = "h")  
plot(df\_date\_death$date, df\_date\_death$cases, type = "h")  
# plot(df\_date\_recovered$date, df\_date\_recovered$cases, type = "h") # no data for recovery  
  
plot(df\_date\_confirmed$cases, df\_date\_death$cases, type = "p")  
cor(df\_date\_confirmed$cases, df\_date\_death$cases)  
  
### Q0. Change the values of the location and the period and see the outcomes.  
### Q1. What is the correlation between df\_confirmed$cases and df\_death$cases?  
### Q2. Do we have a larger correlation value if we shift the dates to implement the time-lag?  
### Q3. Do you have any other questions to explore?  
  
#### Extra  
plot(df\_confirmed$date, df\_confirmed$cases, type = "h",   
 main = paste("Comfirmed Cases in",COUNTRY),   
 xlab = "Date", ylab = "Number of Cases")

:::

## gapminder Package

### Hans Rosling (1948 – 2017)

Hans Rosling was a Swedish physician, academic, and public speaker. He was a professor of international health at Karolinska Institute[4] and was the co-founder and chairman of the Gapminder Foundation, which developed the Trendalyzer software system. ([wikipedia](https://en.wikipedia.org/wiki/Hans_Rosling))

* Books:
  + Factfulness: Ten Reasons We’re Wrong About The World - And Why Things Are Better Than You Think, 2018
  + How I Learned to Understand the World: A Memoir, 2020
* Gapminder: <https://www.gapminder.org>
  + [You are probably wrong about: Upgrade Your World View](https://upgrader.gapminder.org)
  + [Bubble Chart](https://www.gapminder.org/tools/#$state$time$value=2020;;&chart-type=bubbles): Income vs Life Expectancy over time, 1800 - 2020
    - How many variables?
* Videos: [The best stats you’ve ever seen, Hans Rosling](http://www.edtech.events/the-best-stats-youve-ever-seen-hans-rosling/)

### Factfulness is … *From the book*

recognizing when a decision feels urgent and remembering that it rarely is.

To control the urgency instinct, take small steps.

* Take a breath. When your urgency instinct is triggered, your other instincts kick in and your analysis shuts down. Ask for more time and more information. It’s rarely now or never and it’s rarely either/or.
* Insist on the data. If something is urgent and important, it should be measured. Beware of data that is relevant but inaccurate, or accurate but irrelevant. Only relevant and accurate data is useful.
* Beware of fortune-tellers. Any prediction about the future is uncertain. Be wary of predictions that fail to acknowledge that. Insist on a full range of scenarios, never just the best or worst case. Ask how often such predictions have been right before.
* Be wary of drastic action. Ask what the side effects will be. Ask how the idea has been tested. Step-by-step practical improvements, and evaluation of their impact, are less dramatic but usually more effective.

# install.packages("gapminder")  
library(gapminder)

df <- gapminder  
df

## # A tibble: 1,704 × 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779.  
## 2 Afghanistan Asia 1957 30.3 9240934 821.  
## 3 Afghanistan Asia 1962 32.0 10267083 853.  
## 4 Afghanistan Asia 1967 34.0 11537966 836.  
## 5 Afghanistan Asia 1972 36.1 13079460 740.  
## 6 Afghanistan Asia 1977 38.4 14880372 786.  
## 7 Afghanistan Asia 1982 39.9 12881816 978.  
## 8 Afghanistan Asia 1987 40.8 13867957 852.  
## 9 Afghanistan Asia 1992 41.7 16317921 649.  
## 10 Afghanistan Asia 1997 41.8 22227415 635.  
## # … with 1,694 more rows

glimpse(df)

## Rows: 1,704  
## Columns: 6  
## $ country <fct> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan", …  
## $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, …  
## $ year <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 1997, …  
## $ lifeExp <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.854, 40.8…  
## $ pop <int> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372, 12…  
## $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.1134, …

summary(df)

## country continent year lifeExp   
## Afghanistan: 12 Africa :624 Min. :1952 Min. :23.60   
## Albania : 12 Americas:300 1st Qu.:1966 1st Qu.:48.20   
## Algeria : 12 Asia :396 Median :1980 Median :60.71   
## Angola : 12 Europe :360 Mean :1980 Mean :59.47   
## Argentina : 12 Oceania : 24 3rd Qu.:1993 3rd Qu.:70.85   
## Australia : 12 Max. :2007 Max. :82.60   
## (Other) :1632   
## pop gdpPercap   
## Min. :6.001e+04 Min. : 241.2   
## 1st Qu.:2.794e+06 1st Qu.: 1202.1   
## Median :7.024e+06 Median : 3531.8   
## Mean :2.960e+07 Mean : 7215.3   
## 3rd Qu.:1.959e+07 3rd Qu.: 9325.5   
## Max. :1.319e+09 Max. :113523.1   
##

### Questions

* List questions based on this data.
* What do you want to see?
* What kind of chart do you want to construct?