# PUBH732 Weeks 6-9

# R Programming

Emma Wei Chen, Ph.D.
(wei.chen@ku.ac.ae)

2025-02-18

# Lecture 1

# Intorduction to R

KU-PUBH732-Spring2025-Emma

## Outline

### Topics:

- Why use R for data analysis?
- Setting up the environment
- Basic syntax and functionality of R
- Loading datasets
- Running simple scripts

## Outline

### Class Activities:

- Running basic commands in R
- Create a data frame
- Loading datasets
- Writing and executing a small script

## What tools did you use before?

## What tools did you use before?

### List of Other Statistical Tools

- SPSS
- MATLAB
- STATA

### Why use R?

- free of cost;
- open access;
- a large number of up-to-date packages;

## Textbook

Huynh, YW (2019) R for Graduate Students.
**https://bookdown.org/yih_huynh/Guide-to-R-Book/**

- Other Materials
  Batra, Neale, et al. The Epidemiologist R Handbook. 2021.


**https://www.epirhandbook.com/en/** (Advanced with a focus on epidemiology)

## Setting up the environment

### Components

- R
- RStudio

### Installation

**https://posit.co/download/rstudio-desktop/**

## Setting up the environment

- Creating a New Project (.Rproj)
- Creating a Script (.R)
- Layout of RStudio Interface
  - Environment
  - Script
  - Console
  - Package
  - Working Directory

## Installing and loading packages

- What are packages?
  - A collection of free R tools that an R User wrote.
  - Packages are free but must be installed to your computer first. After installation, packages must be loaded into your RStudio library each time RStudio is opened/launched.
    - install.packages()
    - library()

## Installing and loading packages

- Different packages will provide different sets of tools, though some tools may have overlapping functions. In this case, the later loaded function will mask the earlier one.
  - e.g. select() is in both *MASS* and *dplyr*. Use dplyr::select() to specify.
- How do you know which package/function to use

## Useful packages

- Useful packages:
  - "*tidyverse*":
    - This package is actually comprised of multiple packages: graphing (*ggplot2*) and user-friendly formatting (*dplyr*).
  - "*writexl*": To produce Excel files from R data.
  - "*readxl*": To load my Excel files into R.
  - "*utils*": To read or write CSV files

## Basic syntax

- R has all of the basic arithmetic operations available (+, -, /, *, ^) and can function as a calculator. However, R is also a powerful tool for managing our data.
- Frequently used operational symbols:
  - ==, !=, >, >=, <. <=
  - %in% (similar to == but much more useful)
  - |, &
- Missing Value: NA

## Executing code

```
## Example 1
1 + 2 + 3 + 4
```

```
[1] 10
```

```
## Example 2
1 + 2 + 3 + # What happens if only select this line
  4
```

```
[1] 10
```

```
## Example 3
2 %in% c(1, 2, 3, 4)
```

```
[1] TRUE
```

```
## Example 4
```

```
2  1 + 2 + 3 + NA
```

```
[1] NA
```

## Executing code

- Note the behavior of logic operations when NA is involved.

```
## Example 5
TRUE | FALSE
```

```
[1] TRUE
```

```
## Example 6
TRUE | NA
```

```
[1] TRUE
```

```
## Example 7
FALSE | NA
```

```
[1] NA
```

## Executing code

- Note the behavior of logic operations when NA is involved.

```
## Example 8
TRUE & NA
```

```
[1] NA
```

```
## Example 9
FALSE & NA
```

```
[1] FALSE
```

## Using objects

- Assigning values to objects (<-)
- What's the difference between <- and = in R?
  - <- is used for general assignments to object
  - = is used for parameters in functions

## Using objects

```r
## Example for using <-
x <- c(1, 2, 3, 4)
print(x)
```

```
[1] 1 2 3 4
```

```r
y <- c(1, 2, 3, 4, NA)
## Example for using =
mean(y, na.rm = FALSE)
```

```
[1] NA
```

```r
mean(y, na.rm = TRUE)
```

```
[1] 2.5
```

## Using objects

- Why you want to use objects
  - save information in an object so you don’t need to type them in again
  - to apply functions on the object
- Most commonly used object type in R: **data frame**
  - data.frame() is the function to create a new data frame

## Data frame

```r
## Example for create a data frame
df <- data.frame(
  Height = c(150, 60, 145, 187, 175),
  Weight = c(55.5, 60.2, 62.3, 50.0, 48.1)
)

print(df)
```

```
  Height Weight
1    150   55.5
2     60   60.2
3    145   62.3
4    187   50.0
5    175   48.1
```

## Data frame

- Similar as a table.
- Each column can have a different data type
  - Commonly used data types: logical, integer, numeric, character,factor

```
## Example for logical data type
df$Smoker <- c(TRUE, FALSE, TRUE, FALSE, FALSE)

## Example for integer data type
df$ID <- c(1, 2, 3, 4, 5)

## Example for numeric data type
df$Height
```

```
[1] 150  60 145 187 175
```

## Data frame

- Note the difference between character and factor

```
## Add another column to df
df$City = c("Abu Dhabi", "Dubai", "Dubai", "Abu Dhabi", "Al Ain")
df$City_f = factor(df$City, levels = c("Abu Dhabi", "Dubai", "Al Ain"
summary(df)
```

```
     Height          Weight        Smoker              ID        City
 Min.   : 60.0   Min.   :48.10   Mode :logical   Min.   :1   Length:5
 1st Qu.:145.0   1st Qu.:50.00   FALSE:3         1st Qu.:2   Class
:character
 Median :150.0   Median :55.50   TRUE :2         Median :3   Mode
:character
 Mean   :143.4   Mean   :55.22                   Mean   :3
 3rd Qu.:175.0   3rd Qu.:60.20                   3rd Qu.:4
 Max.   :187.0   Max.   :62.30                   Max.   :5
       City_f
 Abu Dhabi:2
 Dubai    :2
 Al Ain   :1
```

```
Sharjah  :0
```

## Data frame

- Each row represents a single observation across all variables.
- Columns have names (variable names), and rows can optionally have names.
- You can access elements using `df[row, column]` notation.

## Data frame

```r
# ## Add another column with less observations
# df$Gender = c("F", "M", "M")
#
# ## Add another column with more observations
# df$Gender = c("F", "M", "M", "F", "F", "M")

## Add another column with the same number of observations
df$Gender = c("F", "M", "M", "F", "F")

## Add row names
rownames(df) <- c(1, 2, 3, 4, 5)
print(df)
```

```
  Height Weight Smoker ID      City    City_f Gender
1    150   55.5   TRUE  1 Abu Dhabi Abu Dhabi      F
2     60   60.2  FALSE  2     Dubai     Dubai      M
3    145   62.3   TRUE  3     Dubai     Dubai      M
```

```
4    187   50.0  FALSE  4 Abu Dhabi Abu Dhabi      F
5    175   48.1  FALSE  5    Al Ain    Al Ain      F
```

## Datasets

- Build-in Datasets
  - data()
- Datasets from R Packages. Below are PH&E related packages offers practice datasets:
  - *pubh* **link**
  - *Epi* **link**
- External data files
  - read_xlsx() from *readxl* package
  - read.csv() from *utils* package

## Writing a simple script

- Header and Comment Organization
- Naming (Note: R is case-sensitive!)
  - Choose Descriptive and Concise Names
  - Do Not Start with a Number or Symbol
  - Avoid using punctuation other than a period (.), dash (-), or underscore (_)
  - No Spaces
  - Consistency
  - Avoid Names Already in Use

## Writing a simple script

- A few naming styles
  - Snake case (snake_case), e.g., group_mean
  - Camel case (camelCase), e.g., groupMean
  - Dot-separated (dot.case), e.g., group.mean
    - This is not recommended because dot is used in some R functions, e.g. data.frame()
  - Uppercase (UPPER_CASE), e.g., GROUP_MEAN

## Writing a simple script

- My personal style
  - I name key objects using Title Snake Case, e.g., Total_Count
  - I name temporary objects using snake case, e.g., city_tbl

## End of the current lecture

## Any questions/feedback?

## Class activities:

- Practice basic commands in R
- Create a data frame that include at least 5 different types of data: logical, integer, numeric, character, factor
  - do operation on one column to create another column
  - apply function summary() for your data frame to see what happens
- Loading datasets
  - try different ways to load datasets
- Writing and executing a small script
  - source()

## Before Next Class

- Read “II Tidyverse” on the **R for Graduate Students**