Agenda

• Introduction
• Getting Started
• Data Manipulation with dplyr
• Plotting with ggplot2 package
Why R?

• R is a popular statistical programming language
• Excel has its limitation:
  • Hard to implement statistical models
  • Graphing in Excel can also be challenging at times
• R can solve all the above problems and also perform the task you can do in Excel faster
Getting Started
Starting R

- We will use Rstudio, an Integrated Development Environment, to program
Starting R

• To actually write and save your code, you will need a R script, we can add a R script under “File”, or (for window) use Ctrl+Shift+N
Assignment

• We use “<-“ for assignment
• Use “c(...)” to create “vector”
Arithmetic

- R do element-wise operation automatically
- It also recycle through vector if their length does not match

```r
> c(1,2,3)+c(4,5,6)
[1]  5  7  9
> c(1,2,3)*c(4,5,6)
[1]  4 10 18
> c(1,2,3,4) - c(5,6)
[1] -4 -4 -2 -2
```
Subsetting

- You can subset a data table (matrix/dataframe/tibble) using indices
  - `d[1,3]` represent the 1\textsuperscript{st} row 3\textsuperscript{rd} column of `d`
- You can also subset using column name of the data table with "\$
- You can subset with row name and column name as well

```r
> d
  col1 col2 col3
row1 1 4 7
row2 2 5 8
row3 3 6 9
> d[1,3]
[1] 7
> d$col1
[1] 1 2 3
> d[row1,]
  col1 col2 col3
row1 1 4 7
> d[, "col1"]
[1] 1 2 3
```
Your Turn!

1. Create a vector from 1 to 10 and assign it the variable “v”
2. Increase the odd elements of v by 1 and decrease the even elements by 2
   (Hints: vector recycling)
3. We will practice subsetting with the built-in data set “iris”
   a. Type iris on your console or script to examine the data
   b. Subset the 4th row and 2nd column of iris
   c. Subset the Petal Length column
Basic Data Manipulation with dplyr
Setup

- We can install and load “packages” from internet in R
- We will load “tidyverse” package, which includes the “dplyr” package and “ggplot2” package

```
1  install.packages("tidyverse")
2  library(tidyverse)
```
Reading Data

• R has build-in functions for reading data, but generally they are slow
• The “tidyverse” package contains the readr package that reads data quickly
• Usually, data are in “.csv” type
Filter() function

- The filter() function extracts rows by their value
- The syntax for filter is `filter(data, logical_expression_for_extracting_rows)`
- The filter() function ignore rows that evaluates to FALSE and NA
Select() function

• The select() function extracts columns by their value
• The syntax for select is `select(data, col_names_condition)`
• There are helpers function for select
  • `start_with()`: select name that start with whatever you specify
  • `ends_with()`: select name that ends with whatever you specify
  • `contains()`: Select names that contain a specific string
Select() function

```r
> select(iris, Petal.width)
   Petal.Width
1     0.2
2     0.2
3     0.2
4     0.2
5     0.2
6     0.4
7     0.3
8     0.2
9     0.2
10    0.1
11    0.2
12    0.2
```

```r
> select(iris, ends_with("Length"))
   Sepal.Length Petal.Length
1     5.1      1.4
2     4.9      1.4
3     4.7      1.3
4     4.6      1.5
5     5.0      1.4
6     5.4      1.7
7     4.6      1.4
8     5.0      1.5
9     4.4      1.4
10    4.9      1.5
11    5.4      1.5
12    4.8      1.6
13    4.8      1.4
```
**Mutate() function**

- Compute new variable from existing ones and add it to the data table
- Syntax: `mutate(data, var_name = [operations with existing columns])`
Your Turn!

1. Using the filter function, capture all the rows that has sepal width greater than 1 in the dataset iris
2. Using the select function and its helper function, capture all the columns that start with the letter “s”
3. Create a separate column in iris that contains the sum of the sepal width and length
Plotting with ggplot2
What is ggplot2?

• ggplot2 is a R package that is very useful for data visualization
• “gg” stands for grammars of graphics
• Provides a systematic way to create visually-pleasing graphs
• Basic syntax
  • ggplot(data = data_name, aes(x= .., y=...) ) +geom_...
Scatterplot

- To create a scatterplot, we use the `geom_point` function
- `ggplot(data = data_name, aes(x= .., y=...)) + geom_point()`
Line plot

- To create a lineplot, we use the `geom_line` function
  - `ggplot(data = data_name, aes(x= .., y=...) ) + geom_line()`
Histogram

- To create a histogram, we use the `geom_histogram` function
- `ggplot(data = data_name, aes(x)) + geom_histogram()`
Your Turn!

0. Type in mtcars on the your script or console and take a look at the data.
We will be using this dataset for the following exercises
1. Create a scatterplot for “disp” and “mpg” and save it as splot
2. Create a line graph for “wt” and “qsec” and save it as line_graph
3. Create histogram for “cyl” and save it as histo
How to Make Plots Prettier

• We can specify additional elements in the ggplot to make plots look nicer!
• Example:
  • Rename axis
  • Add a title
  • Color
  • Themes
  • Add a smoother function
  • And More!
Line plot Revisited

```r
library(tidyverse)
install.packages("ggthemes")
library(ggthemes)
plot2_nice <- ggplot(data = economics,
aes(date, unemploy))+
  geom_line(size = 1, color = "blue")+
  xlab("Year")+
  ylab("Unemployment")+
  ggttitle("Unemployment over Time")+
  theme_economist()
plot2_nice
```
ggplot2 is very powerful
Questions?