Review on Basics of R

1. Objectives

- Importing data
- Dataframe
- Basic descriptive functions
- The '**by**' function

This review lab is not intended to be comprehensive. Students thus are strongly advised to review all labs in the course of Probability and Statistics (PAS).

2. Installing R & Rstudio

Detailed instruction on how to install R and Rstudio will be posted on your Google class.

3. How to create a variable

If we want to create a variable **x** that holds the result of the expression: 27*log10(3) + sqrt(65), we can try:

```
➢ x <- 27*log10(3) + sqrt(65)
```

The "<-" is called the **assignment operator**, which assigns the value of the expression on the right to whatever object we specify on the left. Note that R is a case-sensitive language, so x and X are different. We are **not** required to declare the type of the variable. It is possible for a variable to hold any object such as a **function** or a **plot**.

Exercise 1: Let's create a variable **y** that takes the value equaling the square root of ten times of x. Then check the data type of **y** using the **class** function.

```
➢ class(y)
```

4. Main data types

In R, the basic data types include numeric, character and logical. Numeric data is the most common data type. Numeric data comprise integers and decimals. For example, type in the following code:

```
➢ a <- 5
➢ class(a)
```

The next type of data is character. As its name suggests, character data are represented by strings of characters and/or numbers.

```
➢ b <- "blue"
➢ class(b)
➢ b <- "blue5"
➢ class(b)
➢ b <- "5"
➢ class(b)
```

Logical data can be either TRUE or FALSE. Let's type in:

- `x <- FALSE`
- `y <- TRUE`
- `x==y`

Notice that in the last statement, we use a comparison operator ==. Other comparison operators are: <, >, !=, >=, <=. For example,

- 5 >=7
- 3 < 6
- TRUE !=FALSE
- 9 <= 10

5. Managing working directory

To set our working directory, we can use **setwd("mydirectory")**. Next, type **getwd()** at the command prompt. Our current working directory should be listed. R will read files from and save results to this directory. **It is necessary that you set up your working directory in every lab before you run any codes.**

Exercise 2: Create a folder in your D directory and name it **Lab1**. Go back to R and change the working directory to "D:/Lab1". Now try to display our current working directory.

6. Importing data from a delimited text file

You can create a CSV file by entering data into Excel, and then save the file as a CSV (comma delimited) file. Now, open the file **mtcars.xlsx**, and try saving it as a CSV file with the same name **mtcars**. Copy and paste the file to be used for the lab into your working directory. We can use the **read.table()** or **read.csv()** function to import csv data into R.

The code for reading the .csv file is shown below:

-
  ```
  data1 <- read.table("mtcars.csv",header=TRUE, sep=",",
  quote="\"", stringsAsFactors=FALSE)
  ```

Exercise 3:

a. What does the following code do?
   - `extracted1 <- data1[1:10,]`
   - `extracted2 <- data1[,1:3]`
b. Check the structure of **data1** using **str()** function. How many variables and observations are there? Extract data for **mpg**, **cyl** and **am** variables and assign it to a data frame called **extracted3**.

7. Data frames

A data frame in R is analogous to a dataset in SPSS, SAS or Stata. It is more general than a matrix in that it permits different data types (numeric, character…) for different columns. Usually, a column represents data for a variable (column name = variable name), and each row represents data for one observation (or case in the language of SPSS). Data for one column must be of the same type.

Note down the variable names, the number of variables, and the number of observations in **data1**. What does the first column represent?

Let's try the following

```
mean(data1$mpg)
sd(data1$mpg)
summary(data1$mpg)
plot(data1$mpg, data1$wt)
```

In the above code, we access the **mpg** variable of the **data1** data frame via the **$** notation. This notation signifies that we are dealing with a specific variable from that data frame.

8. Descriptive statistics

The following code produces summary statistics for 3 variables simultaneously.

- `summary(data1[c("mpg", "drat", "wt")])`

If we want more descriptive measures, the function **stat.desc()** in the **pastecs** package can do a good job. Let's install the **pastecs** package. Then follow the code listing below:

- `library(pastecs)`
- `myvars <- c("mpg", "drat", "wt")`
- `stat.desc(data1[myvars], basic=TRUE, desc=TRUE, norm=FALSE, p=0.95)`

Note that by default, **basic=TRUE, desc=TRUE, norm=FALSE, p=0.95**.

Note that all numerical descriptive measures introduced in the labs (e.g. sd, var, etc.) are sample measures. If you wish to compute measures for a population, in some cases you have to adjust the obtained values by an appropriate formula.

The **describeBy()** function in the **psych** package provides more summary statistics by group.

Let's try:

- `describeBy(data1["mpg"], data1$cyl)`

Exercise 4: Use **describeBy()** function to produce summary statistics for mpg variable grouped by am and cyl. Use **mat=TRUE** to display results in the form of a matrix. You should use **list** to include more than 1 grouping variable.

9. Apply a Function to a Data Frame Split by Factors

The **by( )** function applies a function to each level of a factor or factors of a data frame.

The basic format of this function is as follows:

```
by(data, a factor or a list of factors, a function to be applied, ...,
simplify = TRUE)
```

For instance, if we want to obtain a summary on **mpg** variable separately for each number of **cylinders** in **data1** data frame, we can use the following code:

```
➢ by(data1$mpg,data1$cyl,summary)
➢ by(data1$mpg,list(Cylinder=data1$cyl),summary)
```

If we want to produce a summary for **mpg** variable for each level of both **am** and **cyl** variables, we can type in R the following code:

```
➢ by(data1$mpg,list(data1$am, data1$cyl),summary)
➢ by(data1$mpg,list(AM=data1$am, Cyl=data1$cyl),summary)
```

Exercise 5. Load the **textbooks** data from the **openintro** package. Examine a summary of the price differences between new textbook sold at the UCLA bookstore and on Amazon for two subsets: courses where one book is required and where more than one book is required.

```
➢ install.packages("openintro”)
➢ library(openintro)
➢ data(textbooks)
```

10. Recap questions

a. What does the assignment operator do?
b. What are the main data types?
c. What function is used to check for data type of an object?
d. Print the **data1** dataset. Find the sample mean and standard deviation of the **weight** variable. Display the five-number summary for the **weight** variable. Provide a summary for **hp** variable grouped by **cyl** variable using the functions **describeBy()** and **by().**