Lab Session 1

1 An introduction to R and RStudio

1.1 What are R and RStudio?

During the lab sessions, we will work with R. R is a computer programming language that can be used for statistical analyses. You can find a manual at http://www.r-project.org/, following "manuals" to "An Introduction to R". For this course, however, you are only required to be familiar with the R instructions used during the classes and the lab sessions. You can download R for free using the previously stated link.

RStudio is an integrated development environment for R which provides a split screen view of an editor window, the R console, workspace items, the file directory, plot images and several other key features all at the same time. I will use RStudio during the computer sessions since I enjoy having easy access to and visibility of these features. You can download a free version of RStudio (which is what I will be using) at http://www.rstudio.com. Note that you must download R prior to downloading RStudio.

1.2 First steps in RStudio

When you open RStudio, the graphical user interface (GUI) appears. In this interface, you can see the R console on the left-hand side of the screen, i.e. the window in which commands are given and output is returned. Under the default text that is already given when you fire up this window, you can see the ">" sign, followed by the text cursor. This is called the "prompt", through which R indicates that it is ready to execute a new command.

Firstly, R can be used as a simple calculator. Try to enter

```
1350 + 6750
```

R will give you the following answer:

```
[1] 8100
```

[*Tip: If you want to re-execute your last command, you do not have to type it in all over again. Just press on the upwards arrow button on your keyboard and the last command will reappear. If you want, you can now make adjustments to this command by using the left and right arrow*

buttons.]

One can also create objects in R. You could consider an object to be a "box" to which you give a name and in which you store data such as numbers, vectors, matrices, etcetera. Suppose that we want to create the object "x" to which we want to attribute the value "5". You can do this by executing the command

```
x <- 5
```

If you would now want to ask R what the object "x" contains, you can simple give the command

```
x
```

and R will reply with

```
[1] 5
```

Similarly, you can attribute a vector to "x":

```
x <- c(3,7,10)
```

If you would now like to find out what the second element of "x" is, you can give the command

```
x[2]
```

and R will reply with

```
[1] 7
```

[When assigning objects, please take into account that:

- *You cannot use object names that belong to R's internal vocabulary. For example, you cannot create an object with the name "sqrt".*
- *If you use an object in a calculation and you change the value of this object afterwards, the result of your calculation will not be changed automatically. You need to re-run the calculation command yourself.*
- *R is case sensitive. "x" and "X" are thus two different objects.]*

1.3 Logarithm and differences

Let "x" again be defined as the vector described previously:

```
x <- c(3,7,10)
```

You can now create a new object called "log_x" that represents the natural logarithm of "x"

```
log_x <- log(x)
```

To see the value of "log_x", type in

```
log_x
```

and you get

```
[1] 1.098612 1.945910 2.302585
```

Similarly, you can define "d_x" as the first differences of "'x"

```
d_x <- diff(x)
d_x
[1] 4 3
```

and "dlog_x" as the first differences of the log-transformed "x"

```
dlog_x <- diff(log(x))
dlog_x
[1] 0.8472979 0.3566749
```

[*If you ever need some further documentation on one of R's functions, for example on the log function, you can use the question mark functionality:*

```
?log
```

If you would like to execute a function, for example taking the logarithm of a number, but you do not know the exact name of the function, you can use:]

```
??logarithm
```

1.4 R scripts

Suppose that you have been working in R for several hours, but it is getting late and you want to continue your work tomorrow. If you would now close R, all of your work would be gone. To avoid this problem, we will not give R commands directly through the R Console, but save them in an R script. You can open a new one by clicking on File > New File > R Script. You can also use the "New File" icon in the upper left-hand corner of the RStudio screen, and select "R Script." You can now enter commands like we did before and execute them by first selecting the command and then using Ctrl + R (Command + Return for Macs). When you are finished working in R, save the script by "File" and then "Save as". You can later re-open these R scripts in R to continue working with them or you can also open them in Notepad.

[Tip: When you are writing R scripts, the code can become very long and obscure. To clarify your work, you can use commentary lines in R to provide your code with additional info. Such commentary lines should always be preceded by the "#" sign.]

2 An introduction to R Markdown

2.1 What is R Markdown?

R Markdown is a file format for making dynamic reports in R. We call these reports *dynamic* because they allow the user to include not only text, but script, tables and figures programmed in R as well. Only a few basic tools are required to transform R code into a well formatted R Markdown report. *Note that all homework assignments will be submitted as R Markdown reports.*

The following overview of R Markdown is based on the walkthrough provided by:
http://rmarkdown.rstudio.com/articles_intro.html.

2.2 First steps in R Markdown

To create a new R Markdown report, click on File>New File>Text File. You can also open a new R Markdown file through this series of commands; however, the default R Markdown file has many unnecessary lines of code which we will avoid by creating a blank text file. Once the file is opened, save the document using your preferred title and the extension ".Rmd". This extension indicates that the file is an R Markdown document, and it enables several useful buttons on the RStudio screen.
Begin your document with a title, author list, the date, and the output style. As a default, it should look like:

—
title: "Untitled"
author: "Megan Gelsinger"
date: "January 25, 2018"
output: html_document
—

Text in this file format is written in *markdown*, which is very similar to writing in Word. Only several new conventions need to be learned:

- Headers need to be surrounded by pound symbols on their own lines (#)
- Italicized and bold text need to be surrounded by one or two asterisks (*), respectively

- Lists are initiated by a blank line, followed by an asterisk indicating an item in the list

Once you are ready to render the document, click the "Knit" button at the top of the screen. Note that you do not need to download any additional programs for html documents to render. If you want to produce a pdf or Word document, you will need a version of LaTex or Microsoft Word, respectively, loaded on your computer.

2.3 Adding R Code to Report

Now let's learn how to add R code to the report. The most simple *chunk* of code to include in the report is normal R code. To initiate this chunk, one surrounds the code with two lines of three backticks. After the first set of backticks, include {r}, hit enter, and type your desired code:

````
```{r}
x <- c(2,3)
length(x)
plot(x)
```
````

Sometimes, we do not wish to see all of the output from the code. By modifying the preamble of the chunk, we can adjust what output is included in the report:

- {r, eval = FALSE}: puts code in report, but suppresses output

- {r, echo = FALSE}: suppresses the code, but includes the output (useful for plots)

More advanced modifications can be made, but these basics are enough to produce the level of reports expected for this course.

3 Some tips before you go

- Do not forget to save your R script before closing RStudio!
- Practice working with the basic functionalities in R before the next computer session.