Lab 1: Introduction to R

In this lab, you will learn the basics of coding in R and apply your new skills to calculating descriptive statistics and plotting data. The learning goals of this lab are to:

- Install and become familiar with R;
- Understand the basics of visualizing and summarizing data - a skill that will be used in future labs.

The first task is to read and work through the below introduction to R (sections 1 - 14). If you want an additional introduction, see *A (very) short introduction to R* by Paul Torfs and Claudia Brauer, which is also posted on Canvas under Lab 1. Both documents will get you started using R. R functions included below are not always defined, so your task is to figure out what they do so that you can use them in the future. All functions used in this lab are basic functions necessary for daily use of R.

The second task is to answer basic questions about descriptive statistics using R. At the end of the document there are a few problems to solve. *Submit your answers, including any requested graphs, and your R-code to the class Canvas site under the Assignments folder before our online class. Your lab is due the day before your lab meets. So, for example, if you have a Tuesday lab, your lab is due by 11:59pm on Monday night.*

What is R?

R is powerful software for interacting with data. With R you can create sophisticated graphs, carry out statistical analyses, and develop and run simulations. R is also a programming language with an extensive set of built-in functions, so you can, with some experience, extend the language and write code to build your own statistical tools.

R is an open source implementation of the S language, which has been around and widely used for more than twenty years, first as S and then as the commercially available S-PLUS. A core team of statisticians and many other contributors work to update and improve R. Most importantly, R is free, high-quality statistical software that is useful for learning and doing statistics.

As described on the R project homepage:

> "R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files."
>
> "The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions. Most of the user-visible functions in R are written in R. It is possible for the user to interface to procedures written in the C, C++, or FORTRAN languages for efficiency. The R distribution contains functionality for a large number of statistical procedures. Among these are: linear and generalized linear models, nonlinear regression models, time series analysis, classical parametric and nonparametric tests, clustering and smoothing. There is also a large set of functions which provide a flexible graphical environment for creating various kinds of data presentations. Additional modules are available for a variety of specific purposes."

R Resources

There are hundreds of books, online resources and documents, blogs, and webpages on R. See the Resources file on the class Sakai webpage for a few useful R resources and shortcut reference cards. Or, look under

Documentation on the R project homepage for a list of manuals, FAQ's, books, etc. This introductory document just scratches the surface of R, but will hopefully get you started.

Here are a few additional resources in addition to those on the class Sakai webpage.

https://www.rstudio.com/resources/cheatsheets/ - cheat sheets, especially helpful are the R Markdown cheat sheet and the data visualization cheat sheet (ggplot)
https://rfun.library.duke.edu/ - workshops and tutorials
https://r4ds.had.co.nz/ - R for Data Science book
http://www.cookbook-r.com/ - Cookbook for R – random helpful tasks
https://kapeli.com/cheat_sheets/LaTeX_Math_Symbols.docset/Contents/Resources/Documents/index - Math symbols used for R Markdown

Downloading R from the Web

Go the R project homepage at http://cran.us.r-project.org/.

Download R for Windows. Click on the link `Download R for Windows`, then click on the link `base`, and finally click on `Download R 4.0.2 for Windows` (or the recent version, which could have a larger number). This begins the download of a file whose size is about 62MB. After the download is complete, double click on the downloaded file and follow the onscreen installation instructions.

Download R for (Mac) OS X. Click on the link `Download R for Mac OS X`, then click on `R-4.0.2.pkg` (latest version) to begin the download. Once it is downloaded in your Downloads folder or on your Desktop, double click on the downloaded file. Your Mac will unstuff the downloaded file and create an R folder. Inside this folder, there are many files including one with the R logo. Drag a copy of this to your Desktop and then drag the whole R folder to the Applications folder (located on the hard drive). After completing this, you can drag the original downloaded file to your trash bin.

R Studio After setup, you should see the R icon on your desktop. Clicking on the icon would open the standard interface, but many users prefer the RStudio interface. RStudio makes R easier to use, by including a code editor, debugging and visualization tools. It also facilitates printing R code and output as pdf, html, and Word files (we will come back to this in a future lab).

To install RStudio, go to http://www.rstudio.com/, click on "Download RStudio"", and then scroll down and click on"RStudio Desktop" for your system. Make sure to choose the Open Source License, not the $995 Commercial Licence. Download the software as above, and then explore. Note that if you click on the RStudio icon, you do not also have to open the standard R program - it interfaces for you.

MiKTeX After installing RStudio, go to https://miktex.org/download and download the appropriate version of MiKTeX for either Windows or Mac. MiKTeX installs many of the things needed to use TeX for typesetting. You'll want to download the Installer first on the page by clicking "Download".

Getting Started in R

This document will describe some of the basic functionality of R. Usually, you interact with R through a command-line interface (CLI) - type or paste a command in the console and R responds. Alternatively, commands can be written in a text editor and submitted to the console using the 'source' command. Once you have mastered a few commands, the CLI gives you control of an extremely powerful tool for interacting with data. Gaining mastery of a few R commands does take some learning and patience, especially if R (or any object-oriented programming environment) is new to you. You will undoubtedly experience some challenges as you work to learn a new skill that is not wholly intuitive, but it is well worth the effort. Onward!

The CLI version of R provides us with two prompts, that is, two signals that it is ready for input. They are the caret:

`>` which means that R is ready for a new command, and the plus sign:

+ which means that R is waiting for you to finish the current command. A principal advantage of the CLI is that it simplifies the development and use of scripts. These allow us to keep a permanent, documented text record of the steps that we have done. Also note that # can be used to annotate code within a text editor or text file. Anything text or code on the same line after the # will not be run.

The below commands provide a glimpse of what R can do. Work through each of the examples below, typing the commands into R to see the results for yourself.

Using R as a Calculator

You can use R as a calculator. What do each of the following lines do?

```
2 + 2
12 * 3 - 10/2 + sqrt(16)
3^2
1:10
sum(1:10)
mean(1:10)
sd(1:10)
```

Defining a variable and assigning values

You can use R to create a variable (what is a variable?). Assign variables using the <- sign (syntax that is meant to like an arrow). For example, type the below to assign a vector of numbers to the variables x and y.

```
x <- c(0,1,2,3,4,5,6,7,8,9,10)
y <- c(2.96, 4.20, 2.84, 3.84, 6.57, 6.95, 9.32, 10.57, 9.72, 11.57, 11.53)
x
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
y
```

```
##  [1]  2.96  4.20  2.84  3.84  6.57  6.95  9.32 10.57  9.72 11.57 11.53
```

Creating a data frame

You can also use R to create a data frame, which is a collection of variables. More simply, a data frame is a data set. If you have a collection of variables in your environment (such as x and y), you can create a data frame, as follows:

```
xy <- data.frame(x=x,y=y)
xy[,1]
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
xy[,2]
```

```
##  [1]  2.96  4.20  2.84  3.84  6.57  6.95  9.32 10.57  9.72 11.57 11.53
```

Note in the above code we created a data frame called xy, and that we then looked at the first variable (x), and then looked at the second variable (y). We did this using square braces. The [,] syntax is used to denote rows and columns. For example, were to have typed [1,2] it would have returned the entry in the first row, second column. Generally, [i,j] indexes the i^{th} row and j^{th} column. We can also do this using a different syntax:

```
xy$x
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
xy$y
```

```
##  [1]  2.96  4.20  2.84  3.84  6.57  6.95  9.32 10.57  9.72 11.57 11.53
```

Similarly, if we wanted to look at the 1st row of the `x` variable, we could type:

```
xy$x[1]
```

```
## [1] 0
```

All of this is to begin to show you that R has a lot of flexibility for interacting with data. Since we're using R in a statistics class, you might imagine that R also has lots of statistical capabilities. You imagine correctly. So, let's get started.

Plotting in R

To plot data in R, we will use an R package (also known as a library) called 'Tidyverse'. The tidyverse is a collection of other R libraries/packages that all have a common syntax and way of handling data. To install a package for the first time in R, you can simply type:

```
install.packages("tidyverse")
```

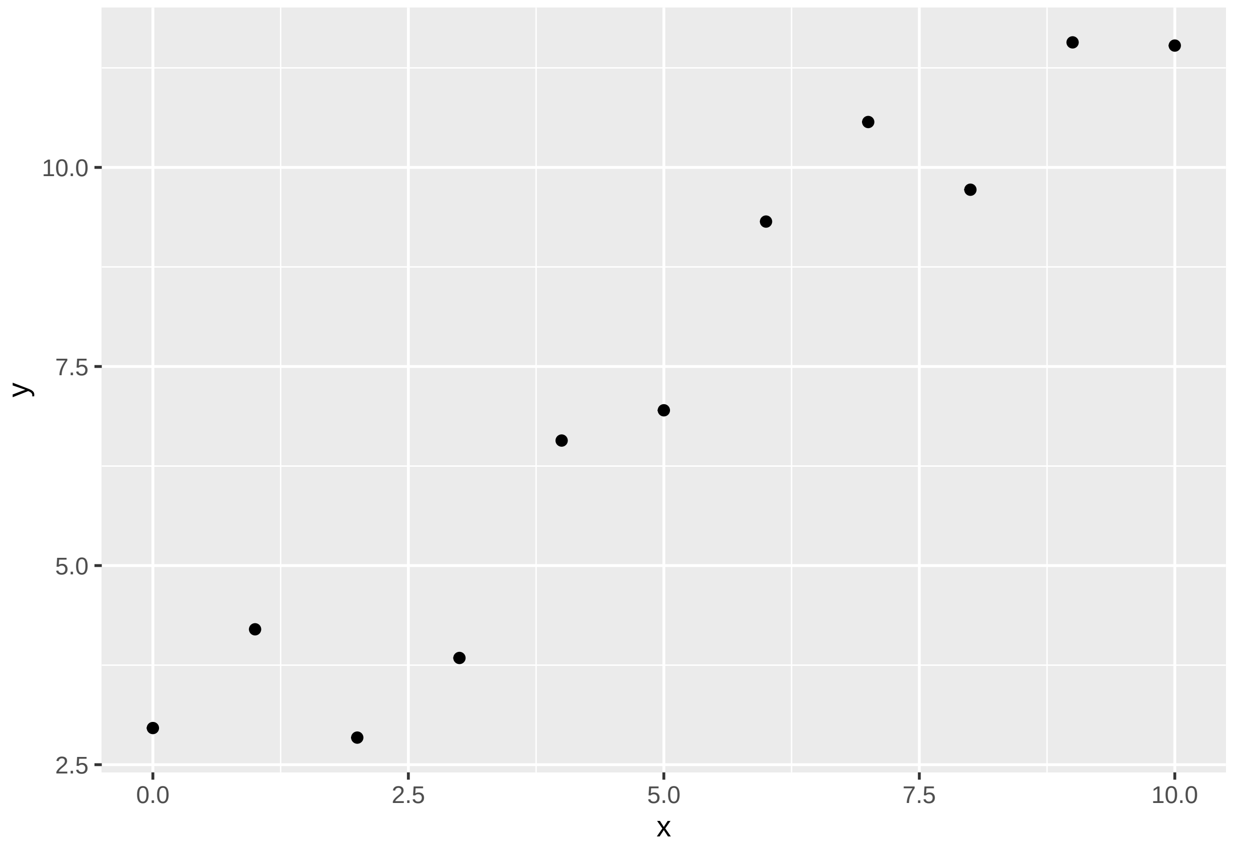
Then, whenever you open R, you need to load the packages you are going to use. You can do that, as follows:

```
library(tidyverse)
```

We can then make a basic scatter plot of our data using an R packages called ggplot2, which is contained within the tidyverse. The syntax is as follows:

```
### Create plot
gg <- ggplot()+
      geom_point(data=xy,aes(x=x,y=y))

### Print plot
gg
```



# INSTRUCTIONS FOR LAB: You will answer the following questions using R statistical software and prepare a document using R Markdown, which will allow you to compile the document as a PDF. Your TAs have been instructed to help you learn to do this. There are also resources included above (e.g. R, RStudio, MikTex). When reporting numbers, make sure to include units. Also, you must show your code, which is why you're using R Markdown. Also also, you must include comments in your code that describe what your code is doing. Make sure you invest time in learning R Markdown in the early part of the semester, since this course will only get more challenging as we move forward. Note, you may talk to your classmates, but you *must* do your own work. Your TAs are a resource for you. Ask them questions in the lab; ask them questions via email.

Submit your assignment as: LASTNAME_FIRSTNAME_LAB1.pdf.

DESCRIPTIVE STATISTICS

Let us suppose that the data contained in $(\mathbf{x}, \mathbf{y})$ consists of two variables: $\mathbf{x} = x_1, x_2, \ldots, x_n$ is a vector of tree heights measured to the nearest meter), while $\mathbf{y} = y_1, y_2, \ldots, y_n$ is a vector of tree diameters (in centimeters).

Remember, a one of the goals of this course is to improve your skills at interpretation and communicating statistics. This means, among other things, indicating the units that things are measured in. In other words, don't just report a number.

1 Measures of location (1 point)

a) What is the mean tree height (to the nearest meter) and the mean tree diameter (to the nearest centimeter)? Hint: to estimate the mean of a variable named 'blah' in a data frame named 'df in R, use 'mean(df$blah)'.

b) What are the medians of x and y? Hint: to estimate a quantile of a variable named blah in a data frame named df in R, use:

```
quantile(df$blah,probs=p)
```

where p is the percentile of interest. Note, you can return multiple quantiles if you feed in a vector-valued argument:

```
quantile(df$blah,probs = c(p_1,p_2,...,p_k))
```

where each p_i is a percentile.

2 Measures of spread (1 point)

a) What is the range of both x and y. Report the range in the technically correct way, as well as the technically incorrect (but better) way. Hint: There is more than one way to compute this. Is there a range() function in R? Is there a min() function? A max() function?

b) What is the variance of both x and y. Interpret the variance in terms of how we defined it in class.

c) What is the standard deviation of both x and y. Interpret the variance in terms of how we defined it in class.

3 Interpret and Communicate Results (1 point)

Properly interpreting and communicating results is a central goal of this course. We will start here.

a) Make a scatterplot of tree height (to the nearest meter) and tree diameter

b) Develop a hypothesis about the relationship between tree height and tree diameter

c) Suppose that you fit a regression line and you estimated an intercept b=2 and a slope m = 1. Interpret both of these numbers in terms of tree height and tree diameter.

d) Add this regression line to the scatter plot above. Note: this will require you to explore the functionality of the ggplot2 library.