Lab 3 Notes

Jon Ashbrock

January 31, 2018

1 Introduction

In today's lab the student will learn to:

1. Create functions in **R**
2. Create basic plots to help visualize data

Throughout today we will write a program that, using functions, estimates some statistical quantity and creates some associated visualizations of the data we generate. The following discussion point is the program we will be writing throughout the day.

Discussion 1. *Write the pseudo-code for a program that does the following. Take a random sample of n numbers selected uniformly from the interval $[0, 1]$. What is the average value of the smallest number that we generated?*

2 Functions in R

A function is a particularly useful object in **R**. We have seen many examples of functions already such as "runif" or "matrix". Just like functions in your calculus class, functions in **R** take some variables as input, transforms them, and returns some other variables as output. The input to a function is called the arguments of the function. For example, the "runif" function picks random numbers uniformly from some interval. However, we need to tell the function how many random numbers and from what interval to select the numbers.

Discussion 2. *To write the program described in the introduction, we are going to first write a function that generates n random numbers from* $[0, 1]$ *and returns the value of the smallest one. What parameters and outputs should this function have?*

Functions are used for multiple reasons. First, they are used to save time in programming by turning multiple lines of code into a single line that can be used multiple times. Second, they make your program more "readable" and easier to write. The syntax of a function is as follows

```
Function_Name <- function(argument1, argument2, ... ) {
# Function Body
return(variable1,variable2,...)
}
```

What the above function does is accepts argument1, argument2, etc. as inputs to the function. Then the code will do whatever lines are in the function body. Finally, the output of a function is variable1, variable2, etc. A function in **R** is the same thing as a function in mathematics, it takes a number of inputs and transforms them into some number of outputs based on how it is defined.

Lastly, in order to use a function in a code, you need to open the function script and run it. Then you can run another script which uses the function.

Discussion 3. *Write the function that we described at the beginning of this section. Then finish the code.*

3 Data Visualization in R

This is only meant to serve as a first introduction to data visualization. There is a common saying that a picture is worth $1,000$ words. When those words are trying to describe numbers and data this statement is even more true than normal. I could ask you what the average value computed in the program we wrote today was. But that wouldn't tell me really what was happening with the data. If I really want to see what is happening, I need to see the data. A few of the basic plotting commands are built into **R**. Let us try a histogram on our data:

```
hist(X, col = "red")
```

This returns a histogram of the data stored in x and colors in the histogram red.

The function "hist" accepts many different arguments, many of which are optional. At this point in the course I will no longer provide exact syntax on every single function I define. Students should now know enough programming to be able to do a google search for an **R** function, find an appropriate web-page, and figure out how the function works. For example, search for "help hist R" the first link will be documentation showing exactly how the function works, including the optional arguments.

Now suppose I ask the (completely natural) question: How does the number of points I select affect how small I expect the minimum value to be?

Discussion 4. *Write an **R** script that runs the program for the number of points* $4, \ldots 99$ *and stores the average value in a list. Then plot the result in a scatter plot*

The scatter-plot is perhaps the most common data visualization tool because it shows how two different variables relate to each other. If my averages are stored in the list "Smallest" and the number of points I selected before computing the average is stored in the list "Num_Points" , the following code will create a scatter plot of the data: And, again, there are many different

```
plot(Smallest,Num_Points)
```

possible parameters we can provide to alter our plot. Searching for "help plot R" will reveal exactly all the things this function can do.