**Lab Assignment 05**

**I. Objectives**: Based on our previous exploration of sampling distributions, this lab will explore consequences of the Law of Large Numbers and the Central Limit Theorem. In addition, students will use simulation to generate confidence intervals. Students will practice downloading a data set and loading and running an R script.

**II. Datasets**: “hurricanes.csv”

**III. Packages**: none

**IV. Preparation**

1) Open RStudio by double-clicking the icon or selecting RStudio from the Windows Start menu.

2) Clear data from memory: > rm(list=ls())

3) To change R’s working directory, you will need to fill in the directory name *in quotes* inside the parentheses of this command: > setwd()

4) Download data: “hurricanes.csv” and place it in your working directory

5) Download R script “POLS 6480 Lab 05.R” and place it in your working directory.

6) Open the R script by typing *Ctrl*+*O* or by clicking on File in the upper-left corner, using the dropdown menu, and navigating to the script.

**V. Instructions for Lab 05**

You will perform two simulations in this lab, using a dataset on tropical cyclones in the Atlantic basin over 85 years, from 1930 to 2014. After examining (both numerically and visually) the distribution of a variable, you will choose random samples of values of that variable. As the size of the sample grows, your precision in estimating population means (i.e., the expected number of tropical storms and hurricanes in a year) should improve.

Revise line 2 of the R script to open the dataset and name it. You can view all the data as a spreadsheet by typing View(hurricanes), or you can type head(hurricanes), which will show you the first five lines of data in the **Console** window. Make the dataset active by typing:

> attach(hurricanes)

**A. Tropical Storms**

1. Find the population mean and population standard deviation for the number of tropical storms.

> mean(TStorms) -> mu

> sd(TStorms) -> sigma

Examine the population distribution of tropical storms:

> hist(TStorms, breaks=25)

Although the number of storms is a count, its distribution appears to be roughly Normal, and we will treat it as if it is. For the rest of the lab, it is easiest to understand the process by imagining the counterfactual: suppose it would be inefficient to collect the entire population of data.

2. Next, we will draw 20 small samples from the population. Run lines 5–9 of the script.

> set.seed(87654321)

> d <- seq(1, 20, length = 20)

> m <- numeric(length(d))

> for(i in 1:length(d)) {

m[i] <- mean(sample(TStorms, size=4, replace=T)) }

The first line sets the pseudo random number generator. Recall the importance of this step: a simulation will be perfectly repeatable only if it uses the same random numbers, so we tell R to begin at a particular number. The second and third lines create blank vectors that will be filled in with the sample number (from 1 to 20) and the sample mean. The last line uses the built-in sample command, which tells R to draw 4 values of TStorms.

Plot the distribution of these samples; does it follow a bell-curve centered at approximately the mean value of TStorms? For instance, you might type hist(m, breaks = pretty)

Now turn your attention to the summary statistics. If the sampling process is unbiased, then:

a) the mean of the samples should approximate the population mean; > mean(m) =\_\_\_\_\_

b) the standard deviation of the sampling distribution for samples of size *n* should equal the population standard deviation divided by the square root of *n*; > sd(m) =\_\_\_\_\_

3. In lecture, you are learning about confidence intervals. There are two ways to approach this topic; the first is to recognize that in 95% of samples, the sample mean should belong to an interval that is centered at the population mean, and extends to *mu* +1.96 *s.e.* in the positive direction, and extends to *mu* −1.96 *s.e.* in the negative direction.

Run lines 11–13 in the R script to calculate the population mean, population standard deviation, and standard error. Next, run lines 14–15 to generate upper and lower bounds for the interval. Out of 20 samples, 19 should have generated sample means that are inside the interval (i.e., less than *upper* and greater than *lower*), and 1 should have generated a sample mean outside the interval.

There is a slightly different way to perform this analysis, using the z-statistic. Run lines 16–18 in the R script to create a function for the z-statistic and to generate z-statistics for each sample mean. Examine the z-statistics (you can just type z in the **Console** window); are 19 of the values between −1.96 and +1.96?

To visualize the results of the simulation, plot a histogram of sample means or z-statistics using the hist() command.

4. The second way to approach confidence intervals is to state that for 95% of samples, the interval centered at the sample mean, and *extending* *roughly double* the standard error in either direction, should contain the true population mean.

Run lines 20–21 in the R script to generate upper and lower bounds for each interval. Next, run lines 22–25 in the R script to plot the intervals as well as a red, horizontal line set at the population mean. Out of 20 samples, 19 should have generated intervals that contain the population mean, and 1 should have generated an interval that excludes the population mean.

5. Samples of size 4 are quite small, especially given that there are 85 observations. Repeat the process above for samples of size 25 – which should lead to much smaller standard errors and therefore much narrower intervals. You will need to revise and re-run the code in lines 8–9 (which will replace the values of m automatically), and revise and re-run the code in line 13 to reflect the larger sample sizes. Run lines 20–25 to plot the confidence intervals, which should have gotten much narrower, indicating a greater level of precision in estimating. Check that the mean of sample means is close to the population mean (mu) and that the standard error is close to one-fifth of the population standard deviation (sigma)

**B. Hurricanes**

Although you might be tempted to clear the Environment by typing rm(list=ls()), we will continue using the hurricanes.csv data in what follows. However, we will want to replace most of the values that we already estimated. If you use the rm() command and insert the names of things you wish to drop inside the parentheses, you can clean up the Environment panel but keep the hurricanes data loaded. See line 27 of the R script.

6. Find the population mean and population standard deviation for the number of hurricanes.

> mean(Hurricanes)

> sd(Hurricanes)

Examine the population distribution of hurricanes:

> hist(Hurricanes, breaks=14)

Considering that the number of hurricanes is a count, it might follow a Poisson distribution. Recall that the Poisson distribution has a single parameter, which can be denoted lambda, which equals the expected number of occurrences (hurricanes, in this case) in a discrete time period (a year, in this case) > lambda <- mean(Hurricanes)

Run lines 30–33 in the R script to generate a Poisson distribution and plot it. With an average of just over 6 hurricanes per year (512 hurricanes in 85 years), the Poisson distribution suggests that in 60% of years, we should observe between 4 and 7 hurricanes (type p[4]+p[5]+p[6]+p[7] in the **Console** window and hit Enter).

If you wish to compare the Poisson and Normal distributions, you may run lines 35–36; if you wish to add a Normal curve to the histogram, you may run line 37.

7. As before, to experiment with sampling, we will draw 20 small samples from the population. Re-run lines 39–43 of the script, which have changed the variable name to Hurricanes:

> set.seed(87654321)

> d <- seq(1, 20, length = 20)

> m <- numeric(length(d))

> for(i in 1:length(d)) {

m[i] <- mean(sample(Hurricanes, size=4, replace=T))}

The first line resets the pseudo random number generator. The second and third lines create blank vectors that will be filled in with the sample number (from 1 to 20) and the sample mean. The last line uses the built-in sample command, which tells R to draw 4 values of Hurricanes.

Test your understanding by answering each of the following three questions:

For each sample, what do you expect the mean of *x* to equal? \_\_\_\_\_\_

What do you expect the mean of sample means to equal? \_\_\_\_\_\_

What do you expect the standard deviation of the sample means to equal? \_\_\_\_\_\_\_

Now type mean(m)to find out if the sampling is unbiased, and type sd(m) to find out if the standard deviation of the sampling distribution equals half of the population standard deviation.

8. Recall that 19 of 20 samples should yield sample means that are no more than 1.96 standard errors away from the population mean. Run lines 45–47 of the script to calculate the upper and lower bounds (where the standard error equals the standard deviation ÷ √4) and to plot all the sample means plus boundary lines.

9. A sample of size four is not very large, so repeat the process with samples of size 25:

> for(i in 1:length(d)) {

m[i] <- mean(sample(Hurricanes, size=25, replace=T))}

The above line of code is included in the R script as line 49.

Check whether the mean of the samples is close to 6 and the standard deviation of the samples is close to 0.52 (which equals sd(Hurricanes) ÷ 5). You can also plot the distribution of the sample means against a Normal distribution using the following code:

> hist(m, breaks=10, prob=T) +

curve(dnorm(x, mean=6.03, sd=.517), add=T, col="blue")

10. Finally, recall the second approach to confidence intervals: assuming unbiased sampling, the interval extending from 1.96 standard errors below the sample mean to 1.96 standard errors above the sample mean should contain the true population mean in in 95% of samples (or 19 out of our 20 samples). Run lines 50–51 of the R script to calculate the upper and lower bounds for each of the 20 samples, then run lines 52–55 together to plot the confidence intervals and a red, horizontal line indicating the population mean.

The confidence intervals shown here are narrower than the interval displayed by running line 47. The reason is that standard error was based on samples of 4 (so √*n* = 2), while these standard errors are based on samples of 25 (so √*n* = 5). Larger samples create much more precision in estimating the population mean without sacrificing the rate of accuracy.

11. To clear the **Environment**, type: > rm(list=ls())

To clear the **Console** window, type Ctrl-*l*