**Lab Assignment 02**

**I. Objectives**: To explore descriptive statistics, both numerical and visual. Students will calculate means, variances, standard deviations, as well as medians and quartiles. Students also will be introduced to some of R’s graphing capabilities. Finally, students will practice acquiring a dataset two ways – using a web URL and using an R package.

**II. Datasets**: *arbuthnot.R*, *crime2005.R*

**III. Packages**: *smss*

**IV. Preparation**

Open RStudio on a lab terminal by double-clicking the icon or selecting RStudio from the Start menu in Windows. Next, clear any data in the memory (which would be listed under **Environment**) by typing… > rm(list=ls())

**IVb. Loading “Foreign” Data**

You will often be working with different data sets as you engage in analysis. To load these data sets into R, you need to follow a couple steps.

First, load the “Foreign” package, to allow R to read non-R datasets.

> Library(foreign)

Then, include the command for the relevant data file, after “read”, such as,

> data 🡨 read.dta(“stata.data.dta”)

The above command is for Stata data files (ending in “.dta”). It is important to not that for some newer Stata files in Stata 13, you will need an additional library and a different command.

> Library(readstata13) # will need to install.pacakages if its not already loaded on your computer

> data.13 🡨 read.dta13(stata.13.file.dta)

Other file extensions are, for example, .csv, .txt (use, read.table), delimited files, .sav (use, read.spss), and so on.

**V. Instructions for Lab 3**

Last week we used simulations and briefly explored the binomial distribution. In this week’s readings, you learn that the expectation of a binomial distribution is *np*, and the variance of a binomial distribution is *np*(1–*p*). Let us combine these themes.

Begin by setting the pseudo random number generator, by typing the following line:

> set.seed(1234567)

Now set up the simulation for a fair coin. Type the following three lines:

> k <- c(0, 1)

> p <- c(.5, .5)

> x <- sample(k, size=100, prob=p, replace=T)

The first line indicates the values to be sampled, which are 0 (tails) and 1 (heads). The second line indicates the probabilities. The third line uses the built-in sample command, which tells R to draw 100 values of either 0 or 1, using the probabilities specified by the second line. What do you expect the mean of *x* to equal? \_\_\_\_\_\_ What do you expect the variance of *x* to equal? \_\_\_\_\_\_

If you look in the **Environment** tab, you should see a series of values for *x*.

To find the mean value, type: > mean(x)

To find the variance, type: > var(x)

The mean of *x* in this sample equals \_\_\_\_\_\_; the variance of *x* in this sample equals \_\_\_\_\_\_.

Click next to the cursor in the **Console** window, then use the up arrow until it repeats the code for sampling 100 coin tosses. Hit Enter to run a new simulation; find the new mean of *x* (= \_\_\_\_\_\_ ) and the new variance of *x* (= \_\_\_\_\_\_ ).

You should get different values when you repeat the simulation *unless* you retyped the set.seed command. The degree of similarity between the means is a property of the sample size. We will explore this in Lab 04 when we sample from a distribution.

2. In Japan, 30% of the population have blood type O, which is the ‘universal donor’ type. Suppose we randomly selected 400 people from the Japanese population; first you should ask if this fits the requirements of a Bernoulli experiment. Once you are confident that it does, calculate the expected number who will have type O = \_\_\_\_\_\_ ; the variance of the distribution = \_\_\_\_\_\_ .

First, let us plan on 1000 simulations > n <- seq(1, 1000, length = 1000)

The reason that we have to do this is that you cannot find the variance of one solitary number!

The above line of code will create a data frame containing one variable for the trial number; the next line of code creates an empty vector for the total number in the sample with type O blood:

> o <- numeric(length(n))

Next, define the parameters: > k <- c(0,1)

> p <- c(.7, .3)

Finally, create a loop: > for(i in 1:length(n)) {

> o[i] <- sum(sample(k, size=400, prob=p, replace=T) == 1) }

The first of these two lines tells R to repeat the process *n* times (earlier, we set *n* to equal 1000). The second of these two lines tells R to sample a 0 (A, B, AB) or 1 (O type) 400 times according to the probabilities specified in *p*, and to save that value as the *i*th entry for the variable *o*.

If you look in the **Environment** tab, you should see a series of values for *o*.

To find the mean value, type: > mean(o)

To find the variance, type: > var(o)

The mean of *o* in these samples equals \_\_\_\_\_\_; the variance of *o* in sample equals \_\_\_\_\_\_. How well do these agree with your expectations?

To see a histogram, type: : > hist(o)

For a histogram overlaid with a normal curve (with mean *np* and standard deviation √*np*(1–*p*)):

> hist(o, prob=T); curve(dnorm(x, mean=120, sd=9.165), add=T)

3. Simulating data is useful, but the practical purpose in this course is to learn how to analyze data. This time we will use a web URL to load a dataset:

> source("http://www.openintro.org/stat/data/arbuthnot.R")

This opens a dataset in R. The data were collected by Dr. John Arbuthnot, an 18th century physician, who gathered baptism records for children born in London from 1629 through 1710. Arbuthnot was inspired by John Graunt, one of the first demographers, who collected death statistics. This dataset has 82 observations for three variables. To find the variable names, type:

> names(arbuthnot)

To activate the arbuthnot data frame (so that you do not have to constantly retype its name), type:

> attach(arbuthnot)

To plot the birth data for boys as a line graph, with year as the horizontal axis, type the following:

> plot(year, boys, type="l")

To plot the birth data for girls as a line graph, with year as the horizontal axis, type the following:

> plot(year, girls, type="l")

For a scatterplot of boys’ births against girls’ births, type the following

> plot(girls, boys, pch=19); abline(a=0, b=1)

The second command – after the semicolon – added a diagonal line representing the number of boys and the number of girls being equal. More boys were born than girls in every year!

4. Above, we plotted the births against time and against the other gender. Another way of summarizing data is to examine the numerical properties of that data. To find the five-number summary (minimum, first quartile, median, third quartile, and maximum), you can type the following: > summary(boys)

> summary(girls)

The summary command also provides the mean, but hold off on that for a moment.

5. Suppose you wanted to graphically display the five-number summaries. The box-and-whisker plot (which you saw in Lab 01) does precisely this. The box represents the first- and third-quartiles, with a line through the box representing the median; the whiskers extend to the minimum and maximum values (unless there are outliers; more on that another time). Type:

> boxplot(girls, boys, horizontal=TRUE)

Examining the data visually provides some information that may have been missed by looking only at numerical summaries, for instance that the lower whisker extends farther from the box than the upper whisker, indicating that the data are skewed. It is not as hard to quickly and dramatically reduce the number of births – the Great Plague struck London in 1665-1666, for example – as it is to quickly and dramatically increase the number of births.

You can add horizontal lines drawn at the median for the line plots you graphed above by adding a little code: > plot(year, girls, type="l"); abline(a=median(girls), b=0)

> plot(year, boys, type="l"); abline(a=median(boys), b=0)

6. Usually, we use means, variances, and standard deviations in place of medians and quartiles. To find these summary statistics, you typically type lines separately. For example, for girls you would type: > mu = mean(girls)

> var(girls)

> sigma = sd(girls)

One way to examine the normality of a variable is to plot its histogram against a normal curve. (We did this above for the 1000 simulations of blood types.) You can type the following code:

> hist(girls, prob=T); curve(dnorm(x, mean=mu, sd=sigma), add=T)

If you wanted to plot the histogram against the normal distribution for boys, you would need to re-define *mu* and *sigma* (or create other names for the boys’ mean and boys’ standard deviation).

Above, we used a dataset that was available online (from the OpenIntro Stats website), but sometimes scholars or textbooks will make their data available as a package. To put that another way, some packages contain tools for carrying out analyses or displaying data, while others contain data. We’ll use the data from Agresti and Finlay’s *Statistical Methods for the Social Sciences* in the rest of this lab.

7. Most of what follows can be done by clicking on the **Packages** tab, clicking in the search area and typing smss, installing the package, and then both (a) clicking in the box to load the package, and (b) clicking on the smss link itself to see what the package contains. If you click on the link for crime2005, it will take you to a brief codebook.

To install the package type > install.packages(smss)

To load the package type: > library(smss)

To load a dataset type: > data(crime2005)

To make the dataset active type: > attach(crime2005)

8. Find the average crime rate across the 50 states (plus the District of Columbia) in 2005. \_\_\_\_\_\_

Find the standard deviation of states’ violent crime rates in 2005. \_\_\_\_\_\_

Now create a histogram plotting crime rates against the normal distribution.[[1]](#endnote-1) Does the distribution of actual crime rates follow a bell curve? Why not?

9. To this point, we have examined the crime rate in isolation; a second form of descriptive statistics examines the association between multiple variables. As before, we can have both numerical summaries and visual summaries. The primary numerical summary is correlation – more specifically, Pearson’s product-moment correlation, or Pearson’s *r*. This is a number that ranges between –1 and +1, with 0 representing no association. To find the correlation between a state’s violent crime rate and a state’s percent of the population below the poverty line, type: > cor(PO, VI)

10. A scatterplot will provide a visual representation of the data; to plot states’ violent crime rates against their percent population below the poverty line, type:

> plot(PO, VI, pch=19)

Do there appear to be any cases that deviates from the pattern? In POLS 6481, you will spend an entire week discussing “outliers.” For now, you should know that the District of Columbia does not fit the pattern very well, which gives us an opportunity to learn some new code.

To re-run the analyses (correlation and scatterplot, respectively) without the District of Columbia, it may be easiest to create a data frame that excludes DC. To do so, type:

> fifty <- subset(crime2005, STATE != "DC")

> detach(crime2005)

> attach(fifty)

Now you have created a new data frame, detached the old data frame with 51 observations, and attached the new data frame with 50 observations. To re-run the correlation and scatterplot, type:

> cor(PO, VI)

> plot(PO, VI, pch=19)

Has your impression of the relationship between poverty and crime rates changed? If so, then the District of Columbia may rightly be called an “influential case.”

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

1. > m = mean(VI)

   > s = sd(VI)

   > hist(VI, prob=T); curve(dnorm(x, mean=m, sd=s), add=T) [↑](#endnote-ref-1)