171:161: Introduction to Biostatistics
Breheny

Lab #2

In Lab #2, we are going to be more hands-on with the programming to get you familiar with writing and interpretting simple code. Most datasets are very large and it is convenient to analyze them with computer programs such as `R` and SAS. Today's lab will walk you through loading datasets into the computer program and teach you to perform simple calculations (Refer back to Lab 1 to help you remember how to import datasets).

1 PRACTICING SIMPLE ARITHMETIC

Let's begin by practicing coding simple arithmetic in `R`. This is very similar to a calculator:

```
> (5^2)*(10-8)/3 + 1
[1] 17.66667
```

One helpful advantage over that provided by simple calculators is the ability to store the result of calculations so that you can use them again:

```
> x <- (5^2)*(10-8)/3
> x
[1] 16.66667
> x + 1
[1] 17.66667
> n <- 50
> x/n
[1] 0.3333333
```

`R` is built on *functions*. For example, one can list the numbers from 1 to 9 with `1:9`. One can then calculate things such as the mean, median, and standard deviation with built-in functions:

```
> x <- 1:9
> mean(x)
[1] 5
> median(x)
[1] 5
> sd(x)
[1] 2.738613
```

You can collect numbers into a list with the function `c`; `R` lets you do all sorts of things with lists of numbers. For example,

```
> x <- c(2,6,8,12,20,-5)
> min(x)
[1] -5
> x^2
[1]   4  36  64 144 400  25
> sum(x^2)
[1] 673
> x/100
[1]  0.02  0.06  0.08  0.12  0.20 -0.05
```

To get more information about any function in `R`, just type the function with a question mark in front of it. To search for a keyword among `R` functions, use two question marks. For example:

```
> ?mean
> ??regression
```

Over time in this course, we'll see a number of functions in `R` and how they are used.

2 Working with Data and Importing Datasets

`R` can also be used to read in and work with data sets. All of our class data sets can be read into `R` with `read.delim`. For example:

```
women <- read.delim("http://myweb.uiowa.edu/pbreheny/161/data/women.txt")
women <- read.delim(file = "H:\\Intro to Biostats\\women.txt")
women <- read.delim(file = "H:\\Intro to Biostats/women.txt",
                    header = T)
```

All three of these are valid ways to read in a dataset, however the correct file path must be provided.

Once you read in the data set, you can see what it looks like by simply typing in `women`, or `head(women)` to just look at the first few rows:

```
> women
> head(women)
> View(women)
```

Note that `R`, unlike SAS, is case-sensitive, meaning that in `R`, `Women` and `women` are two different things. At this point, you may wish to type `attach(women)`, which tells `R` that you are working with the women data, and it should look there for variables. For example, there is a variable called `Height` in the `women` data set. In general, every time you wanted to access this variable, you would need to tell `R` where it is located; if you attach the data, you can save this extra typing. To illustrate:

```
> mean(women$Height)
[1] 59.53161
> mean(Height)
Error in mean(Height) : object 'Height' not found
> attach(women)
> mean(Height)
[1] 59.53161
```

NOTE: Don't forget to use `detach(women)` when you're done working with the dataset.

The expression `women$Height` means 'the variable `Height`, located in the data set `women`. The average height of a women was 59.5 inches. If we try to calculate that mean directly with `mean(Height)`, `R` gives us an error message that it can't find a variable called `Height`. If we attach the data set, then we're telling `R` to always look there if there's something it can't find. Once we do this, `mean(Height)` works because `R` knows where to look in order to find the variable.

3 Entering Data Into SAS Manually

Small Datasets

Sometimes researchers will work with small datasets. Instead of creating a new spreadsheet or file and importing it into SAS, it is easier (and sometimes more beneficial) to code the dataset directly into SAS. Let's look at the example from the first lab:

```
data temp;
input x y $;
/* the $ indicates y is a character variable */
datalines;
2 a
5 y
8 k
;
run;
```

Remember: **temp** is the name of the dataset (you can call it anything), **input** are the columns in the dataset where if a $ follows it is a character column, and **datalines** is the actual data.

We'll look at this data set in more detail later in the course; for today, the main goal of lab is simply to obtain basic familiarity with `R` and SAS, and some of the basic ideas involved in working with them.