Math 5305 Lab 0

1. This problem shows how we can use R to simulate data from any distribution. The variables defined below could represent math and verbal SAT scores, both with a mean of 500 and a standard deviation of 100, and with a correlation of 0.5.

   (a) Generate a vector `math` of length 1000 whose entries are normal random variables with mean 500 and standard deviation 100.

   (b) This vector can be regarded as a sample of size $n = 1000$ from the $N(500, 100^2)$ distribution. Calculate the sample mean and sample standard deviation for `math`.

   (c) Plot a histogram for `math`. Do the results from problems (1b) and (1c) match your expectations?

   (d) Calculate `length(math)`. What does this represent?

   (e) Create a vector ϵ of length 1000 whose entries are normal random variables with mean 0 and standard deviation $100\sqrt{1-(0.5)^2}$.

   (f) Define a vector `verbal` using the equation

   $$\texttt{verbal} = 250 + (0.5)\frac{100}{100}\texttt{math} + \epsilon$$

   (g) Find the sample mean and standard deviation for `verbal`, and plot its histogram.

   (h) Plot a scatterplot of `verbal` vs. `math` (This means `verbal` is on the y-axis, and `math` is on the x-axis, so you should use the command `plot(math,verbal)`.)

2. This problem illustrates a powerful task that R can perform called indexing. Let's say we would like to find the average verbal SAT score for students whose math SAT score is above 700. The following steps show how to accomplish this.

   (a) First, we create an index vector using the following command:

   ```
   index = (math > 700)
   ```

   (b) Look at the entries in the vector `index`. Notice that they are TRUE and FALSE values, that is, `index` is a *Boolean* vector.

   (c) Using the command `cbind(math,index)`, we can bind the vectors math and `index` together to form the columns of a 1000×2 matrix. Look over the values in this matrix. When is `index` TRUE, and when is it FALSE?

   (d) Note that you can add up all the values in `index` using the `sum` command, which treats TRUE as 1 and FALSE as 0. Calculate `sum(index)`. What does it represent?

   (e) For any vector $X = (X_1, \ldots, X_n)'$, the sample mean of the entries of X is just their sum divided by n.

   $$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

   In R's notation, this is `mean(X) = sum(X)/length(X)`. Compute `mean(index)`. What does it represent?

(f) Once we have a Boolean index vector, we can input it into a vector. For instance, generate the scatterplot `plot(math[index],verbal[index])`. What does `math[index]` represent? What does `verbal[index]` represent?

Another plot that makes this even more clear is `plot(math,verbal,col=(index+1))`. Here, we are plotting all of the original math/verbal SAT scores. The option `col=(index+1)` treats FALSE as 0 and TRUE as 1, so the color for FALSE points will be $0+1 = 1 =$ black, and the color for TRUE points is $1 + 1 = 2 =$ red.

(g) What is the average verbal SAT score for students with a math SAT score above 700?

(h) Set up a new index vector to determine the percentage of math SAT scores between 400 and 600. Is this about what you expect to see? Repeat for the range from 300 to 700 and 200 to 800.

(i) What percentage of students have a math SAT score higher than their verbal SAT score?

3. The command `U = 1:100` will generate the vector $U = (1, 2, 3, \ldots, 100)$. Let's say we wanted to construct the vector $V = (6, 7, 8, \ldots, 105)$, using the formula $V_i = U_i + 5$. There are two ways to do this:

- *Vectorization*. The command `V = U+5` will directly generate V from U.
- *Using a For Loop*.

```
V=NULL  #This just creates a variable V to store values in.

for(i in 1:100){
    V[i] = U[i]+5
    }
```

The advantage of vectorization is it is more computationally efficient than doing a loop. However, there are some problems that are not easy/possible to do with vectorization and therefore require loops.

(a) Let x by the `math` vector from problem (1), and let $\overline{x}$ be its sample mean.

(b) Define the vector D by

$$D_i = (x_i - \overline{x})^2$$

using vectorization, and then do it using a loop. Compare your results to make sure you are getting the same vectors with both methods.

(c) Recall that the sample standard deviation is equal to

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}}$$

Use D to calculate the standard deviation of x, and compare that to the value obtained from `sd(math)`.

4. An important skill in R is writing your own functions, for example

```
hypotenuse.length=function(a,b){
  c=sqrt(a^2 + b^2)
  return(c)
}

hypotenuse.length(3,4)
hypotenuse.length(5,12)
```

We can even write a function that will accept an entire vector, matrix, or other object as an input, such as

```
my.sample.mean=function(X){
    n=length(X)
    X.bar = sum(X)/n
    return(X.bar)
}
```

(a) Write a function called `cone.volume` that that accepts the radius `r` and height `h` of a cone as inputs and returns the volume of the cone as output.

(b) Write a function called `my.sd` that accepts a vector `X` as input and returns the sample standard deviation of the entries in `X` as output. It is possible to do this without using any loops.

(c) Write a function called `trace` that accepts a square matrix `A` as input and returns its trace as output.