Stats 195/CME 195 HW#1 Due April 18, 2016

## Instructions:

- 1. Upload R script to Canvas using the filename <your SUNetid>stats195hw1.R). E.g., I would name my submission "hgm7stats195hw1.R".
- 2. Name your functions/variables exactly as indicated below (e.g., "my.mean" for question (a)).
- 3. Check that your source code loads into memory without throwing errors by using source. E.g., I would ensure there are no errors when I run source(hgm7stats195hw1.R).
  - 1. Write a function my.mean to compute the average of a given vector of numbers. The function should work with vectors of any length. The average of  $x_1, \ldots, x_n$  is  $\frac{1}{n} \sum_{i=1}^{n} x_i$ .

```
my.mean <- function(x) {
... # fill in computation here
return( ... ) # return the result here
}</pre>
```

You can check your work by comparing my.mean against R's mean function mean.

2. Write a function my.var to estimate the variance of a given vector of numbers. The function should work with vectors of any length.

```
my.var <- function(x) {
... # fill in computation here
return( ... ) # return the result here
}</pre>
```

Given a set of numbers  $x_1, ..., x_n$ , you can estimate the variance as  $\frac{1}{n-1} \sum_{1}^{n} (x_i - \bar{x})^2$ , where  $\bar{x}$  is the average of x. Use my mean to compute the average of x.

You can check your work by comparing my.var against R's sample variance function var.

3. Write a function my.cor to compute the correlation coefficient of two given vectors. You can assume both vectors have the same length, but your function should work whatever this shared/common length is.

```
my.cor <- function(x,y) {
    ... # fill in computation here
return( ... ) # return the result here
}</pre>
```

Compute the correlation coefficient between  $x_1, \ldots, x_n$  and  $y_1, \ldots, y_n$  as  $\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{(n-1)\sqrt{var(x)var(y)}}$ , where  $\bar{x}$  and  $\bar{y}$  are the averages of x and y, respectively, and var(x) and var(y) are variance estimates or x and y, respectively. Use your functions my.mean and my.var to compute these quantities.

You can check your work by comparing my.cor against R's Pearson correlation coefficient cor.

Of course, your functions "my.mean", "my.var", etc., should not call the corresponding R functions.

4. A conjecture states the following about natural numbers. Take any natural number n. If it is even, divide it by 2; if it is odd, multiply it by 3 and add 1. Repeat this step on the result. The conjecture is that continuing in this fashion, you will reach 1. Write a function collatz that, given a number n, checks whether the conjecture holds for n, printing out all the intermediate steps. Sample output for n = 89 is:

> collatz(89) [1] 89 [1] 268 [1] 134 [1] 67 [1] 202 [1] 101 [1] 304 [1] 152 [1] 76 [1] 38 [1] 19 [1] 58 [1] 29 [1] 88 [1] 44 [1] 22 [1] 11 [1] 34 [1] 17 [1] 52 [1] 26 [1] 13 [1] 40 [1] 20 [1] 10 [1] 5 [1] 16 [1] 8 [1] 4 [1] 2 [1] 1

5. Extra credit. Use rnorm to generate two sets of 50 observations from a standard normal distribution (i.e., you should run rnorm twice, ending up with two vectors of length 50). Use

my.cor to compute the correlation. What is it, and what do you expect it to be? (You don't need to record your answer to this last sentence.)

Repeat this procedure 100 times, recording the resulting correlation each time. That is, generate a pair of normally distributed vectors 100 times, each time computing their correlation and saving the result somewhere. So at the end you should have 100 correlations recorded. What proportion of these correlations is greater than .3 or less than -.3? If you used a loop in your answer to this problem (e), you should also attempt a solution without using any loop structures.