Penn State STAT 540
Homework #1, due Wednesday, September 16, 2015

What you have to submit in an Angel submission folder: (i) Your `R` code in a file titled PSUemailidHW1.R (e.g. muh10HW1.R), (ii) pdf file that contains a clear writeup for the questions below named PSUemailidHW1.pdf (e.g. muh10HW1.pdf), (iii) Your `python` code in a file titled PSUemailidHW1.py (e.g. muh10HW1.py). Note that your R and python code should be readily usable, without any modifications.

1. Computational complexity study through Monte Carlo: Compare the *average* computational cost of two different sorting algorithms: quicksort versus shellsort. Here you are averaging the costs with respect to all possible (equally likely) orderings of a vector. You need to compare how the computational cost scales as the length of the vector, n, increases. Explore computational cost for n at least up to 200 million (you are welcome to go beyond if you are able to handle the computation). Useful commands: `runif` for generating U(0,1) random variates (the numbers to be sorted), the `sort` command with the argument `method` set to "quick" or "shell", and `system.time` to determine the run time.

   (a) Use Monte Carlo to approximate the computational cost as a function of n. Make sure you write a note (even 1 sentence will suffice) that explains how you determined an appropriate M, the number of replicates for which you will average computational costs. You should use Monte Carlo standard errors of your estimates in order to determine M. M may change for different values of n. Make sure that these errors are comparable across n and the two algorithms.

   (b) Report your estimates along with Monte Carlo standard errors in a table. You will need to determine which values of n to report.

   (c) Plot computational cost versus n. Again, you will have to make some decisions about how best to summarize your results.

   (d) Based on your study, what are your conclusions? Summarize this in just a few sentences, preferably an enumerated list.

   (e) You should look up the analytical results on run times for these algorithms. Please comment on how your results based on simulation compare to the analytical results. In particular, does the computational cost scale in the same way as the analytical results as n gets very large? If not, why not? (Revisit lecture notes about why computational costs are often complicated.)

2. Python introductory exercise:

   (a) Install python by following these instructions `https://sites.google.com/site/pythonbdclass/Classes/installation` Once you have installed python, follow the instructions and exercises here `https://`

`sites.google.com/site/pythonbdclass/Classes/hello-world` to gain some familiarity with python.

(b) Now write a short python program that prints out on 10 separate lines, "Iteration 1", "Iteration 2", .. "Iteration 10". You only need to submit your python code for this exercise, not the output from your code.

3. `Python`: introductory example of data processing. Make sure you provide your python code and your answer to part (b) below. Do not submit matrices or vectors! You will also find an example of python code that is very closely related to the question below here: `http://www.stat.psu.edu/~mharan/540/hwdir/pythonEg1.py`

   (a) Write a python function that has the following inputs: input file name (say inpfile), two specific columns (say col1 and col2), and output filename (say outfile). This python function should calculate the sum of squared differences between the two specified columns (col1 and col2), then write the squared differences into the output file. Test out your code on this data set, say by writing out the squared differences between column 1 and column 3: `http://www.stat.psu.edu/~mharan/540/hwdir/pyEg1.dat`

   (b) Run your python code on this much bigger data set: `http://www.stat.psu.edu/~mharan/540/hwdir/pyEg2.dat` calculating the squared differences between column 300 and column 650. Now that you have a new data set that has just one column (the squared differences), calculate the sum of squares either by using python. (Are you able to check your answer with R code?) Report the result (a scalar) in your homework.