Stat243: Problem Set 6, Due Wed. November 1

October 20, 2017

This covers Units 7 and 8, as well as a part of Units 9 and 10.
It's due **on paper** and submitted via Github at the start of class
Some general guidelines on how to present your problem set solutions:

1. Please use your Rtex/Rnw/Rmd solution from PS1, problem 4 as your template for how to format your solutions (only non-Statistics students are allowed to use R Markdown).
2. As usual, your solution should mix textual description of your solution, code, and example output. And your code should be commented.
3. Your paper submission should be the printout of the PDF produced from your Rtex/Rnw/Rmd file. Your Github submission should include the Rtex/Rnw/Rmd file, any R or bash code files containing chunks that you read into your Rtex/Rnw/Rmd file, and the final PDF.
4. Use functions as much as possible, in particular for any repeated tasks. We will grade in part based on the modularity of your code and your use of functions.
5. Please note my comments in the syllabus about when to ask for help and about working together.
6. Please give the names of any other students that you worked with on the problem set.
7. **Please note the comments in the section materials about setting time limits on your Savio jobs of no more than two hours.**

Problems

1. For this question you will read a journal article that presents a simulation study to assess a statistical method. The journal article and details of the question will be announced on Piazza shortly.
2. Using the Stack Overflow database (http://www.stat.berkeley.edu/share/paciorek/stackoverflow-2016.db), write SQL code that will determine which users have asked only R-related questions and no Python-related questions. Those of you with more experience with SQL might do this in a single query, but it's perfectly fine to create one or more views and then use those views to get the result as a subsequent query. Report how many unique such users there are based on running your query from either R or Python. The Stack Overflow SQLite database is ~ 650 MB on disk, which should be manageable on most of your laptops, but if you run into problems, you can use an SCF machine or Savio (in the latter case you'll need to install the RSQLite package).
3. With the full Wikipedia traffic data for October-December 2008 (available in */global/scratch/paciorek/wikistats_full/dated*), figure out a question to investigate with the data and

use Spark to process the data to answer the question. You should use Spark to process the dataset, but you can then use R or Python as you wish to do the subsequent analysis (which might be simply a graphical presentation or might involve fitting a statistical model). For example, given the time period available, you could consider a U.S. politics-related question, a question related to holidays since many holidays occur in the fall, a question related to seasonality since the data cover the period where the northern hemisphere enters winter and the southern hemisphere enters summer, or a question related to daily/weekly patterns. Given the information on language, you may be able to get at some sort of pattern related to culture or religion. You can also extend the Obama analysis, but it should be more than a trivial extension.

Note that I'm not expecting you to already know Python, so feel free to ask for help (and for those of you who know Python to help out) on PySpark coding questions on Piazza or in office hours. Also there are materials available from a Python workshop I gave at https://github.com/berkeley-scf/python-bootcamp-fall-2017; in particular the *python.html* file.

4. This question asks you to complete the exercise begun in section on October 16. Consider the full Wikipedia traffic data as in problem 3 but use the data in */global/scratch/paciorek/wikistats_full/dated_for_R*. It's the same data as in problem 3, but the partitions are half as big so that one can fit 24 partitions in memory on a single Savio node.

   (a) Using either *foreach* or *parSapply* (or *parLapply*) on a single Savio node, write code that, in parallel, reads in the space-delimited files and filters to only the rows that refer to pages where "Barack_Obama" appears. Collect all the results across the 960 files into a single data frame. Note that some R packages (such as *doParallel* and *dplyr*) are available in the same way as *ggplot2* (discussed in the materials from section on October 16 and a Piazza post), so you should not have to install them. You can do this either in an interactive session using *srun* or a batch job using *sbatch*. And if you use *srun*, you can run PySpark itself either interactively or as a background job. If it's taking a while and you want to run the code on, say, a quarter of the files and then assume the time scales accordingly, that's fine.

   Hints: (a) *readr::read_delim()* should be quite fast if employed effectively, (b) there are lines with fewer than 6 fields, but *read_delim()* should still work and simply issue a warning, and (c) there are lines that have quotes that should be treated as part of the text of the fields and not as separators. Also, as usual, you should test your code on a small subset interactively to make sure it works before unleashing it on the full dataset.

   Alternatively, if you want to explore parallelizing bash shell code, you should be able to do this problem without using R at all.

   (b) When I run the Spark code provided with Unit 8, it takes ~15 minutes using 96 cores to create the filtered dataset that just has the Obama-related webpage traffic using Spark. Assuming that you can achieve perfect scalability (i.e., that if you ran your code in part (a) on 4 times as many cores, it would go 4 times as fast), compare the effectiveness of using parallelized R versus using PySpark in terms of how long it takes to do the filtering. (Note that the distributed memory parallelization tutorial referred to in Unit 7 shows how one could parallelize R across multiple nodes, but I won't ask you to actually do that here.)

   (c) Unit 7 discusses the idea of prescheduling. See if it makes a difference whether tasks are dynamically allocated or statically allocated for your computation in part (a). Would you expect it to make a difference in this case?

5. Details of the Cholesky decomposition presented in Unit 9. Note that you can write your solution by hand if you prefer - just make sure it is legible and is attached to your printout at the end of the other

problems.

(a) Work out the operation count (multiplies and divides) for the Cholesky decomposition, including the constant c, not just the order, for terms involving n^3 or n^2 (e.g., $5n^3/2 + 75n^2$, not $O(n^3)$). You can ignore the square root and any additions/subtractions. You can ignore pivoting for the purpose of this problem. Remember not to count any steps that involve multiplying by 0 or 1. Compare your result to that given in the notes.

(b) Suppose I've written out the Cholesky calculation based on for loops. If I wanted to save memory, can I store the Cholesky upper triangular matrix, U, in the block of memory that is used for the original matrix as I go along, assuming I'm willing to lose the original matrix, or do I overwrite anything I need later in the calculation of the Cholesky?