Stat4DS / Homework 01

Pierpaolo Brutti

Due Thursday, October 25, 2018, 23:59 PM on Moodle

General Instructions

I expect you to upload your solutions on Moodle as a **single running** `R Markdown` file (`.rmd`) + its `html` output, named with your surnames.

You will give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Your responses must be supported by both textual explanations and the code you generate to produce your results. *Just examining your various objects in the "Environment" section of RStudio is insufficient – you must use scripted commands and functions.*

R Markdown Test

To be sure that everything is working fine, start `RStudio` and create an empty project called `HW1`. Now open a new `R Markdown` file (`File > New File > R Markdown...`); set the output to `HTML mode`, press `OK` and then click on `Knit HTML`. This should produce a web page with the knitting procedure executing the default code blocks. You can now start editing this file to produce your homework submission.

Please Notice

- For more info on `R Markdown`, check the support webpage that explains the main steps and ingredients: R Markdown from RStudio.
- For more info on how to write math formulas in LaTex: Wikibooks.
- Remember our **policy on collaboration**: *Collaboration on homework assignments with fellow students is **encouraged**. However, such collaboration should be clearly acknowledged, by listing the names of the students with whom you have had discussions concerning your solution. You may **not**, however, share written work or code after discussing a problem with others. The solutions should be written by **you**.*

Exercise 01: Randomize this...

Let $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ be $(k \times k)$ matrices. Our goal is to check whether

$$\boldsymbol{U}\,\boldsymbol{V} = \boldsymbol{W}.$$

It can be shown that naïvely making the matrix multiplication and comparing the result with $\boldsymbol{W}$ will take $\mathcal{O}(k^3)$ operations.

To improve upon this at the expense of (possibly) returning a wrong answer – but with small probability! – we want to use the following **one-step randomized algorithm**:

a. Pick a (single) random vector $\bar{\boldsymbol{z}} = (z_1, z_2, \ldots, z_k)$.
b. Compute $(\boldsymbol{U}\,\boldsymbol{V}\,\bar{\boldsymbol{z}})$ by first computing $(\boldsymbol{V}\,\bar{\boldsymbol{z}})$ and then $\boldsymbol{U}\,(\boldsymbol{V}\,\bar{\boldsymbol{z}})$.
c. Compute $\boldsymbol{W}\,\bar{\boldsymbol{z}}$.
d. If $\boldsymbol{U}\,(\boldsymbol{V}\,\bar{\boldsymbol{z}}) \neq \boldsymbol{W}\,\bar{\boldsymbol{z}}$ then conclude that $\boldsymbol{U}\,\boldsymbol{V} \neq \boldsymbol{W}$, otherwise return that $\boldsymbol{U}\,\boldsymbol{V} = \boldsymbol{W}$.

This scheme requires only 3 matrix–vector multiplications; that is, $\mathcal{O}(k^2)$ operations, but then the problem is:

What is the probability that the algorithm *wrongly* says $\boldsymbol{U}\,\boldsymbol{V} = \boldsymbol{W}$ when they are actually **not** equal?

You will (...try to...) answer this question mainly via a simulation study.

First of all, to simplify the math (trust me on this), assume the matrices and vector involved are defined over the integers modulo 2, in other words, $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$ and $\bar{\boldsymbol{z}}$ are all binary, $\{0,1\}$ objects, and the arithmetic operations $(+,-,\times,\div)$ are performed modulo 2. In programming, all you have to do is to work out the arithmetic in the usual way and then use the `mod` function to find its representation. In `R`, check the help file of the function `?'%%'` and read carefully its `Details` section for some relevant heads up.

Under this assumption, here it is the main result[1] you will have to validate via simulation.

RESULT:
If $\boldsymbol{U}\,\boldsymbol{V} \neq \boldsymbol{W}$ and **if** the k components of the test vector $\bar{\boldsymbol{z}}$ are independently and uniformly from $\{0,1\}$, then

$$\Pr(\text{Error}) = \Pr(\boldsymbol{U}\,\boldsymbol{V}\,\bar{\boldsymbol{z}} = \boldsymbol{W}\,\bar{\boldsymbol{z}}) \leqslant \frac{1}{2}.$$

1. Bearing in mind that the only random object in the previous probabilistic statement is the test vector $\bar{\boldsymbol{z}}$, do the following:
   a. Pick 4 different triplets of binary $(k \times k)$ matrices $\{\boldsymbol{U},\boldsymbol{V},\boldsymbol{W}\}$ with $k \in \{5, 50, 100, 500\}$ such that $\boldsymbol{U}\,\boldsymbol{V} \neq \boldsymbol{W}$ – you *may* want to randomize their construction too.
   b. For each of these triplets, repeat $M = 100$ times the one-step testing procedure and *approximate* the probability of error with the proportion of times (out of M) you get a wrong output.
   c. Do it again but with $M = 1000$ and $M = 10000$. For speed, check and use the `foreach` package or any other parallelization scheme.
   d. Extensively comment the results obtained in terms of speeed (in k) and error probability, and compare them with their theoretical counterparts highlighted above. Are the theoretical bounds *tight* or not? Did the parallelization help or not?
2. As we did in class talking about verifying polynomial identities, to improve on the error probability bound, we can again use the fact that the algorithm has a *one–sided error* and run the algorithm multiple times, say p, to get a *p-step randomize algorithm.*
   a. If we attempt the verification $p = 100$ times (independent from each other), how can we bound the probability of error then?
   b. As we noticed in class, an interesting related problem is to evaluate the gradual change in our confidence in the correctness of the matrix multiplication as we repeat the randomized test. Let E be the event that the matrix identity is correct, and let B be the event that the test returns that the identity is correct. How likely is E in light of B and how sensible is this result to your initial, "personal" probabilistic assumptions?
   c. Assume now that running the p-step randomize algorithm, it always returns that the identity is correct. As p varies from 1 to 50, how does the probability of E change based on this increasingly overwhelming empirical evidence?
   d. Use the `manipulate` package to iteractively show how the probability of E varies as a function of p and of your prior choices.

Exercise 02: Pick one...

Choose & solve one (and only one) of the following two exercises

Version A

You're new in Rome and you just moved into a new house. Surprise surprise, the phone is connected as in the old days! Now, you're pretty sure the phone number is `067405111`, but not as sure as you would like to be. As an experiment, you pick up the phone and dial `067405111` – don't try it at home please! – you obtain a "busy" signal.

[1] To prove it, condition on the value of any of the k components of $\bar{\boldsymbol{z}}$, say z_1, try to express the error condition w.r.t. z_1 and then just use the *Law of Total Probability*. Not trivial, but doable.

Are you now more sure of your phone number? If so, how much?

Please notice: to get a realistic answer, you will have to make a series of realistic, educated guesses on the unknowns involved in this exercise. Think carefully and do your best.

Version B

We mentioned Monty's three doors game a bit ago... read it carefully. Now imagine that during a show, the contestant had initially chosen *door 1* and, just as Monty is about to open one of the other doors, a very violent earthquake rattles the building and one of the three doors flies open.

It happens to be *door 3*, and it happens **not** to have the prize behind it. Well, since none of the rules was violated by the shaking, Monty decided to keep calm and carry on... the show must go on!

The question is obvious: should the contestant stick with *door 1*, or switch to *door 2*, or does it make any difference? (You may assume that the prize was placed randomly, that Monty does **not** know where it is, and the door flew open just because of the earthquake).

If you reallyreallyreally wanna impress me (?), setup a simulation in `R` to validate your math and comment the results.
