CMEE Masters: Computing Coursework Assessment

Assignment Objectives: To work on a series of computing/programming exercises and problems in a coherent, modular, reproducible workflow under version control.

Note that:

- *All script/code files, errors and other info mentioned below are in the weekly log/feedback files.*
- *The overall assessment will typically have significantly lesser marks than a simple weighted average of each week's points because the overall assessment is based on not just the "Computing Coursework Assessment Criteria", but also the the "Marking Criteria for Exams, Essays and Coursework". Both sets of marking criteria are in the Assessment Appendix of the online TheMulQuaBio notes and git repository.*
- *In your 1:1 post-assessment feedback session, we will discuss where you gained or lost marks, and what you could have improved further. To the extent possible, please come with questions about specific scripts based upon the overall and weekly feedback you have received. This may require you to compare your code with the solution code in many cases.*

Student's Name: Amy Solman

1 Specific feedback

1.1 The Good (what you did well!)

1. Found all the expected weekly directories in your parent directory, with reasonably clean project organization and code
2. You had an overall readme file including a description of what the overall project structure is.
3. You had weekly Readme within each week. The weekly Readmes were clear and informative.
4. Good job with the coding overall (with some caveats - see below). I noticed rapid progress over the weeks.
5. You did many extra credit Qs.
6. Scripts from Weeks 4, 5 & 6 were not part of the assessment, but you tried to keep these weeks well organized - good.
7. Nice work with the temperature Autocorrelation practical and report. I liked the fact that you took the effort to cite papers, and your interpretation and discussion were great. Compare with the solution though, which uses a different approach for the permutations. Some issues/comments: (1) You did not report the correlation coefficient and p-value!; (2) You could have plotted the histogram of random correlation coefficients along with the observed coefficient.

1.2 The Bad (errors, missing files, etc)

1. There were many syntax errors (mainly in week 7), and path specification errors (throughout). Many could have been fixed quite easily.

2. There were also, on occasion, some extra files.

1.3 The Ugly (niggling issues like commenting, cosmetics, complexity of code, etc)

1. Your Git repo size when I checked week 7 was about 251 MB — somewhat on the larger size. This suggests you might have kept binary files under VC, and that you might have committed excessively.

2. You had a .gitignore throughout, but not enough meaningful exclusions specific to certain weeks.You can fine tune the exclusions further; this might be useful: `https://www.gitignore.io`.

3. In your weekly readmes you could have included the language and dependencies requirements. Check out this resource: `https://github.com/jehna/readme-best-practices`. As you become a more experienced programmer, you will learn to make the readme file descriptions even more informative yet succinct.

4. In many places, especially early weeks (e.g., UNIX), you could have broken the description of certain complex commands or code lines into key components using a comment.

5. The code could have been more consistently well-commented. You will learn to find the right balance of clean vs commented code that works for you. Some docstrings were missing.

6. It would have been good if you had added some input checks and returns a meaningful message with error for missing input files to certain (e.g., shell) scripts. That is because these would then become true utility scripts that anybody could use. I had not asked for these explicitly, but was hoping this would be something you would arrive at yourself after gaining some experience with coding and revisiting your code. But most students don't get to this, so don't feel too bad!

7. In as many cases as possible, it is a good idea to write scripts to be self-sufficient (modularize your code). For example, align_seqs.py was nicely done, but it could have been written to be a module also take external inputs optionally (though I did not ask for it specifically). Compare with the solution.

8. Please do compare as many of your solutions with the ones I have given (e.g., Unix-Prac1.txt, using_os.py) as possible. There are simpler ways to solve some of them, especially the last one, and in general it will be insightful to see how the same code/solution can be written/found. In particular:

   (a) using_os.py: the script could have provided some more meaningful output to screen.

   (b) You did a good job with lc1.py, lc2.py, dictionary.py, and tuple.py, but if you compare with the solutions on the repo, you will notice that you could have make them produce better-formatted output.

   (c) There are more compact solutions to the pp_regress.R problem

2 Overall Assessment

You did a good job overall. I was impressed by your efforts to understand as many details of the programming languages and coding as possible.

It was a tough set of weeks, but I believe your hard work in them has given you a great start towards further training, a quantitative masters dissertation, and ultimately a career in quantitative biology!

Provisional Mark: 73

Signed: Samraat Pawar

March 8, 2020