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: Zongyi Hu

1 Specific feedback

1.1 The Good (what you did well!)

1. Found all the expected weekly directories in your parent directory.
2. Very neat, clean project organization and code.
3. Your Git repo size when I checked week 7 was about 6 MB — a small size, suggesting you did not keep unnecessary binary files under VC, and that you did not commit excessively. It could also mean that you did not commit enough, and/or somehow along the the way lost parts of your git history — but I won't check these possibilities!
4. You had an overall .gitignore, with tailored and meaningful exclusions specific to certain weeks. Good! You will likely struggle to fine tune these further as you have used sensible templates, but check out this resource anyway for some tips: `https://www.gitignore.io`.
5. You had both an overall Readme (which contained pertinent information) and weekly Readme's which gave some detail. Good, this information is critical for reproducibility!
6. Nice job with the coding overall. Good attention to detail.
7. Great job with the shell scripts - you made them more robust, providing a message if necessary inputs were not provided. Indeed, in general, it is a good idea to add some input checks and return a meaningful message with error for utility scripts like these, especially in case somebody else uses it. 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, great work!
8. Good use of loops in PP_Dists.R, but consider if you could have generalised this further by using functions?

9. Nice job on the final output from run_fmr_R.py!

10. The Autocorrelation practical code was good. You used a nicely vectorised approach, making your code very clean and easy to understand. You plotted the histogram of the permuted correlation coefficients - good. You could have also plotted the the correlation pattern.

11. Your Groupwork practicals were all in order, and your group did well in collaborating on it. More feedback on this in the 1:1 sessions.

1.2 The Bad (errors, missing files, etc)

1. You forgot to require / library ggplot2 in Girko.R. This caused an error as the ggplot() function was not loaded into R's namespace, so R didn't know what you wanted it to do!

2. You require argument for LV2.py, but have not provided defaults or an error message if these are missing. This leads to an out of index error when run without input.

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

1. Because you only had one overall .gitignore it was a little unwieldy, you should consider using inividual .gitignore files for each week, as these would allow you to be as specific and meaningful as you have, but would keep each file of a manageable size and complexity. Having an overall .gitignore is a good idea too, as you can use this to make general exclusions e.g. files over a certain size.

2. The weekly Readmes could have included more useful information. In these Readmes it is best practice to include details of the language(s) used, as well as depedencies, and the versions of all of these. Also, you should include brief descriptions of each file present, and possibly instructions on how to use each file. Check out this resource: `https://github.com/jehna/readme-best-practices`. As you become a seasoned programmer, you will learn to make the readme file descriptions even more informative yet succinct.

3. You had some extraneous files present e.g transformed.txt, these should really have been in either your results directory (which should be empty), or sandbox (which doesn't get tested by the testing script).

4. Your week 3 results folder is very cluttered, and you have one file in week 2. Ideally these should be empty.

5. In places, you could have made scripts more robust. For example, you have not added any input checking to Countlines.sh, but you have done so for most other Bash scripts.

6. The output for preallocate.R was quite cluttered / not very useful. The information you wanted to get from this script was the time taken for preallocated method vs not preallocated methods. The amount of memory used is not of much interest, as it should ultimately be the same.

7. You should take a look at your logic in using_os.py, as you repeatedly print out the same directory name (which should really have been printed just the once!). You have printed it out for the number of files in the home directory!

8. Commenting could be improved – too verbose at times, and inadequate/missing in others. It will get better with coding experience.

9. 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.

10. The occasional missing Python docstring

11. Please do compare as many of your solutions with the ones I have given (e.g., using_os.py) as possible. There are simpler ways to solve some of them, 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.

2 Overall Assessment

You did an excellent 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: 78

Signed: Samraat Pawar

March 18, 2021