Mini-Projects, detailed instructions and grading criteria

This document contains the instructions for the formalities related to the mini-projects and their presentation. For descriptions of the projects themselves, refer to separate documents available for download in the Student Portal.

Structuring your work

You will work in teams; addressing part of the data analysis needs of a given scientific application. It is important for your learning that both students participate in all parts of the project, i.e. the planning, implementation and project paper writing. We suggest that you meet frequently, with a start-up meeting as soon as possible where you:

- Decide on a timeline for completion of subtasks.
- Make sure that both of you have access to the Git repository.
- Agree on how to work with the code, is it OK to push to the master branch? Should you use feature branches, or will you work mainly with forks and Pull requests/code review.
- Assign shared responsibilities, is it possible to divide up some responsibilities on individuals, such as reading selected references and summarizing them to each other?

Then, when you feel that you have a decent understanding of the task at hand (do *not* wait until you feel confident about this before starting to "play with code"), decide how you plan to address the problem. What tools do you plan to use? What is already available and what do you need to deploy yourselves? Remember that the VMs can be configured to your liking, and you may have to install packages and libraries that you need.

Writing the report

The report should be written on the format of a short scientific paper. The length can be at most 3000 words. For inspiration on outline and structure please see the papers provided for the literature seminar. Although short, the course papers should contain an introduction to the problem, a clear statement of the problem solved, how it was solved, a discussion of the result, as well as a critical evaluation of the proposed solution. It could also contain a short discussion about potential future extensions and improvements.

Formal requirements

For a passing grade (1 point) you should:

- Hand in a written report showcasing your best efforts to solve the problem. Please hand in the report on time even if there are still parts of the solution that you have not managed to complete.

- Participate in the final project presentation.
- Provide a link to the Github/Bitbucket repository where your project code is hosted. If you prefer a private repository, you should send an invite to the teacher to view the repository.

For higher marks (2-4 points), we will look at how well the project was executed, and assess the quality of both the implemented solution and the written report.

Grading criteria

To assign points (1-4) to your project, we will focus on these aspects of the project:

- Quality of execution of the analysis and software development
- Quality of the project paper

Participation in the oral presentation is a formal requirement, but we will not grade your performance.

Quality of analysis and software development
To assess the quality of the proposed solution we will look at:

- How well does the taken approach fit the problem?
- How well can we expect the chosen approach to scale?
    - Realistically, you will not be able to make extensive runs on complete, very large datasets, but you should plan to accommodate a growing data size.

- We will look at the activity in GitHub or BitBucket, such as commit histories, issues/discussions, pull requests etc.
- How well is the code structured, annotated and usable by a third party?
    - Good solutions may not require a lot of code to be written. Attempt to use external libraries/tools whenever possible.
    - Is there some degree of automation? What is assumed about the data sources?
    - How hard is it to use or extend the code to accommodate more data than the one provided (of the same type)?

Quality of the project paper

To assess the quality of the project paper, we will look at:

- How clearly is the problem stated?
- How well is the problem contextualized?
- How well does it cover related, relevant work?

    - This is supposed to be a short project, and we do not expect an extensive bibliography. We will look at how relevant the cited related work is and if it is accurately used to support the discussion, rather than the sheer amount of references. Aim at approximately 4-5 references. You may of course cite papers from the literature seminar if they are relevant.

- How well do you argue for you proposed solution?
- How well do you demonstrate the suitability of your solution?
    - Performance benchmarks? Weak scaling properties?
    - How well are limitations discussed?

- Language: Is the paper clearly written using correct English?

Tips:

- Devote the majority of your wordcount to describing your proposed solution and the results.
- Figures can be very effective at conveying a message. Consider "concept figures" to support the description of your solution.
- Write the introductory sections early, including citing relevant work, while you are still struggling to understand the problem.
- If you divide writing parts of the text amongst yourselves, set internal deadlines to allow time to proofread each other's contributions.