Computational Physics Projects / General Instructions

For Computational Physics, you have each been assigned a group and an assignment. You should work together, and hand in the assignment as a group.

The deliverables for this assignment are:

- Two intermediate drafts of the write-up (you will not be graded on content—you just have to hand in the draft), with deadlines in the syllabus. By the first draft you should definitely have some idea of how you are going to address the questions and have an outline. By the second draft, it would be best to have the major pieces of code you need written, even if all the analysis and plots are not done. I will give you feedback on these drafts that will be useful for your final project.
- A final version of a write-up, in PDF formatted by LaTeX. This should be 5–10 pages including figures. It should be written in clear, proofread, expository English, that explains the results and addresses all the tasks and questions in the project description. It should include figures demonstrating the results. It should include references when appropriate (use BibTeX to do the citation management).
- A 15 minute presentation to be given to the class. This will occur in the week after classes end.
- The software that produced the results you present, in a GitHub repository. Your team should develop the code using this system, so you can easily coordinate. The software should be implemented as one or more Python scripts, which where appropriate imports modules that you may have created. Unless the project description specifies otherwise, use only standard Python modules (like `os`, `sys`, `argparse`, etc), plus `numpy`, `scipy`, and `matplotlib`; if there is another library you think you want to use, let me know. Each function and class should have a docstring with the inputs and outputs and other relevant information described, and other inline documentation as appropriate.

In general, the structure of the software should be:

- One or more Python scripts that can be called from the command line (with perhaps command line arguments) to perform computation and output results as data files. These scripts should control the overall flow of the computation, but should make use of modules.
- One or more modules in separate files that contain the core computational functions and classes you are using, with well-defined inputs and outputs.
- One or more files that `pytest` can be used on for unit tests.
- A separate script to create plots based on the data files.

Do not use a Jupyter notebook, and do not put the entire computation and plotting in one file! Use the files to organize your work and to make your software modular.

Some parts of the projects are experimental in the sense that I have not implemented the code before myself, and I cannot guarantee that it is possible to answer the question that I have asked. So you should definitely ask me questions if there are things that seem confusing/impossible.

The grading will be based on Completeness (i.e. all of the parts above are complete), Correctness (i.e. they seem to be done correctly), Carefulness (appropriate unit and functional tests have been done), and Clarity (of the write-up and the code documentation).