Computational Physics I (PHY4140/PHY5340)

Additional information

Lectures and labs

Attendance of the lectures is **not mandatory**. I will make my lecture slides available to you. I may show some material (e.g., run code) that is not in the slides during the lectures, but it will usually be mentioned on the slides. Also, the slides may not contain all the details of the derivations that I will do on the board.

The lab is an opportunity for you to work on your lab/homework problems and projects and get help from me if necessary. I may also spend some time discussing previous assignments. If you are confident that you understand the assignment perfectly and can do it on your own and also have no questions about the previous assignments, you may skip the lab.

Students are strongly encouraged to bring their own laptops to the lab (with compilers and other necessary software installed — see below), but there will be desktop computers available in the lab as well.

There will be 5 or 6 labs and 3-4 projects. Most labs and all projects will involve writing code to solve a particular physics problem and then investigating the system by varying different parameters, just like in a regular lab, only computationally. Some problems will also involve simple analytical derivations. Students will be able to choose between different projects.

Graduate students will have more tasks to do and will need to do one more project.

Homework requirements and grading criteria

Homework problems will be posted online and a message will be sent to you letting you know that the new problem set is available.

For all homeworks (both labs and projects), you will need to submit both a report and your code. The report can be submitted either in electronic

form (as a **pdf file**) or in paper form to the dropbox with the course number on the 2nd floor of MCD. **The code should be submitted in electronic form** — no exceptions! — I need to be able to compile and run it.

All files submitted electronically should be packaged **in a single archive** (`*.tar.gz` or `*.zip`) and emailed as an attachment to `chubynsky@gmail.com` , with PHY4140 (or PHY5340) in the subject line and your full name in the subject line or the body of the message. If the report is not included and submitted separately (as a hard copy), please mention this in the body of the message. Your code should include **detailed comments** explaining what each part of the code does and the meaning of the most important variables (unless their names are self-explanatory). Put very long descriptions (more than a few lines) in a separate `README` file and refer to them in the code. If the task requires you to produce an output file, it should be included. If you have used scripts (e.g., for processing the output of your program or for plotting), they should be included as well. All these additional files should be described in `README`. If there are very many files, you can place them in separate subdirectories, but again, everything should be sent as a single archive!

Every figure you include in your report should have a caption describing in detail what it contains, the meaning of all lines and symbols in the plots, etc. Schematic drawings can be done using software of your choice or by hand. Plots using data you have obtained cannot be done by hand, of course.

If you need to do data fitting, you can do it using software of your choice — you don't need to write fitting code yourself.

The submission deadline is 23:59 on the due date (usually Monday). There is a 20% penalty for submitting up to 12 hours late and a 50% penalty for submitting up to 7 days late. One (and only one) resubmission (in case you forgot to include something, realized you've made an error, etc.) is allowed with a 10% penalty before the deadline **only**.

The most important thing is: I should be able to compile and run your code and it should produce the results that you claim. If this is not the case, I reserve the right to give you **0 points** for the whole problem! (I may still give you some points, especially if there is a trivial bug that is easy to fix, so submit all your work anyway.) You can appeal by showing that the code (**as submitted!**) runs on your system.

There can be a penalty of up to 10% (from the total number of points for the problem) for poorly documented and/or unnecessarily complicated code that is hard to understand. There can also be an up to 10% penalty for very obviously inefficient (slow) code.

The grading will be done jointly by me and the TA, Tyler Shendruk. If you have any questions about your grade or your homeworks in general, you can talk to either me or him.

You are free to discuss the problems with me and other students, but **you should code and write your report on your own!** Using software and code fragments written by others is not allowed (with the exception of plotting and fitting software), unless stated explicitly. No math and scientific libraries other than the standard math library in C/C++ and similar libraries in other languages may be used, unless stated otherwise.

Programming languages and other software

Recommended: Fortran (77, 90, 95), C, C++
Allowed: Java, Python
Explicitly disallowed: MATLAB, Mathematica, Maple

If you want to use other languages, please talk to me beforehand.

Scripting languages (including Python) and shell scripts may be used for auxiliary tasks (e.g., for plotting and to automate running your code with different sets of parameters). Their use for main code is allowed without penalty, but discouraged.

To be able to work on the labs and projects on your own computer, you will need language compilers. If you work under Unix (including Linux), you will have at least C and Fortran compilers installed. Under Windows, I suggest installing **Cygwin** (`cygwin.com`), which is a Unix-like environment including, in particular, the `gcc` compiler collection (Fortran, C, C++, Java, ...) On a Mac, it is apparently possible to install the compilers by installing Xcode and Xcode command tools.

For plotting and data fitting, under Unix and Cygwin, `gnuplot` and `xmgrace` can be used. Another option is `matplotlib` library for Python (`matplotlib.org`). For typing the reports, LaTeX is recommended. If you

need to do a schematic drawing, you can use `xfig` or any other software you prefer or do it by hand.

Textbooks and other materials

There is no required textbook. While there are several good textbooks (some of which are listed below and some of which are available online for free, at least on campus), most of the material covered in the course can be easily found online in various lecture notes, slides, etc.

Note that there are two basic types of computational physics textbooks: those that are mostly about computational methods and only mention physics problems to illustrate the methods (methods emphasis) and those that are mostly about teaching physics with the help of numerical methods which are only introduced inasmuch as they are needed to solve the physics problems (physics emphasis). In addition, there are books specifically on numerical methods (aka "numerical analysis"), some more and some less mathematical.

A great overview of available literature, software, etc. is
R. H. Landau, *Resource Letter CP-2: Computational Physics*, Am. J. Phys. 76, 296 (2008).

An extremely useful series of reference books is
W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes*, Cambridge University Press.
There are different publication years and different languages (i.e., Numerical Recipes in C, Numerical Recipes in Fortran 77, etc.), but otherwise the books are similar. The website for the books is `http://www.nr.com/` . Older editions are available for free by chapter. New editions are available by paid subscription, but you can access a limited number of pages per month for free. This book covers nearly all methods I will be talking about and much more. It explains the methods, but without "unnecessary" mathematical rigour. Note you need to buy a licence to use the code from the book (and, of course, it is not allowed for the assignments anyway!)

Some other textbooks:

R. H. Landau, M. J. Paez, and C. C. Bordeianu, *A Survey of Computational Physics*, Princeton U. P., Princeton, 2008. Physics emphasis. A new Python

version is available online as a pdf file at
http://www.science.oregonstate.edu/~rubin/Books/eBookWorking/

N. Giordano and H. Nakanishi, Computational Physics, 2nd ed., Prentice Hall, Saddle River, 2005. Physics emphasis. TOC and sample programs at
http://www.physics.purdue.edu/~hisao/book/

H. Gould, J. Tobochnik, and W. Christian, *Introduction to Computer Simulation Methods*, 3rd ed., Addison-Wesley, Reading, 2006. Physics emphasis, Java-based. TOC and sample chapters at
http://physics.clarku.edu/sip/

T. Pang, *An Introduction to Computational Physics*, 2nd ed., Cambridge U. P., Cambridge, 2006. Methods emphasis. TOC and code at
http://www.physics.unlv.edu/~pang/cp.html

J. Franklin, *Computational Methods for Physics*, Cambridge U. P., Cambridge, 2013. Mostly methods emphasis. Available on campus and by proxy at
http://ebooks.cambridge.org/ebook.jsf?bid=CBO9781139525398

J. R. Hauser, *Numerical Methods for Nonlinear Engineering Models*, Springer, Dordrecht, 2009. Methods emphasis, uses Lua programming language. Very large (1000 pages). Available on campus at
http://link.springer.com/book/10.1007/978-1-4020-9920-5//page/1

A. Tveito, H. P. Langtangen, B. F. Nielsen, X. Cai, *Elements of Scientific Computing*, Springer, Berlin, 2010. Methods emphasis. Available on campus at
http://link.springer.com/book/10.1007/978-3-642-11299-7/page/1

B. Gustafsson, *Fundamentals of Scientific Computing*, Springer, Berlin, New York, 2011. Mostly methods emphasis. Available on campus at
http://link.springer.com/book/10.1007/978-3-642-19495-5/page/1

P. O. J. Scherer, *Computational Physics*, Springer, New York, 2013. Methods emphasis. Available on campus at
http://link.springer.com/book/10.1007/978-3-319-00401-3/page/1

S. Širca and M. Horvat, *Computational Methods for Physicists*, Springer, New York, 2012. Methods emphasis. Large (700 pages). Available on campus at
http://link.springer.com/book/10.1007/978-3-642-32478-9/page/1

J. Thijssen, *Computational Physics*, Cambridge U. P., Cambridge, New York, 2007. Methods emphasis. Mostly more advanced methods than those covered in the course.

M. Hjorth-Jensen, *Computational Physics*, lecture notes, `http://www.physics.ohio-state.edu/~ntg/6810/readings/Hjorth-Jensen_lectures2012.pdf`