PHYS 340: Computational Physics Fall 2012

Instructor: Francis Starr (fstarr@wesleyan.edu), Science Tower #227, x2044

Teaching Assistant: Paul Z. Hanakata (phanakata@wesleyan.edu)

Class: Tuesday 1:10 – 4:00 PM, Room SCI 71. Paul will run an extra help session; the time is to be determined. If you wish, you will be issued a key, so that you can access the lab independently. Obviously, use this access responsibly. If you are issued a key, you are required to turn in the key at the end of the semester to pass the course. You can always access the computers remotely via ssh to phys340.phys.wesleyan.edu. (We will discuss remote access in detail later).

Office Hours: I am on partial leave this semester. As a result, I am only at Wes on Tuesdays and Fridays. If you need help, please use either email or Paul as a first resort. If you need to meet with me personally, please schedule an appointment.

Goal: The overall goal for this course is to provide you with the basic tools you need to carry out computational research, be it academic or applied. Primarily this means understanding the use of programming languages and algorithmic methods. However, it is also very important to have a basic understanding of how to set up a computer so that it is useful as a tool for computational physics. Hence we will cover both the setup of a workstation and the use of cluster facilities. Note that these skills are not exclusive to theoretical work; often, basic computational methods are just as critical for the analysis of experimental data.

Certificate in Modeling: This course is fulfills part of the requirements to receive the Certificate in Informatics and Modeling; for more details, see `http://www.wesleyan.edu/registrar/catalog/cimt.htm`

Course Text: There is no one book that covers the material we will discuss. Some of the material is simply not in any books. However, I list a collection of books that I will reference during the class. When relevant, I will provide handouts.

1. *Numerical Methods for Physics*, by A. Garcia, Prentice Hall.
2. *Computational Physics*, by N.J. Giordano and H. Nakanishi, Prentice Hall.
3. *An Introduction to Computer Simulation Methods: Applications to Physical Systems*, by H. Gould, J. Tobochnik, W. Christian, Addison Wesley.
4. The "Bible" of numerical algorithms: *Numerical Recipes in C (or Fortran): The Art of Scientific Computing*, by W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. Unfortunately, this book is not extremely user-friendly.
5. For basic programming reference: *A Book on C* by A. Kelley and I. Pohl,

Class Structure: The course will run more like a lab course than a lecture course. As such, we will complete nearly all the work for the course during class time. The one major exception will be the end-of-class project, which will require outside effort.

Participation/Attendance: I expect all students to actively engage in class. Since this is more like a lab course, this should be unavoidable. Similarly, attendance is **mandatory!** If you must miss class, discuss it with me as soon as possible.

Background: I am assuming relatively little prior knowledge, other than basic computer skills (web-browsing, etc). We will use the Linux operating system exclusively. I will start with very basic instruction on how to use the Linux command line. For those of you who are already very familiar with Linux, please be patient at the start of the semester, as we need to establish a level playing field. Mathematically, I expect that you have been exposed to integral calculus, differential equations, and linear algebra. I do not expect that you have taken formal courses in all these subjects, but that you at least have worked with them previously.

Note that Mac OS X is based on BSD UNIX. As a result, you can access a command-line interface on any Mac and use it in the same way we use our Linux workstations.

Computational Tools: This is what the course is all about. We will setup Linux machines, connect with the university cluster, develop skills in the C programming language, and finally learn specific algorithmic methods that apply all these skills to real problems.

Final Project: Each student will complete a final project at the end of the semester.

Grading: Grading will be split into 2 components: (i) in-class assignments (65%) and (ii) final project (35%). Success on assignments is very well defined, so the work and you should do well. Obviously, the final project will be important to your grade.

Schedule: The schedule below will be altered if needed. If there are special topics the class wants to cover, these can potentially be included.

Week 1: Installing the Linux Operating System
Week 2: Basics of Hardware, Linux, and Networks
Week 3: The C programming language
Week 4: Numerical differentiation and integration; application to radioactive decay
Week 5: Root finding
Week 6: Integrating equations of motion: Oscillator motion
Week 7-8: Integrating equations of motion: N-body molecular dynamics
Week 9: Boundary Value problems: Quantum mechanical applications
Week 10: Monte Carlo methods: The Potts model
Week 11: Minimization methods
Week 12: Final Project
Week 13: Final Project

Finally: I will not give an incomplete for the course, except under the most extraordinary of circumstances.