Physics 615 / Eng. Physics 615 (Numerical and Computational Methods in Physics) and Astronomy 815/ Physics 815 / Chemistry 914 (Computational Methods in Physical Sciences)

Course Information ***Spring 2022***

Welcome! Scientific computation is both an art and a science. Let's try and unleash your creativity, productivity and physical insight.

Web Page
https://grahamwwilson.github.io/815-615-Home/

Location The assigned class-room is Malott 2005. We also have access to the department's computer lab in room 2076. This room is equipped with a number of desktop computers running Linux (Ubuntu 18.04 LTS) each with two monitors. I will distribute PINs for student access to the room using the OmniLock electronic door lock. We will sometimes meet during the class period in that room. At the time of writing, this door lock is not working, but should be repaired early in the first week of class.

Prerequisites
For 615, the prereqs are listed as PHSX313, MATH320 and EECS138.
For 815/914, six hours of computer science courses numbered 300 or above, and six hours of physics and/or astronomy courses numbered 300 or above.

The expectation is that you already have, or are able to ***quickly acquire***, reasonable proficiency in scientific programming. Skills and knowledge acquired from upper-level undergraduate and graduate courses in physical sciences and corresponding math courses should facilitate understanding of particular problems and projects leading to a better appreciation of their importance.

Course Goals:

The primary goal is to develop your ability to solve physical problems correctly using computational methods. It is **not** a computer science course designed to teach you how to program, **nor** is it a course primarily on numerical analysis. But we will be spending time on training you to more effectively use scientific computing including its necessary tools such as version management (git), high performance computing (HPC), and scientific report writing with LaTeX. The emphasis will be on developing, adapting, and exploring numerical solutions to physical problems. In many ways, the class will be a "lab" type course rather than a lecture, where the experiment is designed, carried out, redesigned etc on your computer, and the project reports and accompanying code are the primary method of assessment.

According to Paul DeVries,

Physics 615 / Eng. Physics 615 (Numerical and Computational Methods in Physics) and Astronomy 815/ Physics 815 / Chemistry 914 (Computational Methods in Physical Sciences)

Course Information *Spring 2022*

"Computational physics is now widely accepted as a third, equally valid complement to the traditional experimental and theoretical approaches to physics", and the course should give you a better appreciation of this.

Acquiring an ability to excel in computational methods is a key aspect to succeeding in research in a variety of sub-fields which span experiment and theory. So, one of my main goals is to develop your ability to do research, and facilitate you developing skills which will make you more employable. Synergistic with this, is the fact that the research skills requirement of the Ph.D. is satisfied by this class.

Instructor: Graham Wilson. Regular office hours will be on Zoom. Please be prepared to share your screen with code examples. My zoom room is https://kansas.zoom.us/j/3879191996 . My office is Malott 6006A inside the Malott 6006 suite (If the outer door is locked please knock loudly). Office hours schedule on Zoom to be arranged. Also, you may make appointments for other times (contact me by E-mail gwwilson@ku.edu). If and when the COVID situation improves, I will be spending more time on campus; but at least in the near term, besides just after class, a zoom appointment is your best bet.

Programming Language Requirement: Before discussing textbooks which often depend on particular programming languages, let's address programming language issues.
I do not require you to use a particular programming language, but I do require you to use a programming language with which you can easily demonstrate the ability to develop your own numerical algorithms. So the basic constraint is that the programming language is *algorithmic* rather than symbolic. Mathematica is symbolic. Or in object-oriented (OO)/procedural terms, I want to see implementations/subroutines and not just interfaces/library calls.

Obvious candidate programming languages are C++, python and Fortran which are well established in the scientific community. If you are already very comfortable with one of these you are likely to be off to a flying start in being able to excel in this course.

Python is now very much mainstream, with lots of associated resources, and has a relatively easy and fun learning curve.

TextBooks:
The primary resource for this course is the book "Numerical Recipes, The Art of Scientific Computing", Third Edition (2007), by W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery. The book uses C++ in its examples. See http://www.nr.com/ for more details.
The older 2nd Edition (1996) with support for F77, F90 and C can be accessed as a free resource for your personal use from http://www.nr.com/oldverswitcher.html

Physics 615 / Eng. Physics 615 (Numerical and Computational Methods in Physics) and Astronomy 815/ Physics 815 / Chemistry 914 (Computational Methods in Physical Sciences)

Course Information ***Spring 2022***

Please consider this a *resource book* and not a text book.

Highly recommended is also “Computational Physics: Problem Solving with Python”, 3rd Edition by Landau, Paez and Bordeianu. This has a python emphasis.

Highly recommended also is “Applied Computational Physics”, 1st Edition (2018) by Boudreau and Swanson. This is a great reference with an emphasis on C++ and modern techniques at a fairly advanced level.

The Press et al. textbook is also accompanied by source code for various languages and platforms, which you may want to buy a license for, or use an existing installation.
Please make sure that any time you base elements of your program on particular program elements from such resources that your project report properly acknowledges this fact.
This book is heavily weighted towards numerical algorithms rather than particular physical applications, and so is a necessary but not sufficient component of background reading. Its age augurs well for many of the bugs/errors having been shaken out over the years. It appears to do a good job in many areas, sometimes with lots of detail and beyond the scope of much of the work in this class. The F77 code is OK but not exactly what I consider to be exemplary practice and this is also espeically the case for the C++ code.

There are a number of books which I am likely to use as a resource in preparing the classes and projects which are more focused on the physics.
I also recommend the following book especially to the undergraduate students.

“Computational Physics” (2nd Edition) by Nicholas J. Giordano and Hisao Nakanishi, Prentice Hall. This is very good at putting the physics first, and addressing interesting and accessible problems.

There are also a number of books that are more focused on teaching scientific computing. A good one I found recently, is “Introduction to Python for Science and Engineering” by David Pine.

Project Reports: There will be about 6 projects over the course of the semester. Each project will be over about a 2-3 week period, and should reflect something like the expected 3:1 ratio of out-of-class hours to in-class hours (ie. around 18 hours of work per project). The project report should be well structured and written, including an introduction to the problem at hand, some more theoretical discussion with references to the problem and methods used, results, cross-checks and conclusions. It is expected that the report is written using LaTeX, which is now fairly to use, using for example Overleaf. As an appendix, the properly commented computer program should be included. The results section will almost always benefit from proper attention to making graphics presentable. Be careful to not underestimate the time needed to make timely progress towards completion of each project. It will usually take quite a bit longer than you think to actually do the cross-checks, fix bugs, etc.

Project Delivery: I prefer to see all the files that you used. A bare minimum is the program, the output, and the report. A convenient way to do this is to deliver electronically a single zip file of the directory/folder which contains your work and writeup. Preferred is if you point me to your git repository and make sure that I have access to it.

Homework / In-class Exercises: Homework and some occasional in-class exercises will be used to encourage learning/application of particular sub-topics in a manner intended to facilitate timely and thorough completion of project topics. The emphasis here is on quick assessment of whether you have mastered the concepts.

Programming Language Choice Considerations: There are a variety of issues which need to be taken into consideration. I have put together a long and growing list in a separate hand-out/document.

Exams : Unless you all object there will be no formal exam component to this class.

Grading: Projects and Homework **: 100%**

Department information and policies regarding tutoring, academic misconduct, harassment and other topics can be found on a separate handout. You should read this handout, as the statements in there will apply to this course.

Course Information Disclaimer:
This information may be updated during the course of the semester, and any changes will be announced in class.