Physics 530: Introduction to Scientific Computing

Course Description

J.R. Gladden
Department of Physics and Astronomy
University of Mississippi

Course Description and Motivation

This is a course targeted toward undergraduate and graduate students in the physical sciences (Physics, Chemistry, Biology, Geology, Engineering, ...) and mathematics. The purpose of the course is to provide these students with a coherent picture of the role of computers in the sciences as well as the practical skills they will need if they choose to pursue further graduate study or take technical jobs in the private sector. The role of computers in science is, of course, an enormous topic with many highly specialized niches. This course will focus on what one might call a “base level of knowledge” which most professional scientists will be expected to have. It also will provide a good starting point from which a more advanced course or self-study may follow.The course will be offered at the 500 level (530) through the Department of Physics and Astronomy.

Students will not be required to know computer programming before the course, however they should be comfortable with the use of computers. They also must also have had at least 3 semesters of calculus. This will be a very practical course in the sense that students will learn skills which they can immediately apply in their other courses or professional endeavors. For some topics, designing proper computational tools requires an understanding of the mathematics behind them. So basic theory in optimization (linear regression), numerical derivatives, and numerical integration will be covered.

There are many excellent computer languages on which to base such a course, and the list changes year by year. I believe the Python programming language is an excellent tool for both quickly learning programming concepts and developing truly useful programs. Python is gaining popularity in the physics and biology communities because of it’s rapid development cycle, extensibility, and portability. Some of it’s features are: full object orientation, an intuitive syntax, a large and openly available scientific code base, it is open source and cross platform (even when graphics and windowing are incorporated), and it is a scripting language so “experimentation” is quick and painless. Also, there is a library called “matplolib” which provides a syntax very similar to the popular commercial package Matlab. I generally will use only open source and cross platform tools so students will be able to load these packages on their machines.

Topics to be covered: (time estimates in parentheses - 14 weeks total)

- An introduction to the Python programming language and fundamental programming concepts. (4 weeks)
    1. Data structures: integers, floating point numbers, lists/arrays, functions
    2. Flow control: `while` and `for` loops
    3. Conditional statements: `if-then-else` and `while`
    4. Input/Output: file I/O, user I/O, text formatting
    5. Object oriented programming for scientific applications
- Graphical representation of data. (1 week)
    1. Python graphics and plotting with the *matplotlib* module.
    2. What makes a "publication quality" graph.
    3. More intuitive and informative ways to represent data such as scaling and normalization and multiple data sets.
- Linear and non-linear regression methods. (1 week)
  We'll develop the theory for linear regression and write python programs to perform it. We will also explore packages for fitting non-linear data and develop an understanding of the basic concepts and possible "pitfalls" associated with non-linear fits.
- Numerical differentiation methods and numerical precision and error issues. (2 weeks)
- Numerical integration methods and solutions to systems of ODEs. (2 weeks)
- Roots of polynomials and transcendental functions. (1 week)
- Numerical solutions to PDEs such as the wave and diffusion equations using methods like Lax-Wendroff and Gauss-Seidel Relaxation. (1 week)
- Introduction to parallel computing (1 week).
- Optional Topic Options (1 week)
    - Introduction to graphical user interfaces (GUIs)
    - Time series analysis (Fourier, Wavelet, ...)
    - Cloud Computing for Scientists
    - 3D graphics
    - Integrating Python and compiled code (C, C++, Fortran,...)

Location and Format

This course must be taught in a room which allows students to comfortably use laptops. I suggest a ratio of no more than 1 student per computer so each student is actively engaged in the task at hand. As of Spring 2018, the course will be taught in the Lewis Hall 104 . My experience has been most students like to bring their own laptops (which I encourage). For those that can not bring their own, there will be several Mac desktops available during class time. Students can either access a network storage location for their files or keep all files on a USB key which they would keep with them. Because all the software used in this course is open source, students would be encouraged to install all the packages on their personal computers for work outside the class.

Evaluation

Students will complete weekly homework projects related to the current topics in the course. Most tasks will be done on the computer, however some will require written out answers and some mathematics. There will be two larger scale projects (midterm and final) in which students will choose from a set of topics designed to incorporate a wide range of the concepts covered in this course with the specifics of the problem relevant to their particular field of study. They will have to produce working code and a 4-5 page written description of relevant issues which had to be solved, the capabilities and limitations of their tools, and suggestions for future improvements.

Textbook

Currently there is no textbook which matches the goals of this course very well. I have chosen the book: *A Primer on Scientific Programming with Python, 5th ed.* by Hans Petter Langtangen (ISBN: 978-3-662-49886-6). The book should be a good resource for the students, but we will not be stepping through it chapter by chapter. I will also supply extra resource materials specific to Python, Pylab, etc. to the students.