Physics 91SI: Practical Computing for Scientists

Spring Quarter, 2016–2017

Instructors: Joey Murphy (murphyjm@stanford.edu), Arthur Tsang (atsang2@stanford.edu), Abdallah Abuhashem (aabuhash@stanford.edu)

Course Goals:

This course teaches essential computer skills for researchers in the natural sciences. The goal is to provide students with the essential and most powerful tools used in modern research environments. The course will be taught primarily using the UNIX operating system and the Python programming language, but with an eye toward the different computing environments used in research situations.

By the end of this course, you should be a self-sufficient programmer and software user. This means that you will be able to:

1. Navigate the UNIX operating system and use many of its powerful utilities, including shell scripting, version control, and distributed file systems
2. Be confident using the Python program programming language, including advanced data structures, object-oriented programming, functional programming, and debugging tools.
3. Use scientific analysis and plotting libraries and integrate Python with the operating system and data workflow
4. Plot and present data in an effective and informative manner
5. Autonomously find, incorporate and learn to use the best external libraries for the task at hand

While examples will be drawn primarily from physics, these skills are highly useful in any scientific discipline involving quantitative analysis and are generally considered very desirable qualifications for research positions.

Reading List:

Readings will consist of occasional preparation for lecture topics. This will include

- Online Software Documentation
  - Python Standard Library (http://docs.python.org/library/)
  - Numpy, Scipy, Matplotlib (http://www.scipy.org/)
  - scikit-learn (http://scikit-learn.org/stable/)

- Tutorials and Videos
  - CS1U lectures (http://www.stanford.edu/class/cs1u/)
  - Interactive vim tutorial (http://www.openvim.com/tutorial.html)

Lecture/Lab policy:

- Course meets Tuesday, Thursday 4:30-6:20 in Hewlett 102.

- 2 combination lecture/lab sessions per week
    - Tuesday lectures introduce concepts, lab consists of small conceptual exercises
    - Thursday lectures focus on applications, longer lab with more involved problems

- Office hours by appointment

Assignments, Grading, and Attendance:

- Grades are based on adequate attendance of lectures and lab sessions as well as submissions of assigned lab work. The attendance of all sessions is required, unless there are special circumstances.

- Assignments are completed in the lab sections

- Code submission to be turned in at the end of lab to receive credit

- In order to receive course credit, students must attend all 19 sessions (lecture + lab counts as one session). Under special circumstances, if a student cannot attend class, they should email the instructors and complete and submit the missed lab work before the start of the next lecture.

Syllabus:

(tentative; Tuesday: a, Thursday: b)

Week 1, Introduction to Unix and Python:

a. Unix, File System, Text Editors
b. Language Basics
    i) python interpreter
    ii) python scripts
    iii) control (if, while/for loops, zip, enumerate)
    iv) functions

Week 2, Python Introduction (continued), GIT:

a. Data structures
    i) lists, tuples
    ii) dictionaries
    iii) sets, stacks, queues
    iv) arrays
b. Advanced Unix, mercurial (GIT), shell scripts, piping

Week 3, Numerical and Scientific Python:

a. Numpy, intro to modules
b. Scipy, matplotlib

Week 4, Advanced Python:

a. Functional Python
    i) list comprehensions, mappings
    ii) lambda functionals
    iii) passing functions as arguments
b. More matplotlib, scipy

Week 5, Object-Oriented Programming:

a. Classes, Modules, Exceptions
    i) scope, namespaces, reloading, dir
    ii) operator overloading and special functions
    iii) python data hierarchy: code as data!
b. Exception-catching and debugging
    i) iPython notebook
    ii) unit tests

Week 6, Python Capstone Project:

a. How to find, incorporate and learn new libraries
b. Autonomous project

Week 7, Advanced Topics

a. Intro to machine learning
b. Intro to threading

Week 8. Advanced Plotting/Other languages:

a. Guest Lecture Prof. Hogan - Integrated packages (tentative)
    i) Mathematica
    ii) Matlab
b. Project time

Week 9. Speed:

a. Guest Lecture - Intro to optimization
    i) Runtime analysis
    ii) Loops
    iv) Vectorization
b. Project time

Week 10, Project Presentations/Guest lecture

- Moving on from here
    i) Trends in scientific computing
    ii) Other important topics to learn about
    ii) Courses at Stanford