Practical Computing for Scientists

Spring 2014

Instructors: Sunil Deolalikar, Vincent Su, Marc Williamson, Alex Zannos, Gabriel Ehrlich

Course Goals:

This course teaches essential computer skills for researchers in the natural sciences, with a specific focus on physics. 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 filesystems
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/)
  - *Software Carpentry*: (http://software-carpentry.org/)
  - Cython documentation (http://cython.org/)

- 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 4:15-6:05 in 04-131 (Green Earth Sciences, Rm 131)

- 2 combination lecture/lab sessions per week, with mandatory attendance
    - The first half of each class will consist of lecture, while the second half is devoted to labs that apply the material.

- Office hours TBA

Assignments and Grading:

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

- Completion of the Capstone Project

- In order to receive course credit, students must attend all 20 sessions (lecture + lab counts as one session). The permission of the instructor must be obtained for missing a session and missed lab work must be completed and submitted.

Syllabus:

(tentative; tuesday: a, thursday: b)

Week 1, Unix Introduction:

a. File systems, text editors (focus on vim), basic UNIX

b. Shell scripting, mercurial, advanced UNIX

Week 2, Python Introduction:

a. Language Basics
    i) python interpreter
    ii) python scripts

iii) control (if, while for)
iv) functions

b. ADT's (advanced data structures) and Ipython
i) lists, tuples
ii) dictionaries
iii) sets,stacks queues
iv) arrays

Week 3, More ADT's and debugging:

a. More ADT's
b. Debugging
i) pdb
ii) ipython

Week 4, Scientific Python:

a. numpy, matplotlib, scipy, intro to documentation
b. advanced matplotlib, other packages

Week 5, Advanced Python:

a. Object Oriented programming
i) classes, modules, scope, and namespaces
ii) operator overloading and special functions
iii) python data hierarchy: code as data!
b. Functional Python
i) list comprehensions, mappings
ii) lambda functionals
iii) passing functions as arguments

Brief Introduction to Capstone Project

Week 6, Python Capstone Project:

a. Regular Expressions, Full Introduction to Capstone Project
b. Independent work/brainstorming and in class Office Hours for Capstone Project

Week 7, C and Cython:

a. C programming language, C++

b. Cython, integrating C code into Python for optimization

Week 8, Other Languages and Debugging:

a. Mathematica and Matlab

b. Debugging, Deciphering unfamiliar code, readability, style , documentation

Week 9, Special Topics:

Lectures: Guest lectures by graduate students and postdocs
Labs: Independent work on Capstone Project, in class Office Hours

Week 10, Advanced Topics (tentative):

a. Parallel Programming, Machine Learning

b. Capstone Project student presentations

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