Psychology 611
Introduction to Programming for Behavioral Scientists

Credits: 5
Time Allotment: One session of 2 hours 18 minutes per week.
Audience: Entry-level graduate and advanced undergraduate students.
Prerequisites: None.
Instructor: Simon Dennis or Alexander Petrov

Required Text: How to Think Like a Computer Scientist: Learning with Python.
2nd Edition by Jeffrey Elkner, Allen B. Downey and Chris Meyers, 2008. The text is available free of change online at http://openbookproject.net/thinkCSpy/index.html. We will also be using the pyEPL package and associated manual by Aaron Geller, which is available free of charge at http://pyepl.sourceforge.net.

Acknowledgment: We would like to thank Jeffrey Elkner, Allen B. Downey and Chris Meyers for the work that they have put into the production of the textbook. We would also like to thank Michael Kahana, Aaron Geller and the members, past and present, of the Computational Memory laboratory for their work on the pyEPL system.

Description

Programming is an essential skill for behavioral scientists. Whether you wish to present stimuli to subjects and collect measurements, analyze and plot data or build and test computational models of the underlying processes, being able to program allows you to get the job done just the way you want and on your own time scale. The rigor enforced by programming helps to clarify thinking and often exposes errors in understanding. However, most researchers in the behavioral sciences are interested in how people do things, not how computers do them. Sometimes students don't feel at ease under the tyranny of the command prompt and find the rigid and literal paradigms of programming frustrating and far from intuitive. This course is designed to foster an intimate relationship between student and machine based on understanding and mutual respect.

Objectives

By the end of this course, you should:

1. Understand how programs are used to make computers do useful things
2. Be able to speak the python language
3. Be able to program behavioral experiments
4. Be able to write programs to analyze behavioral data
5. Be able to produce publication-quality graphics
6. Be able to understand and perform signal processing in speech, EEG and image behavioral research

Prerequisites

The course is designed for graduate students with little or no exposure to formal programming languages. Some exposure to statistical techniques is useful but is not necessary. It will be assumed that the student has an active requirement in their home laboratories that will be the subject of the major project (see assessment below). The course is also open to advanced undergraduate students, particularly those who intend to enter graduate school in psychology.

Evaluation

There will be two major components of assessment:

1. In class problem sets (45%)
2. Programming project (55%)
   - Project proposal
   - Code, documentation and unit tests

Problem Sets

Learning programming in lecture format is considered ineffective and inhumane. Consequently, all classes will occur in the laboratory and will primarily involve working through exercises. 5% of the assessment will be assigned to each of these problem sets. These will be marked on a satisfactory, unsatisfactory basis. Attending class and making a genuine effort at completing the exercises will constitute satisfactory performance. If you are unable to attend a given class then you must complete the exercises and submit them electronically through the drop box on Carmen (https://carmen.osu.edu/) by midnight on the Friday of the week of the class in order to receive credit.

Programming Assignment

The remaining 55% of the assessment will involve creating a program to do a task that you need to complete for your own research purposes. This might involve writing experimental code, doing data analysis and/or the creation of graphs of some results, analysis of speech, EEG or image data etc. A one paragraph project proposal will be due mid course and you should consult with the professor to ensure that your proposal is feasible. The final product will include the code, documentation on how to use it and unit tests demonstrating that it works. You will be marked on the clarity, commenting etc of the code, the clarity of the documentation, and the completeness of the unit tests.

Choice of Programming Language

There are a number of languages that could have been chosen on which to base this course. MatLab and R, in particular, are widely used in the behavioral sciences. We have chosen to use the python language (http://www.python.org/). The reasons are as follows:

1) Python has an elegant well designed syntax that incorporates object oriented programming, list and dictionary based data types and functional programming in a simple way.
2) Python is easy to learn. It has an interpreter, so one can start with simple examples and you are freed from the compilation cycle which is confusing for beginners. You can waste a lot of time trying to understand and debug unnecessary bracketing conventions etc.
3) Python promotes and in some cases enforces good programming habits. Indenting is mandatory and it has the literate programming mechanisms that turn comments into HTML documentation. It also has a unit testing framework.
4) Python has all of the components needed by behavioral researchers – an experimental programming laboratory (pyEPL), statistical and signal processing libraries (numpy and scipy) and graphics libraries (matplotlib).
5) It can be used for a wide variety of other tasks as well - it has fully developed web programming modules, regular expression libraries (necessary for doing corpus work) and GUI platforms. These wont be covered in the current course, but it is nice to know they are there should you need to use them in the future. Python also scales well to large applications.
6) There is a large community of users. Based on google document counts the python community is approximately 4 to 5 times bigger than the matlab community, for instance, and includes organizations such as Google and NASA.
7) It is easy to interface to other languages - both high level and low level. So, there are interfaces to

matlab and R (but the reverse is not true). Interfacing to C is done directly - it does not require writing intermediate files (like mex files).
8) There is a free online text (http://openbookproject.net/thinkCSpy/index.xhtml) under the GNU open
document license that has had many contributors and debuggers. The text is also available in hardcopy. There are also many other online resources (e.g. python tutorial http://docs.python.org/tut/tut.html).
9) Python is free and available on all major platforms (linux, OSX and windows). This means that when you leave the university environment either to go home or to start a new job, you will be able to access it wherever you are.

Submitting Assessment

All assessment should be submitted using the Carmen dropbox, https://carmen.osu.edu/ . If you don't know how to login to Carmen or are uncertain how to use the dropbox ask either after class or during my office hours.

Academic Misconduct

Don't be naughty! Sexual, racial, religious or political harassment of any kind will not be tolerated.

All students at the Ohio State University are bound by the Code of Student Conduct (see http://studentaffairs.osu.edu/resource_csc.asp). Suspected violations of the code in this class will be dealt with according to the procedures detailed in that code. Specifically, any alleged cases of misconduct will be referred to the Committee on Academic Misconduct.

Disability Accommodation

If you need an accommodation based on the impact of a disability, you should contact the instructor to arrange an appointment as soon as possible and no later than the end of the second week of classes. At the appointment we can discuss the course format, anticipate your needs, and explore potential accommodations. I rely on the Office for Disability Services for assistance in verifying the need for accommodations and developing accommodations strategies. If you have not previously contacted the Office for Disability Services, I encourage you to do so: http://www.ods.ohio-state.edu , 150 Pomerene Hall, 614-292-3307.

Course Schedule

Week 1: The way of the program, variables, expressions and statements
Reading: Elkner, Downey & Meyers (2008) Chapters 1-2

Week 2: Functions, conditionals and iteration
Reading: Elkner, Downey & Meyers (2008) Chapters 3-6

Week 3: Strings, Lists and Dictionaries
Reading: Elkner, Downey & Meyers (2008) Chapters 7, 9, 12

Week 4: Modules and Files
Reading: Elkner, Downey & Meyers (2008) Chapters 10

Week 5: Classes and Objects
Reading: Elkner, Downey & Meyers (2008) Chapters 13

Week 6: Programming Experiments
Reading: Geller (2006), http://pyepl.sourceforge.net

Week 7: Analyzing Data and Creating Plots

Week 8: Signal Processing (Speech and EEG)
Reading: Vibration Analysis and Theory: Tutorial
(http://www.cage.curtin.edu.au/mechanical/info/vibrations/tutor.htm)
See in particular the section on sampling and aliasing.
Reference: Chatfield, C. (2003). The analysis of time series: An introduction (6^{th} Ed) -- Particularly Chapter 2, Simple Descriptive Techniques
Reference: Fourier Analysis Made Easy
http://www.complextoreal.com/tutorial.htm (Tutorials 4, 5, 6)

Week 9: Signal Processing (Images)
Reference: Graham, N. (1989). Visual Pattern Analyzers. Oxford UP. Chapter 2
Reference: Press, W., Teukolsky, S., Vetterling, W., & Flannery, B. (1992). Numerical Recipes in C: The art of scientific computing (2nd Ed.). Cambridge UP. Chapters 12 and 13: Fast Fourier Transform and Spectral Applications

Week 10: Project Trouble Shooting

Last updated 2008-06-05