Introduction to the Computational Modules

Overview

These modules, developed under a grant from the TIDES (Teaching to Increase Diversity and Equity in STEM) project of the Association of American Colleges & Universities, are designed to teach the basics of algorithmic thinking and computer programming to students in the physical sciences using the Python programming language. It is hoped that the set of modules will work as a largely independent learning experience in basic programming skills for science students; of course, the set also could serve as the core content or a supplement to the main content of a formal course. In fact, they were tested out on and refined with feedback from students who used them in a formal computational methods course (PHYS 350) offered at Bryn Mawr in the spring of 2015. (That course was run essentially as a computational lab: students were asked to do readings in the modules ahead of class time, and they worked on the embedded exercises during class. Lecturing was kept to a very bare minimum.)

An additional goal of the set of modules is to attract to computing and computational science students from groups typically underrepresented in those fields. One approach to that goal is to give students an appreciation of the wide range of individuals who have contributed to the development of computational science, or who have used it in interesting or important ways. This approach is implemented in each module by the inclusion of a scientist profile, most of which were developed by the students in PHYS 350. A second approach, which was employed at regular intervals in that course, was to ask students a "reflection question" meant to encourage them to think about how science and computation was relevant to them and could serve their personal goals. They kept an electronic journal of those repsoonse and some cases shared their responses with another student of the full class. We asked the students in the course at the end of the semester to submit an "e-portfolio" including all of their work in the course: module exercise solutions, their scientist profile, responses to reflection questions, and term project write-ups. The guidelines we provided the students regarding the profiles, and the reflections questions/prompts we asked them to respond to, are provided at the end of this document. These elements of the course were informed by the body of literature describing what we know about how to create culturally responsive and inclusive learning environments. We developed a faculty development workshop based on these resources and it too is included in this repository.

The Modules

Eventually, we hope to have all of the modules ported to IPython "notebook" format. IPython notebooks can incorporate not only executable Python code, but also LaTeX, HTML content, images, videos, and sound files. As of now, all but Module 1 (and a "mini-module" on object-oriented programming) are in the form of PDF documents; Module 1, on the basics of Python, is divided into three interlinked IPython notebooks.

To try out a notebook without having to install any software locally, go to https://try.jupyter.org : there, you can upload a notebook and run it on the remote server. Notebooks can be run on a local server by installing the Jupyterhub system. See https://github.com/jupyter . (Doug Blank, of the Bryn Mawr Computer Science department, has extensive experience with Jupyterhub and could provide some guidance on installing it.) They also can be run on a desktop or laptop computer by installing a commercially-produced (but free) Python software environment, as discussed in Module 0.

Below is a brief listing of the topics covered in the modules

- Module 0 provides an overview of computing and lists many resources for learning Python and running it and IPython notebooks on a local machine.
- Module 1, in the form of three IPython notebooks, presents the basics of Python (version 3), with exercises that students should work through in sequence. (If the three notebooks reside in the same directory, the links in the notebooks should correctly interconnect them.) Students who have prior familiarity with Python may know of simpler or more elegant ways to do some of the exercises, but they should be encouraged to use only the information provided in the module prior to each exercise, to ensure that they understand the associated programming concepts and Python content.
- Module 2 is a brief discussion of numerical errors and computational speed.
- Module 3 introduces some basic programming constructs in the context of the time-evolution of the kinematic equations. The Euler and Euler-Cromer methods of numerical integration are presented.
- Module 4 discusses numerical differentiation and interpolation.
- Module 5 covers numerical integration.
- Module 6 presents techniques for numerically solving linear equations.
- Module 7 covers the solution of eigenvalue-eigenvector equations.
- Module 8 discusses some approaches to modeling data: least-squares fitting, singular-value decomposition, and principal components analysis.
- Module 9 introduces Fourier analysis and its numerical implementation.
- Module 10 deals with the solution of ordinary differential equations.
- Module 11 covers the solution of partial differential equations.
- Module 12 introduces some basic Monte Carlo techniques.
- Module 13 presents basic topics in object-oriented programming.

Nearly all of the topics in these modules can be explored in greater depth in the excellent text *Computational Physics* by Mark Newman, which provided the foundation for these modules. A useful supplement is *A Student's Guide to Python for Physical Modeling*, by Jesse Kinder & Philip Nelson.

Licensing

The modules and associated materials are licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license.

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

These uses are granted under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for commercial purposes.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Scientist Profile Instructions

Our objective for the Profile Collection is to gather together contemporary (last ~50 years) stories of men and women who have contributed to the development and use of computing techniques in the sciences. We would like the collection to showcase the achievements of people from a diversity of backgrounds and the ways in which their stories inspire us. Particularly, those who may have overcome challenges or whose accomplishments may have been overlooked.

Please address these questions in your profiles:

- Who is the person and what is their context? Tell us about their education background, i.e., where and what did they studied, and something about their career path.

- How does their work connect to computational techniques and/or scientific computing?

- What is inspirational about their story?

Please prepare two depictions of your profile:

- A written text document with a visual element and a description that includes, but is not limited to the answers to the three questions above. Please include any references that you found on the profiled individual. Length ~250 words.

- A single slide to support a 5-minute presentation of your profile. Be creative! And feel free to include material beyond the three questions if you like.

Reflection Prompts

Values affirmation: Think about 2-3 things that you value and care about and why.

Integrating Profiles: What ideas do you have about incorporating the profiles into the Modules themselves?

Programming help: What do you do when you get stuck? What resources do you turn to? Which have been useful?

Profiles succeeding?: Do you think the profiles we've seen so far represent a set of scientists from groups underrepresented in their fields?

Relevance: What connections do you see between this class and other classes/ research/ academic pursuits this semester?