

Python as a Tool for Squeezing More Learning into Mathematical Physics:  
Powerful, Versatile, and Free

Geoffrey Poore  
Assistant Professor of Physics

# Python as a Tool for Squeezing More Learning into Mathematical Physics: Powerful, Versatile, and Free

During the 2011 spring semester, I have thoroughly integrated the Python programming language into my Mathematical Methods in Physics course. So far as I am aware, this is the first time Python programming has been taught in any class at Union. It is certainly the first time within Physics or Computer Science.

Python was first released in 1991 and has been used for general-purpose programming for years by companies such as Google and organizations like NASA.<sup>1</sup> Python is simpler and faster to program than older scientific programming languages (for example, FORTRAN and C/C++), but runs significantly slower. Recently, numerical libraries have been developed that eliminate much of Python's speed disadvantage, while new scientific and plotting libraries have transformed Python into an excellent tool for data analysis. Python is now competitive with more traditional data-analysis programs such as Mathematica and Matlab. (Both of these are used at Union, in Mathematics and Engineering, respectively.) Unlike Mathematica and Matlab, Python is a complete, general-purpose programming language. This means that in Python, you are free to do almost anything you can think of, in the most efficient way you can conceive, without being limited by a special-purpose language.

By introducing my students to Python scientific programming, I am giving them experience with a powerful yet simple language that is increasingly useful in scientific computing. Prior to 2006, the use of Python in mathematical and computational physics classes was limited because many of today's tools did not yet exist. MIT experimented with Python in its introductory computer science and electrical engineering courses in 2006 and began official curriculum adoption in 2007.<sup>2</sup> A search on Amazon for books on Python and physics returns only four general texts, all but one published since 2009. Because Python is free and open source, I am also providing my students with a tool they can continue to use in the future without paying hundreds or thousands of dollars for similar commercial software.

## Python in Mathematical Methods

I have structured Mathematical Methods so that students gradually gain experience with Python. This allows them to become comfortable with the language before they receive more complex assignments. The goal has been to give them experience applying Python to typical problems encountered in mathematical physics and more broadly in physics research.

- At the beginning of the course, I introduced students to arithmetic and plotting in Python. This fit well with our brief review of complex numbers.
- The first major topic of the course was Fourier analysis, which looks at the frequencies present in sound and other waves. We used Python to plot the frequencies present in a clip of Handel's *Messiah* to which a 400-Hz humming noise had been added. Then we considered various ways to remove the unwanted hum. Python also allowed us to create sounds from scratch. We chose the frequencies we desired, and the program created the corresponding sound (for example, 262 Hz for Middle C). This allowed experimentation with beat frequencies and random sounds.
- Our next topic was special functions. In class the students plotted many of the functions to gain an intuitive feel for their behavior. We also used Python to evaluate the functions. In homework, the students used Python to plot functions and to check some of the results they calculated by hand. This gave them experience using Python to solve calculus problems symbolically as well as numerically.
- We are continuing to use Python for plotting and solving equations in our current study of complex functions. This requires building on the plotting and solving skills the students have already learned, since these functions require somewhat different methods.
- We are about to start our final topic of the semester: numerical methods for solving equations. This deals with how computers actually solve various physics equations. We will not simply learn the math and theory behind these computational methods. The students will actually be programming the methods using Python, and then using their own programs to solve problems. This is where the simplicity of Python will be particularly valuable.

## Current practices

Some mathematical methods courses involve no computer work at all. For example, I took a graduate-level mathematical physics course about six years ago at a major research university, and it involved no computer work whatsoever. The same professor continues to teach that course, using the same approach. While using a computer to solve equations is inappropriate for some topics (there are some things you must know how to solve by hand to be good physicist!), I find that plotting exercises are useful even in those cases since they give students an intuitive, visual feel for what they are working with. Thus, I have had my students use computer plots for all topics we have covered, even when I have forbidden using the computer to actually solve the math. This provides a more interactive, exploratory learning experience.

When mathematical physics courses do involve computer work, they typically use the commercial programs Matlab or Mathematica. If I had to choose one of these programs for my class, I would choose Matlab, based on my previous experience with it and on its number-crunching abilities. Python is capable of doing everything I would do with Matlab and has several advantages.

- Some types of plots are much easier to create using Python.
- One of the most common data-analysis tasks is creating figures for a publication, presentation, or dissertation. In several of the sciences, figures may be included in documents typeset with the  $\text{\LaTeX}$  language. Python automatically has full  $\text{\LaTeX}$  support for figures, but Matlab has limited  $\text{\LaTeX}$  support.
- The Python language is a complete general-purpose language. Python is easy to read and write. Programs are simple to organize. The Matlab language was created just for Matlab. It is missing some features of a regular language and is awkward at times.
- Python is usually at least as fast as Matlab. When more speed is needed, Python offers many options, some of which take only moments to program. Matlab offers fewer options to increase speed and these are generally more complicated.

Mathematica does not have the plotting and  $\text{\LaTeX}$  issues of Matlab, but it is still a special-

purpose language and has its own quirks. Compared with both of these, Python has the additional advantage that it is free and open source, and thus always available to students.

## Evaluation

I have been very pleased with the role Python has played in Mathematical Methods. My students have adapted to using Python without difficulty. In part this is a tribute to the Mathematica use in Union's calculus sequence, since many programming concepts carry over. It is also due to the simplicity and clarity of the Python language. My students are effectively learning the computer-related course material while gaining experience with a new programming language. This is particularly valuable since physics majors are not required to take a programming course.

The next time I teach Mathematical Methods I plan to use Python again. Now that I have more experience with Python, I believe I can teach more efficiently and provide improved exercises for students. I have already used Python in every part of the course, but this can be refined. Since students seem to do well with the language, in the future I will introduce some key programming concepts earlier in the semester. I will also customize some of my teaching to account for the students' background. For example, there is a simpler way to introduce Python plotting to students who have used Mathematica.

In some parts of the course, especially our investigation of sound frequencies, Python made it possible to do more than I had hoped for. When I teach Mathematical Methods again, I would like to further develop the work with sound. With some additional programming, it should be possible to record sounds, look at the frequencies they contain, edit the frequencies (for example, remove background noise), and then listen to the results, all within Python. The Department of Physics does not currently have the lab apparatus to do this type of experiment—our lab software only allows recording sounds and plotting the frequencies they contain.

Based on my experience this semester, I anticipate using Python in some other, upper-level physics classes. Python has a powerful yet simple 3D animation package that can be used to animate physics processes, and I have barely experimented with its capabilities.

I have also been pleased with Python personally. At the beginning of last fall, I had never used

Python. I played with it a little during the fall semester as a possible Matlab replacement, and based on that experience decided to use it this semester as part of Mathematical Methods. Continuing to learn a new programming language while teaching a class that uses it has been challenging at times, but for the most part it has been a good experience. Python has many features I found lacking in Matlab (especially a complete programming language), with few if any disadvantages for teaching or for the types of research and data analysis I have done in the past. The main drawback I have encountered is that Python's symbolic mathematics capabilities are less powerful and complete than Mathematica's, and thus it cannot serve as a complete Mathematica substitute. I expect Python to become my primary computational tool in and out of the classroom. My experience this semester has already led to a software development project involving Python and the L<sup>A</sup>T<sub>E</sub>X typesetting language (commonly used for writing and submitting papers in physics, mathematics, and computer science). I hope to present a talk based on that work at the annual Python in Science conference in 2012, and pursue publication in a computational science journal thereafter.

## References

<sup>1</sup>Python Software Foundation (2011). General Python FAQ. Retrieved from <http://docs.python.org/faq/general.html>.

<sup>2</sup>Daher, W. (2006, February 1). EECS Revamps Course Structure. *The Tech*, 125 (65), 1 & 17.