Editors: Larry Engelhardt, lengelhardt@fmarion.edu | Stephen P. Weppner, weppnesp@eckerd.edu

Finally, a Python-Based Computational Physics Text

Eric Ayars | California State University, Chico

This is the book I've been waiting for. There are many good books on computational physics, and many more on programming in Python; but so far there has been a distinct lack of books on computational physics using Python. *Computational Physics*, by Mark Newman, is to my knowledge the first published attempt at filling that gap, and it's pretty good.

For readers who have just awoken from a 15-year nap and are using this journal to find out what happened while they were dozing, Python is a programming language that almost perfectly hits the sweet spot between power and ease of use. Its simple syntax and enforced code structure make it ideal as a first programming language, but it's still powerful enough to be useful for all but the most arduous calculations. The lack of a textbook that uses Python for code examples and introduces the tools that Python offers has so far been a serious impediment to adoption of this otherwise ideal choice of language for the introductory programming course for science majors; now that impediment is gone.

Strengths

Newman begins with a chapter on the basics of Python syntax and structure, which is sufficient to get most students started on programming without prior experience. The chapter immediately following introduces the graphing capabilities of Python. This one-two combination allows students to quickly start using what they're learning in *other* classes, which in turn increases their appreciation of the rest of the material.

Before getting into the rest of that material, though, the author takes students through a short chapter titled "Accuracy and Speed." This choice initially struck me as odd; but interspersed throughout the rest of the book are discussions of precision, accuracy, and comparisons of the relative errors of the different methods presented. The value of these discussions to the student is greatly enhanced by this short chapter, which preemptively ties all that material together.

The rest of the book covers the usual spread of introductory computational topics: integration and differentiation, root-finding, Fourier topics, ODEs, PDEs, and Monte Carlo techniques. Explanations are clear and readable. The author almost invariably begins with a thorough explanation of the simplest approach, and then builds on those simple approaches to more powerful techniques. Just about every section has something that ties back to that "Accuracy and Speed" chapter; it's not mind-numbingly rigorous, but it doesn't skimp on ensuring that students understand the steps necessary to obtain trustworthy results. (Refreshingly, the students also have a chance of avoiding useless over-calculation!) In most cases, the author finishes the section with a mention of the Python functions available through numpy or scipy that efficiently tackle that particular problem.

The book is surprisingly readable. It's never going to make it onto the reading list for Oprah's book club, but it's a vast improvement over the typical "here's how to do such-and-such" book of numerical techniques. Topics I understood completely beforehand were still interesting to read about in the book. Topics I didn't understand as well from my own classroom exposure to "Numerical Recipes in C"—Romberg integration, for example—now make more sense to me than ever before.

Weaknesses

Everyone who teaches a computational physics course will find something missing from whatever textbook they pick up. For me, with this text, those missing things are files and curve fitting.

The entire coverage of curve fitting in this text fits within one end-of-chapter problem. It's a good problem, and the multistep approach the author takes is a nice introduction to least squares, but that's it! There's no mention at all of nonlinear curve fitting, or of the excellent curve-fitting capabilities available in Python via the scipy module.

The lack of file-manipulation coverage is also disappointing. For many students in this sort of course, this one semester may be the only formal computer programming training they receive. Vanishingly few of those students will go on to careers in computational physics, but most will end up needing to do something with computers on a regular basis. The ability to read and write data files is invaluable to students in the lab, and I've heard enough positive feedback about this from former students that I would hate to leave out this topic.

These omissions can be easily supplemented in class, though; and a textbook that included everything every professor ever wanted would be frightening.

Mark Newman, *Computational Physics*, CreateSpace Independent Publishing Platform, 2012, ISBN: 978-1480145511, 562 pp.

 1521-9615/14/$31.00 © 2014 IEEE Copublished by the IEEE CS and the AIP

I teach a one-semester undergraduate course in computational physics; the students are mostly physics majors, with an occasional chemist sprinkled in for variety. As is the case in many other universities, there is no computer-programming prerequisite for this course, and it's the only programming course these science majors receive. The lack of a Python-based introductory textbook for this class has been a problem to me.

There are advanced texts[1,2] and there are specialized texts,[3] but none of these fit my needs for this course. I've taught in the past with a Fortran-based book[4] (rewriting all the example code in Python) and I've taught with the "language-agnostic" version of Nicholas Giordano's book,[5] but neither approach was really satisfactory. For the last couple years, I've been writing my own book for use in this class. I can stop now. I'll keep my project around to supplement his omissions, but realistically, Newman's book is far superior to my efforts and I'll quite happily be adopting this book the next time I teach the course. ■

References

1. R. Landau, M.J. Paez, and C. Bordeianu, A Survey of Computational Physics, Python Multimodal eBook, 2011.
2. H.P. Langtangen, *A Primer on Scientific Programming with Python (Texts in Computational Science and Engineering)*, vol. 6, Springer, 2009.
3. Shai Vaingast, *Beginning Python Visualization: Crafting Visual Transformation Scripts*, Apress, 2009.
4. Tao Pang, *Computational Physics*, Cambridge Univ. Press, 1997.
5. N.J. Giordano, *Computational Physics*, Pearson Prentice Hall, 2nd ed., 2006.

Eric Ayars is a professor of physics at California State University, Chico. In addition to computation, his research interests include development and applications of microcontroller systems for physics and cross-disciplinary undergraduate research. Ayars has a PhD in physics from North Carolina State University, and can be reached at ayars@mailaps.org.