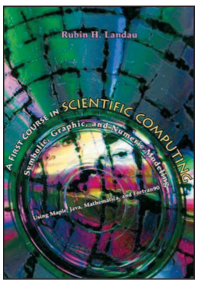
Editor: Mario Belloni, mabelloni@davidson.edu
Send announcements to bortega@computer.org

COMPUTATIONALLY COMPLETE

by Michael Jay Schillaci

R. Landau, *A First Course in Scientific Computing: Symbolic, Graphic, and Numeric Modeling Using Maple, Java, Mathematica, and Fortran90*, Princeton Univ. Press, 2005, ISBN: 0691121834, 472 pages.



Aimed directly at the undergraduate student, the stated goal of Rubin Landau's *A First Course in Scientific Computing: Symbolic, Graphic, and Numeric Modeling Using Maple, Java, Mathematica, and Fortran90*, is "[t]o provide them with tools and knowledge they can utilize throughout their college careers." Adopting a tutorial approach and an "over the shoulder" style that lets students work independently and confidently, Landau addresses almost all the shortcomings of earlier computational physics books and produces a text that will certainly stand the test of time.

Drawing on both a long career of teaching excellence and from his prolific body of research, Landau wisely guides his readers through the scientific computing world with what at times feels like kind, grandfatherly advice. Indeed, in the preface—a part of the book that many students might unfortunately skip—he deftly focuses the reader's attention on the simple observation that, "[t]he basic ideas behind scientific computing are language independent, yet the details are not." With this guiding principle, he sensibly chooses Maple and Java as exemplars of the vast array of tools available to students. Together with the accompanying CD, which includes alternate code for Mathematica and Fortran90, the text's intended role is to be "closer to a workbook than a reference book." In this regard, Landau does very well. With a short chapter on LaTeX, he also addresses the ever-increasing role of electronic document production in the Web-based communication of scientific works and so provides the student with a complete set of tools for the task at hand.

Maple

To begin, although Landau's use of the command-line mode is appropriate because of platform variability concerns, the addition of a "walk through" of Maple's features would serve to give students a literal view of what's to come and could provide witness to the way in which students could integrate different programming environments. By demonstrating Maple's ability to typeset mathematical equations, plot functions, and export graphics and pages as HTML or LaTeX, for example, students could quickly appreciate its usefulness. Instructors could then point to the examples as greater explanation and motivation for the course's scope and content.

Partly due to the workbook approach, the Maple discussion's overall organization is somewhat terse with many sections, subsections, and sub-subsections that don't always seem particularly pertinent to the task of "introducing the relevant mathematics in the course of solving realistic problems." This is especially true when using the book in courses designed to include students from curricula other than physics. For example, the first problem in the text draws on the results of special relativity, and Landau takes great care to entice the student saying, "[a]lthough the theory of special relativity does have its subtleties […] [t]his should give you a good working knowledge of some tools." Given the text's audience, the first example should allow all students, not just physics majors, to fully grasp the phenomena's physical significance. Moreover, nearly 20 pages separate the introduction of relativistic equations with the plots of their basic behavior, with much of the intervening material coming from a computer science perspective. This is avoidable in an electronic format, but in written form, an organizational scheme that sets the tutorial information and exercises in sidebars or boxes would be preferable.

Although the mathematical and computational techniques Landau covers include the detailed use of integration and differentiation as well as matrix algebra and advanced plotting commands, too much is made of Maple's limitations. A measure of this is necessary, but in some cases this approach might challenge students' confidence in Maple—for example, Maple sometimes allows operators to be omitted (as when employing its version of scientific notation); a statement that using operators will decrease your debugging time would be useful in this circumstance. Moreover, when illustrating Maple's computer algebra system, the text focuses on the nature of commands such as `simplify` and

Not So New but Improved

In early 2006, the *CiSE* editorial board evaluated the Books Department. The outcome of this discussion was that the department emerged as an important section of the magazine that we needed to rework and update. With *CiSE's* larger constituency, the Books Department seeks to have "something for everyone," so our goal is to feature at least one book review per issue, which will come from these general categories:

- programming (design, tools, techniques, languages);
- computational sciences (engineering, biology, chemistry, physics, and so on);
- engineering and science applications (CAD/CAE, productivity, modeling, simulation);
- algorithms (high-performance and parallel computing, numerical, symbolic);
- data (mining, databases, analyses);
- textbooks (computer science or science with computation);
- laboratory/experimental (acquisition, control systems, imaging, sensors); and
- hardware/networks (special purpose, distributed systems, communications).

The Books Department is only as good as the books we receive, though, and the reviewers who review them. If you know of a good book worthy of review, or want to review a book in one of these categories, please contact me at mabelloni@davidson.edu.

We kick off this revamped department with Michael Jay Schillaci's review of Rubin Landau's book, *A First Course in Scientific Computing*. Much debate exists as to what a first course in scientific computing should be (see for example, the September/October 2006 issue of *CiSE*), and Landau's approach has been at the forefront of much of this discussion. In addition to our review, the *American Journal of Physics* recently published a comparison review of three first-course books (vol. 74, no. 7, 2006). Our reviewer also has his own first-course approach, which you can download as a PDF from his Web site (www.evsis.org/download.html).

`collect`, whereas it fails to give a detailed treatment of extremely important commands like `unapply`. Consequently, the opportunity to impart high-order heuristics to the student is lost. By including short tables or intermittent summaries of important built-in functions and commands in the text (in addition to those already in the appendix) and with a more judicious use of the command-line help system, the delivery and impact of the examples could be improved.

When tackling more complex ideas, the text's colloquial approach often comes across as abrupt. For instance, in discussing how a 3D plot structure is rendered on a 2D surface, Landau says, "we do that by rotating the object, shading it, employing parallax, and so forth." Arguably, one of the most appealing aspects of software packages such as Maple or Mathematica is that the nontechnical masses can use them to blindly produce complex and beautiful graphics, but one of a scientific computing course's many goals is to give students a deeper understanding of the principles involved. For this reason, it might have been more beneficial in this context if the text demonstrated how the student could add plot options such as viewpoint and shading to the basic command structure to alter the object's appearance. Then, the student could use the matrix rotation techniques to construct a simple animation sequence, directly illustrating the effects and providing a concrete and in-depth example. Indeed, with a similar approach to each of the chapter's problems, Landau could enhance the coverage of Maple methods and programming.

Java

The thorough treatment of Java as a paradigm example of a modern object-oriented programming (OOP) language, complete with plotting and Web-based applications, is where Landau's text makes its real contribution. Although his first example might seem a bit mundane—calculating a circle's area—it goes beyond the traditional "hello world" program and introduces the language's method-based structure. In particular, Landau's discussion of classes and methods and when to avoid thinking too deeply about the required syntax of declarations is very refreshing. To quickly demonstrate Java's power and appeal, he then shows how easy it can be to produce graphical output.

Despite the fact that some of the physical examples covered in the Maple section weren't fully developed, Landau's use and expansion of these same examples later in the text (most notably, the detailed simulation of a large city's electricity usage) should let students move confidently into modern scientific computing's more technical aspects. Moreover, his addition of "new" problems such as frictionless projectile motion provide students with a demonstration of how theory, algorithm development, and logic must all come together to produce working simulations. The comparison of Java and Maple solutions to differential equations helps drive home this point further and also provides the unified approach that's lacking in some of the text's earlier portions.

Landau employs a "just enough" approach when it comes to his discussions of OOP concepts, but the ideas of encapsulation and inheritance are dealt with sufficiently and make the move to Web computing easier to appreciate. Because of a small typographical error (probably placed there by Landau to encourage students and instructors to type in their own code!), the first Web application didn't run "out of the box." By extending this basic applet project with the Swing classes to include user control, he implicitly reminds us that much of scientific computing's lineage was based on copious use of legacy code and libraries. However, the follow-through absent in the Maple section is also evident here as Landau stops just short of a full-blown Web application using JavaScript to allow for direct user interaction via form input.

LaTeX

Despite the fact that the text is formally broken into three sections, the third section—entitled, "The LaTeX Survival Guide"—is very short. Nevertheless, it does an adequate job of introducing the essential LaTeX commands and environments that a student would need to produce high-quality and content-rich documents suitable for laboratory reports, refereed journal submissions, or Web postings. The reasons for including LaTeX in a first course in scientific computing are clear in that the superior mathematical typesetting ability and the resulting electronic documents (that is, EPS or PDF) are ubiquitous. However, LaTeX's integration and connection to the extent material in the text and course isn't evident in Landau's treatment. Specifically, no discussion of styles and packages is included—which is essential if you want to compile the LaTeX source from a Maple worksheet. This level of detail isn't often found in the many TeX primers available online but adds a decidedly more complete view of the subtleties of electronic document production. Instead of viewing this as a liability, you could rearrange the text material to cover LaTeX elements first and then move to Maple (or Mathematica). This would have the added benefit of allowing nontraditional students to learn and refresh their programming and debugging skills while working with less cognitively demanding material.

It's arguable that an introductory scientific computing course ought to be accessible to all students of the broader sciences. A conscientious instructor willing to reorganize and extend some material to make it more suitable and appealing to a multidisciplinary student body could use *A First Course in Scientific Computing* to cast this wider net. Indeed, the colloquial and tutorial approach might help alleviate the many practical problems associated with incorporating computational applications into a more traditional lecture environment. The text provides many concrete and programming examples in action and illustrates how much you can accomplish with a few well-chosen tools. All in all, students impressed with the text's workbook style and reference book quality will add it to their bookshelves and return to it often.

Michael Jay Schillaci is managing director of the McCausland Center for Brain Imaging at the University of South Carolina. His research interests include computational physics and curriculum development, and models of human cognition using magnetic resonance imaging and electroencephalographic data. Schillaci has a PhD in physics from the University of Arkansas at Fayetteville. He is a member of the American Physical Society and the Cognitive Neurosciences Society. Contact him at mjs@sc.edu.