Editor: Denis Donnelly, donnelly@siena.edu

COMPUTATIONAL PHYSICS FOR UNDERGRADUATES

THE CPUG DEGREE PROGRAM AT OREGON STATE UNIVERSITY

By Rubin H. Landau

WE PRESENTLY ARE EXPERIENCING HISTORICALLY RAPID ADVANCES IN SCIENCE, TECHNOLOGY, AND EDUCATION DRIVEN BY A DRAMATIC INCREASE IN COMPUTER USE AND POWER. IN THE PAST, EDUCATORS WERE CONTENT TO HAVE undergraduates view scientific computation as black boxes (an abstraction of a device in which only its externally visible behavior is considered, not its implementation) and have them wait for graduate school to learn what's inside.[1] Our increasing reliance on computers makes this less true today, and much less likely to be true in the future. To adjust to the growing importance of computing in all of science, Oregon State University's Physics Department now offers a four-year, research-rich curriculum leading to a bachelor's degree in computational physics (CP; www.physics.orst.edu/CPUG/). The five computational courses developed for this program act as a bridge connecting physics with the computation, mathematics, and computational science communities.

The Oregon State Program

The Oregon State Board of Higher Education approved the CP degree as separate from its traditional physics degree in October 2001, after two years in administrative processing. Even though the first class did not matriculate until fall 2002, we had our first graduate, a transfer student, in June 2003.

Presently, eight students are enrolled in the program, although classes are well attended because Oregon State requires all physics majors to take the introductory classes, and others students (including graduate students) have the option to take the upper-level classes.

Oregon State University's Computational Physics for Undergraduates (CPUG) program has been building over time from the bottom up. CPUG began in 1989 with a senior/graduate-level, two-term course in CP. IBM supported the course from the beginning with the donation of an RT workstation; the US National Science Foundation followed suit with two grants. The Undergraduate Computational Engineering and Science group (www.krellinst.org/UCES/index.html) recognized my contribution to computational science with an award in 1995. John Wiley & Sons published the course materials in 1996 as the text *Computational Physics* which I wrote with Manuel Páez.[2] It joined the works of Harvey Gould and Jan Tobochnik,[3] Paul DeVries,[4] and Marvin De Jong[5] as models for undergraduate CP courses.[6] Simultaneous with the text's completion were early explorations into the use of the developing World Wide Web to provide multisensory text enhancements (an example of this is Oregon State University's NACSE Research Group in Physics at www.physics.orst.edu/~rubin/nacphy/).[7]

In 1997, the physics department introduced a one-quarter Introductory Scientific Computing course designed to provide first- and second-year students with the computational tools needed throughout their undergraduate careers.

My colleagues, students, and I have now prepared extensive introductory materials[8] that pave a continuous path to upper-level CP courses. We hope to have the introductory materials published in 2004 by Princeton University Press, and have them fit in well with the new edition of *Computational Physics*,[9] planned for 2005.

With the department's addition of an advanced CP laboratory, the extension of our CP course to the sophomore and junior level, and the use of courses offered in the computer science and mathematics departments, we believe we have assembled a coherent and strong undergraduate degree program in CP. (I will discuss a sample curriculum in more detail later.) By teaching five computing classes in the physics department, we can adjust their content and depth to provide balance within the allowed university credit limit. This also avoids the difficulties associated with trying to get other departments to teach shortened versions of courses designed for their majors. As an added benefit, our program meets a need to provide undergraduates with research experience, a hallmark of highly ranked universities.[10]

Need for Computational Science Degree Programs

One of our motivations for starting a CP degree program was the faculty's observation (especially Henri Jansen and I) that the average computer science graduate does not have the mathematics and science background needed for technical employment, and that the average physics graduate does not possess the requisite background in computation. Another motivation was that the President's Information Technology Advisory Committee (www.ccic.gov/pitac), the US Department of Commerce, and *InformationWeek* all observed that computer science departments alone cannot meet the country's needs for computer professionals.

Evidence indicating the general types of skills needed in the technical workplace is provided by an American Institute of Physics survey.[11] The results (see Figure 1) indicate which aspects of their education physics majors found most valuable in their current employment, when polled five to seven years after graduation. For graduates whose primary field of employment is engineering, mathematics, and science, the three most important skills are scientific problem solving, synthesizing information, and mathematical skills.

These skills remain highly important for graduates who find employment related to software, with this group also having a high need for computer programming and software development. A National Science Board report also examines the importance of mathematics and computer skills,[12] indicating that 74 percent of mathematics and computer science doctorates work in the same field as their degree, in contrast to only 52 percent of degree holders in life and physical sciences. We note a similar trend at the bachelor's degree level (35 percent versus 22).

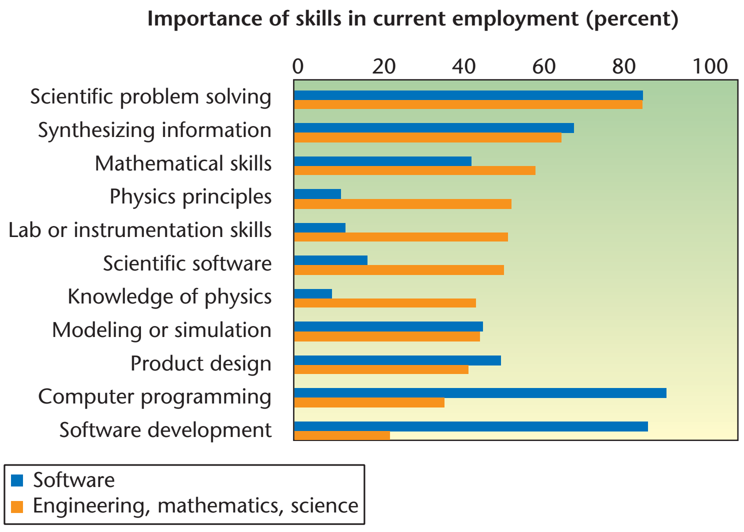


Figure 1. The importance of knowledge and skills for graduates with bachelor's degrees, five to seven years after graduating. The orange bars describe the importance for physics majors whose primary field of employment is engineering, mathematics, and science, while the blue bars are for graduates employed in software-heavy professions. (Data courtesy of the American Institute of Physics.)

A bachelor's degree in any computational science is rare, as Charles Swanson,[13] Osman Yaşar, and my[14] program surveys have shown. Most programs are options or minors to standard degrees, or a course or two, with only seven schools offering actual bachelor degrees. Our CP degree program is one of only three in the US. There are other CP bachelor's degree programs at Illinois State University (www.phy.ilstu.edu/CompPhys/CompPhys.html), State University of New York at Buffalo (www.physics.buffalo.edu/undergrad/cp.html), and Trinity College, Dublin (www.tcd.ie/Physics/Courses/CCCP/CCCPflyer.html), as well as CP minors or specialties at Syracuse University (http://suhep.syr.edu/undergraduate), Clark University (http://science.clarku.edu/compusci.html), and Rensselaer Polytechnic Institute (www.rpi.edu/dept/phys/Curricula/currAppPhysComp.html). In addition, State University of New York at Brockport (www.cps.brockport.edu), the University of California at Berkeley (www.coe.berkeley.edu/engsci/), the Australian National University (http://room.anu.edu.au/bcomptlsci/), Kanazawawa University, Japan (http://cmpsci.s.kanazawa-u.ac.jp/English), and the National University of Singapore (www.cz3.nus.edu.sg/AY2001-02handbook.html) all have bachelor's degree programs in computational science and engineering.

Although its numbers are small, Oregon State University's program is beginning to draw some students who would not otherwise be in physics. Some of this is due to the rarity of degree programs in computational science, some to the existence of students who are interested in both computers and science, and some to our promotion at conferences, orientations, and school visits. For example, prospective students are now presented with a description of a number of programs on campus that contain computing—in addition to the ones in computer science. This benefits the students as well as the computer science department, whose courses are overenrolled.

In addition to new students, the students and I have been pleased to discover that it is relatively easy for physics or engineering-physics majors to be enrolled as dual majors or degree candidates with CP. This is a consequence of the similarity of requirements and the large percentage of students taking more than four years to graduate. Not only does this help the university get a new program going, but it also helps students obtain credentials that will be useful in their careers, be it the job market or graduate school.

Student Learning Outcomes

We want our graduates to possess realistic problem-solving skills and to carry off a competent understanding of physics, applied mathematics, and computing. The students should understand how to perform scientific computations, as well as experience the interweaving of high-performance computing and communications into modern science. As it does for us, we want the students to see physics come alive before their eyes and reveal itself at a level usually attained only in a research environment.

As is true for other computational science programs,[15] our specific learning outcomes include

- learning high-level computer languages and high-performance computing;
- obtaining knowledge of applied mathematics and computational methods;
- learning simulation and modeling basics;
- interpreting and analyzing data visually during and after computation;
- applying acquired computing skills to at least one application area; and
- learning to effectively communicate solution methods and results.

In our approach, we do not try to have an individual course on each of these topics, or even to spend specific time on each learning outcome. Rather, we incorporate them into the projects we develop and the assignments and exams on which the students work.

Our Course of Study

We generally develop our computational materials in the scientific problem-solving paradigm:[13]

problem→theory→model→implementation→assessment,

where the assessment links back to all steps. This paradigm distinguishes the different steps in scientific problem solving, encourages using a variety of tools, and emphasizes the value of continual assessment. It has also been shown that the use of the problem-solving paradigm deepens scientific process skills,[16] and often leads to a research-level insight into physics.

Engaging Students

A key component of our program is having students get actively engaged with projects, as if each were an original scientific investigation, in a large number of areas. Students can then experience the excitement of individual research, become familiar with several approaches, acquire confidence in making complex systems work for them, and continually build on their accomplishments. This project approach is flexible and encourages students to take pride in their work and their creativity. It also works well for independent study or distant learning.

Table 1 shows a sample schedule of Oregon State University's bachelor's degree in CP curriculum. This is just one possible arrangement of the required courses; others exist, as well as ones in which substitutions are made depending on the student's interests and the advisor's consent. Computer-intensive courses are distributed among all four years of study (in Table 1, they're depicted in bold). Suggested electives or substitutions include courses in computer interfacing, quantum mechanics, numerical solution of ordinary differential equations and related subjects, operating systems, software engineering fundamentals, rigid bodies, physical optics, thermal and statistical physics, and classical dynamics. As is true for all science majors, CPUG students graduate with 180 total credits (one credit equals 10 class hours), which is 12 fewer units than an engineering major. They take all but six credits of the standard physics major's courses, but do take 12 credits of computational physics courses that physics majors do not.

Our Classes

The following is a brief description of our five classes, with details included in the "Class Details" sidebar on page 72.

Scientific Computing I. Our introductory course, Scientific Computing I, is designed to provide freshmen and sophomores with basic computational tools and techniques. It is based on a project approach using Maple's problem-solving environment and Java. This is most students' first experience with visualization tools, the use of a cluster of workstations sharing a common file system, and the Unix operating system.

Table 1. Sample curriculum for the bachelor's of science degree; computer-intensive courses are shown in bold.

	Fall	Winter	Spring
Fresh. (46)	Differential Calculus	**Scientific Computing I**	**Introduction to**
	General Chemistry	Integral Calculus	**Computer Science I**
	Fitness/Writing I, 3	General Chemistry Perspective, 6	Vector Calculus I
	Perspective, 3		General Physics
	Computational Physics/		Fitness/Writing I, 3
	Computational Science Seminar		
Soph. (45)	**Introduction Computer Science II**	**Discrete Math**	**Scientific Computing II**
	Writing II, 3	Infinite Series and Sequences	**Linear Algebra**
	Vector Calculus II	General Physics	Applied Differential Equations
	General Physics	Perspective, 3	Introduction to Modern Physics
Junior (44)	**Computational Physics**	**CP Simulations II**	Periodic Systems
	Simulations I	**Data Structures**	Class/Quant Mechanics
	Computational Physics Seminar	Waves in 1D	Energy and Entropy
	Introduction to Probability	Quantum Measurement	Biology, 4
	Oscillations	Central Forces	Perspective/Elective, 3
	Static Vector Fields	Elective/Perspective, 3	
	Writing III/Speech, 3		
Senior (45)	**Numerical Linear Algebra**	**Advanced Computational**	**Thesis**
	Electromagnetism	**Physics Lab**	**Interactive Multi Media**
	Mathematical Methods	**Social & Ethical in**	**Computational**
	Elective, 6	**Computer Science**	**Physics Seminar**
		Electives, 6	Electives, 6
		Synthesis, 3	

We spend about one-third of the course on Maple and about two-thirds on Java. (Our course materials permit Mathematica as a substitute for Maple and Fortran90 as a substitute for Java.) A problem-solving environment like Maple is a friendly and quick way for students to begin scientific computation, and it is used in many regular physics courses. On the other end of the spectrum, a compiled language gets students closer to computers' actual workings, to the algorithms, to the applied mathematics, and to the tools of computational science. It also shifts the burden of proof that the answer is correct from the software to the programmer.

Our use of Java provides an object-oriented view toward programming (fusing methods with data), permits platform- and platform-independent applications, and emphasizes the Web as an integral part of scientific computing. Students get free compilers to use at home with the same packages we have in our labs (packages include JAMA, a Java matrix package, http://math.nist.gov/javanumerics/jama/; PtPlot, a 2D data plotter and histogram tool in Java, http://ptolemy.eecs.berkeley.edu/java/ptplot; and Java Printf/Scanf, http://braju.com/). We find Java's handling of precision, errors, variable types, and memory access superior to C's for scientific computing; moreover Java's platform and system independence provides a broader base, and an extended lifetime, for the educational software and programs we develop. In addition, and for these same reasons, an increasing number of scientific subroutine libraries and visualization packages are available in Java.

Scientific Computing II. The topics are computer hardware, algorithms, precision, efficiency, verification, numerical analysis, algorithm scaling, profiling, and tuning. These are studied in the context of specific applications including series approximations, integration rules, data fitting, Monte Carlo applications, differentiation rules, and differential equations. This course provides the basic mathematical, numerical, and conceptual elements needed for scientific computation.

Computational Physics Simulations. This sequence was the first course we developed. Topics covered vary from term to term based on students' interests, but the sidebar shows a typical selection. The course applies and extends the basic mathematical and numerical techniques introduced in Scientific Computing II to typical physical problems in classical dynamics, electromagnetism, quantum mechanics, and statistical mechanics.

Class Details

An asterisk (*) indicates that Web enhancements are available.

Physics/Mathematics/Computer Science 265, Scientific Computing I
This introductory course is designed to provide freshman and sophomores with the basic computational tools and techniques needed for courses in science and engineering. The course adopts a project approach to problem solving using symbolic and compiled languages with visualization. The one lecture and two labs per week cover Unix, Windows, basic Maple, number types, Maple functions, symbolic computing, visualization, calculus, equation solving, introductory Java limits, methods (functions), logical control, plotting loops, numerical integration, objects, complex arithmetic Web computing, applets, arrays, and file I/O.

Physics 365, Scientific Computing II
Unix Editing and Running*
Floating-Point Errors and uncertainties random walk, decay simulation,* limits, precision, under/overflows, matrix computing with JAMA libe, hardware basics, memory and CPU; tuning, Monte Carlo techniques, interpolation, cubic spline, least-squares fit, differentiation, and differential equations.

Physics 465, 466 Computational Physics Simulations
These two courses apply and extend the basic mathematical and numerical techniques introduced in Scientific Computing II to typical physical problems in classical dynamics, electromagnetism, quantum mechanics, and statistical mechanics. Topics include realistic/double pendulum, anharmonic oscillators, Fourier analysis of nonlinear oscillators, bugs, nonlinear mappings, chaotic pendulum/scattering, fractals, aggregation, trees, coastlines, bound states in momentum space, quantum scattering, integral equations, thermodynamics, the Ising model, quantum path integration, fluid dynamics, electrostatic potentials, heat flow, waves on a string, KdeV solitons, molecular dynamics simulations, and electronic wave packets.

Physics 417/517 Advanced Computational Laboratory
To experience computational solutions to real-world problems, advanced students experiment with computer simulations taken from previous MS and PhD research projects, as well as from national laboratories. Topics include radar maps of archaeological tells molecular dynamics simulations, meson scattering from nuclei in momentum space, quantum wavepacket-wavepacket interactions, serious scientific visualization, earthquake analysis, density functional theory of super lattices, Gamow & resonant states of exotic atoms, data analysis of pion form factor, fluid flow with particle hydrodynamics, principal component analysis of brain waves, and quantum chromodyanmaics.

As you can see, these types of realistic problems are often absent from a regular physics curriculum, with its emphasis on analytic techniques. Our applications often require numerically intensive computing with a compiled language and lead to research-level computing.

As is true for all our computation classes, this course is run with a combination of lectures and "over the shoulder" labs. The students discuss the projects with an instructor and then write them up as an "executive summary." The report contains sections for each project's problem, equations, algorithms, code listings, visualizations, discussions, and critiques. The emphasis is professional, much like reporting to a manager in a workplace. I have encouraged students to set up their own home pages on the department's Web site and prepare their reports as Web documents.

Advanced Computational Physics Laboratory. The newest (and still developing) component of our curriculum is an advanced computational laboratory. In it, senior CPUG students experiment with computer simulations taken from previous masters' and doctorate research projects, as well as from research projects at national laboratories. The scientific descriptions and actual computer simulations are modified to make the research experience accessible to undergraduate students in a short time (in contrast to the people-years required to develop the original research codes). The students get the codes running (which is not easy because many of the codes are hand-me-down Fortran), investigate some suggested problems, make some code modifications themselves, and then compare their results to those published in the literature. This is many students' first experience with truly large programs, old-fashioned Fortran, and reading an article in scientific literature.

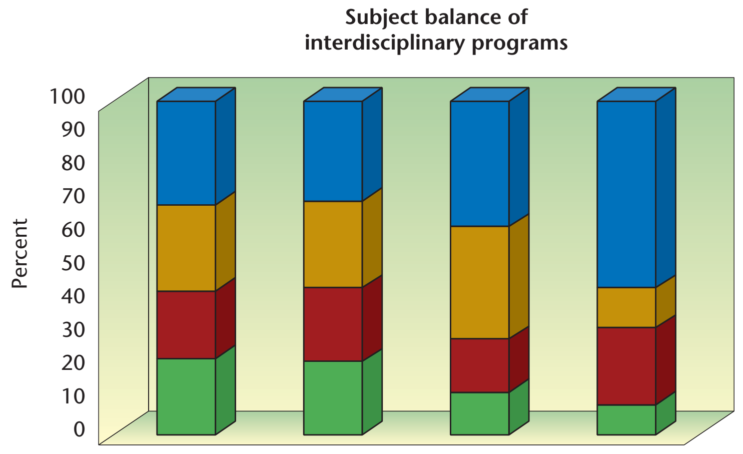
Our CP Seminar. Finally, our program includes a CP seminar that covers reports of modern happenings, campus research results, and journal articles. Sometimes the seminar is held in conjunction with our group meetings, sometimes it is part of the computer science department's orientation program, and sometimes we include appropriate physics department colloquia.

Computer-Mediated Learning and Accessibility

While we do not view the Web as a good teaching medium for general physics or for most college-age students (because it removes the essential interpersonal interactions and practical, hands-on experiences, and because the requisite level of maturity is not yet attained by many beginning college students), you cannot beat having a motivated student sit at a computer in trial-and-error mode to learn scientific computing.[17] The Web is an ideal environment for computational science: projects are always in a centralized place for students and faculty to observe, codes are there to run or modify, and visualizations can be striking, with 3D, color, sound, and animation.[18]

The CPUG texts and lectures contain several Web enhancements[7] that graduate students and faculty in the NACSE Research Group and those working with Manuel Páez at El Universidad de Antioquia, Colombia, developed. These enhancements demonstrate our belief that alternate viewing modes can improve students' understanding of complex and abstract materials.[19]

As an ongoing research project, and to connect with the future National Digital Library (www.dli2.nsf.gov/), we wish to advance digital books using multimodal and interactive elements to increase access to and understanding of mathematics and science. Specifically, we want to develop a *hybrid instrument* that incorporates a tutoring approach to teach objective materials along with computer simulations and embedded problem-solving environments to develop more tacit understanding. Accordingly, the CDs that accompany the published texts will contain interactive versions of the text materials in a variety of formats. There will be direct interaction with Maple, Mathematica, and XML/MathML materials on Java and Fortran (so that we run codes, interact with the figures, and manipulate the equations that occur in the text), and interactive scalar vector graphics (SVG) figures. Not only would this benefit disabled people, but it would also permit any reader to use a variety of senses to understand materials.



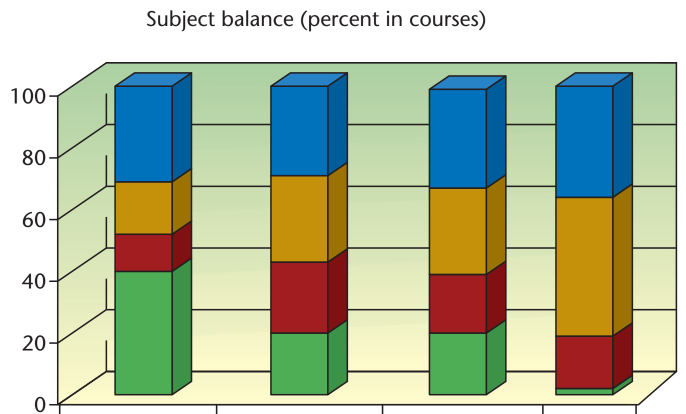
	CP (OSU)	CP (SUNY-Buffalo)	P+CSE (Clark)	M+CSE (Stanford)
🟦 Other	31	30	38	56
🟨 Application	26	26	34	12
🟥 Mathematics	20	22	16	23
🟩 Computing	23	22	13	9

Figure 2. The average percent of total curriculum dedicated to computing, mathematics, application, and other topics for bachelor's of science degree programs in (from left to right) CP at Oregon State University (OSU), CP at State University of New York at Buffalo (SUNY Buffalo), the physics plus computational science degree at Clark University, and the mathematics plus computational science degree at Stanford.

Comparison with Other Degrees

Figure 2 shows the average percent of the total curriculum dedicated to computing (green on the bottom), mathematics (red), application (golden) and other subjects (blue) for four programs. From left to right, the programs are the bachelor's of science degree program in CP at Oregon State University, CP at State University of New York at Buffalo, the physics plus computational science degree at Clark University, and the mathematics plus computational science degree at Stanford. Within a percent or two, the two CP degree programs have the same subject balance. The Clark and Stanford degree programs appear to have less emphasis on computing; however, just what courses students take in other fields of study can easily change that.

A recent paper I wrote with Yaşar[14] presents results of a survey of the various undergraduate programs throughout all the computational sciences. Figure 3, taken from this work, compares bachelor's of science programs in computer science, computational science and engineering, CP, and physics. Figure 3's left column shows the strong computing (green) but weak application (golden) components in the computer science degree; the right column shows the strong application but weak computing components in the physics degree. We see that an undergraduate degree in CP has a similar balance to one in computational science and engineering—namely, approximately equal weights for mathematics and computing (roughly 20 percent) and a higher weight for application (roughly 28 percent). This is a fairly uniform balance among components, and, as expected, a CP or computational science and engineering degree contains less physics than a physics degree and less computing than in a computer science degree.



	Computational science	Computational science and engineering	Computational physics	Physics
🟦 Other	31	29	32	36
🟨 Application	17	28	28	45
🟥 Mathematics	12	23	19	17
🟩 Computing	40	20	20	2

Figure 3. The average percent of total curriculum dedicated to courses in computing, mathematics, application, and other topics for bachelor's of science degree programs in (from left to right) computer science, computational science and engineering, computational physics, and physics.

Beginnings are hard. We have assembled a curriculum for a bachelor's of science in CP that focuses on a common "tool set" of subjects that have proven themselves useful in solving problems across several disciplines. Our examples and applications tend to be close to physics. While most courses students take are taught by traditional departments, our five computational classes serve to put the tools in perspective, promote a problem-solving viewpoint, glue the multipledisciplinary classes together, and promote a sense of belonging to a computational community.

Although there are benefits, such as longevity and focus, in institutionalizing a program such as ours in a distinct computational science department, local politics and tight budgets discourage that much of a change. Nevertheless, we have been encouraging other departments to set up related computational X (where X is the name of these other departments) programs, that possibly share many core courses. We have now begun to present workshop classes as a means of encouraging others to use our materials and improve our model to revitalize and modernize their offerings.

Only time will judge the viability of programs such as ours, but we do appear to be attracting some new students to our department and providing them with a broader preparation for future career choices. We have attempted to ensure some stability for our courses, and encourage similar courses elsewhere, by publishing the materials we develop as commercial texts with Web-based enhancements.[10,11] Further institutionalization demands the recruitment of additional faculty (something our physics department is now working on), adding line items in the budget for the program (critical support now comes from NSF grants), the development of similar programs in other departments so that we can benefit from the efficiency of number and need, improvements in computational support, incorporation of computational modeling and numerical simulation into more of the traditional physics classes, and graduate-level computational programs that build on the undergraduate ones. While I never expected that making a structural change to a traditional university would be easy (or necessarily welcome), it has been deeply satisfying to see a career's worth of research experience become part of the educational infrastructure. To me, it has been a "big thing"—to the students, it has been obvious.

References

1. D. Greenwell et al., eds., *Undergraduate and Graduate Education in Computational Science,* Louisiana State Univ., Argonne National Lab., April 1991.
2. R.H. Landau and M.J. Páez, *Computational Physics, Problem Solving with Computers*, John Wiley and Sons, 1997; www.physics.orst.edu/˜rubin/CPbook.
3. H. Gould and J. Tobochnik, *Introduction to Computer Simulation Methods, Applications to Physical Systems*, Addison-Wesley, 1996, http://sip.clarku.edu/.
4. P.L. DeVries, *A First Course in Computational Physics*, John Wiley and Sons, 1994.
5. M.L. De Jong, *Introduction to Computational Physics*, Addison-Wesley, 1991.
6. W.H. Press, "Book Review," *Physics Today*, vol. 51, no. 7, 1998, p. 71.
7. R.H. Landau, H. Kowallik, and M.J. Páez, "Web-Enhanced Undergraduate Course and Book for Computational Physics," *Computers in Physics*, vol. 12, no. 3, 1998, pp. 240–247; www.aip.org/cip/pdf/landau.pdf.
8. R.H. Landau, M.J. Páez, and C. Bordeianu, *Computational Physics, Problem Solving with Computers and Java*, 2nd ed., John Wiley and Sons, in preparation.
9. R.H. Landau, *Introductory Scientific Computing*, Princeton Univ. Press, in preparation.
10. "America's Best Colleges," *U.S. News & World Report,* 23 Sept., 2002.
11. *Skills Used Frequently by Physics Bachelors in Selected Employment Sectors*, Am. Inst. of Physics Education and Employment Statistics Division, Am. Inst. of Physics, 1995; www.aip.org/statistics/trends/emptrends.htm.
12. *Science and Engineering Indicators*, Nat'l Science Board, 1996, chapters 2–3.

13. C.D. Swanson, *Computational Science Education Survey*, Krell Institute; 2003; www.krellinst.org/learningcenter/CSE survey.

14. O. Yaşar and R.H. Landau, "Elements of Computational Science and Engineering Education," *SIAM Rev.*, vol. 45, no. 4, 2003, pp. 787–805.

15. O. Yaşar, "Computational Science Education: Standards, Learning Outcomes and Assessment," *Computational Science Int'l Conf.* (ICCS 2001), LNCS, vol. 1159, V.N. Alexandrov et al., eds., Springer-Verlag, 2001, pp. 1159–1169.

16. R. Root-Bernstein, *Discovering*, Random House, 1989.

17. P. Davis, "How Undergraduates Learn Computer Skills: Results of a Survey and Focus Group," *T.H.E. J.*, vol. 26, no. 69, 1999; www.thejournal.com/magazine/vault/A2063.cfm.

18. R.H. Landau et al., "Future Scientific Digital Documents with MathML, XML, and SVG," *Computing in Science & Eng.*, vol. 4, no. 2, 2002, pp. 77–85.

19. C. Dede et al., "Multisensory Immersion as a Modeling Environment for Learning Complex Scientific Concepts," *Computer Modeling and Simulation in Science Education*, N. Roberts, W. Feurzeig, and B. Hunter, eds., Springer-Verlag, 1999.

Acknowledgments

The National Partnership for Advanced Computational Infrastructure-EOT and the US National Science Foundation (under grant 998094) partially supports Oregon State University's Computational Physics for Undergraduates (CPUG) program.

Rubin Landau is a Distinguished Professor of Physics and director of the Computational Physics Program at Oregon State University. His research interests include few-body systems in subatomic physics, computational physics, and computational science education. He is a member and fellow of the American Physical Society and an officer of their Division of Computational Physics, the American Association of Physics Teachers, the IEEE Computer Society, the Society for Industrial and Applied Mathematics, and the American Association of University Professors. Contact him at rubin@physics.orst.edu; www.physics.orst.edu/~rubin/.

How to Reach CiSE

Writers

For detailed information on submitting articles, write to cise@computer.org or visit www.computer.org/cise/author.htm.

Letters to the Editors

Send letters to Jenny Ferrero, Contact Editor, jferrero@computer.org. Please provide an email address or daytime phone number with your letter.

On the Web

Access www.computer.org/cise/ or http://ojps.aip.org/cise for information about CiSE.

Subscription Change of Address (IEEE/CS)

Send change-of-address requests for magazine subscriptions to address.change@ieee.org. Be sure to specify CiSE.

Subscription Change of Address (AIP)

Send general subscription and refund inquiries to subs@aip.org.

Subscribe

Visit http://ojps.aip.org/cise/subscrib.html or www.computer.org/subscribe/.

Missing or Damaged Copies

If you are missing an issue or you received a damaged copy (IEEE/CS), contact membership@computer.org. For AIP subscribers, contact kgentili@aip.org.

Reprints of Articles

For price information or to order reprints, send email to cise@computer.org or fax +1 714 821 4010.

Reprint Permission

To obtain permission to reprint an article, contact William Hagen, IEEE Copyrights and Trademarks Manager, at copyrights@ieee.org.