INFUSING COMPUTATION THROUGHOUT THE UNDERGRADUATE CURRICULUM

Paul R. Ohmann, Adam S. Green, and Martin E. Johnston
University of St. Thomas
Department of Physics

Abstract

Over the past several years, the University of St. Thomas physics department has revamped its undergraduate curriculum in order to broaden the abilities of its graduates and provide them with a wider range of postgraduate career options. Our student learning objectives emphasize the acquisition of analytical, experimental, computational, and communication skills. While the first two areas are prominent in traditional physics curricula, the latter two — and especially computation — have justifiably received increasing attention. This paper describes the computational learning objectives of our curricular model, along with specific projects intended to develop student competencies in this area. Although we focus on the education of physics majors in our department, roughly half of our students are also engineering majors. Thus, our curriculum is integrated with engineering, computer science, and mathematics courses in order to provide a cohesive experience for our students. As such, the concepts and projects described in our paper are adaptable to a wide spectrum of science disciplines. Early assessment of our new curriculum's effectiveness has found that our recent graduates have been well-prepared for both graduate school and industry in a variety of fields. Moreover, the integrative nature of computation within the curriculum has led to greater collaboration among our faculty.

Introduction

The introduction of a significant computational component into the physics or engineering curriculum addresses a need of graduates who remain in a technical field. According to a recent American Institute of Physics report[1], the majority of physics bachelor's degree recipients employed in the technical fields of physics, engineering, or mathematics felt that their scientific research experiences, lab skills, and scientific software knowledge left something to be desired. (On a five point scale with 5 = excellent and 1 = terrible, a rating of 3 or lower was given by approximately 60% of the recipients regarding research experience and lab skills and 80% regarding scientific software.)

At St. Thomas, our research program goes a long way toward addressing these concerns. However, we feel we need to better connect our classroom material with the diverse set of technical skills that students will need in the future. Simulation packages, such as the innovative one that accompanies the book *Physlet® Physics*[2] do a wonderful job of helping students visualize difficult physical concepts by building on the power of computation, but they don't introduce computational techniques. As the National Science Foundation's 1996 "Shaping the Future" report[3] states, "…all students [should] learn these subjects by direct experience with the methods and processes of inquiry." Students certainly learn the material much more thoroughly by active participation in all aspects of the problem in question, including computational methods. The question, then, is how to best provide students with opportunities to design and implement their own codes.

Resources such as *Computation and Problem Solving in Undergraduate Physics*[4] by David Cook, *Numerical Methods for Physics*[5] by Alejandro Garcia, and *Computational Physics*[6] by Rubin Landau and Manuel Paez provide organizational structures for a focused course in computational physics. Similar initiatives have been taken in related science

disciplines[7]. But a single course is not usually sufficient for students to become highly proficient with computational methods. A number of programs have therefore sought ways of providing their students with more comprehensive and thorough introductions to computation. Brigham Young University, for example, offers a three-course computational laboratory sequence[8]; and other successful programs at Clark University[9], Lawrence University[10], and Oregon State[11] include computation throughout the entire curriculum.

We find such a wide-ranging approach very appealing. In the sections that follow, we describe the rationale and methods we have exercised to revamp our own physics program by including computation in nearly every course that we offer. For our introductory students, we have established a curricular partnership with our engineering, computer science, and mathematics departments in order to provide complimentary experiences that allow students to see the relevance of computation from a variety of perspectives. For our advanced students, we have developed a unique set of computational modules that are spread among all upper-division courses. These modules effectively elucidate and augment the more traditional analytical and experimental content of the physics curriculum.

Computation in the University of St. Thomas Physics Department

A decade ago, it became apparent that the curriculum in our physics department needed a significant overhaul[12]. At that time we had a very small number of majors, graduating approximately 1-3 each year, and we sensed a growing gap between the preparation of our students and their needs after graduation. We undertook major curricular reforms, implementing a workshop-style format in our introductory classical mechanics sequence and introducing a sophomore-level Methods of Experimental Physics class to give our students useful hands-on skills at an early stage. With several new hires and a growing number of majors, we implemented an undergraduate research program to show our students the relevance of what they learned in class and to inspire them to further develop their talents.

We also looked more closely at the needs of our graduates, most of whom attend graduate school (in a variety of fields), enter industry directly from college, or enter the teaching profession. Nearly all of our graduates needed a solid working knowledge of computational methods (although, as students, many of them probably did not realize it [1]), especially for the purpose of simulating physical situations and then generating and interpreting results appropriately. At that time, we required a programming class taught by our computer science department; and some of our students accumulated additional experiences in computation, either on their own or through undergraduate research. Yet many of our graduates fell through the cracks because we lacked a systematic way to give everyone a core competency in computation. In addition, the faculty of our department were concerned that our classes involved too many “spherical-cow” problems. In other words, the problems that we could attack through standard analytical approaches were often limited, creating a gap between classroom material and the real-world problems that our graduates would need to handle later. We sensed an urgency to make our classroom material more relevant to our students and at the same time improve the understanding of the physics involved. For all these reasons, we undertook an extensive effort to infuse our entire curriculum with computational problems and projects.

Our revamped curriculum is summarized in Figure 1. It is divided into a triad of experimental, analytical, and numerical experiences that develop student proficiencies in these areas. Each category incorporates the use of an appropriate software package: LabVIEW

	Experimental LabVIEW for interfacing, data collection, analysis	**Analytical** *Mathematica* for general homework problems	**Numerical** MATLAB for simulation projects
Freshmen	***Introduction to Classical Physics I & II*** ***Introductory engineering courses*** (engineering department)	***Calculus I & II*** (math department)	***Problem Solving in the Natural Sciences*** (computer science department) general programming exercises in C and MATLAB
Sophomores	***Modern Physics I & II*** ***Methods of Experimental Physics*** ***Introduction to Electronics*** (engineering department)	***Modern Physics I & II*** ***Differential Equations & Linear Algebra*** (math department) ***Calculus III*** (math department)	***Modern Physics I & II*** Rutherford Scattering Module ***Methods of Experimental Physics*** Chaotic Pendulum Module
Juniors/Seniors	***Optics***	***Advanced Mechanics*** ***Electricity & Magnetism I & II*** ***Optics*** ***Statistical Mechanics*** ***Quantum Mechanics***	***Advanced Mechanics*** Gravitational Three-body Module ***Electricity & Magnetism I & II*** Finite Line of Charge Module ***Optics*** Light Scattering Modules ***Statistical Mechanics*** Statistical Ensemble Module ***Quantum Mechanics*** Hydrogen Orbital Module

Figure 1: Outline of computational experiences that students accrue as they pass through our curriculum.

for experimental interfacing, data acquisition, and analysis; *Mathematica* for analytical manipulation and solving differential equations; and MATLAB for numerical simulations (which we will call "computation" in this paper). Taken together, these three platforms comprise a complete set of tools that can be used to investigate a wide variety of problems. In addition, their prominence in both industry and academia will allow students with experience in these platforms to immediately begin making contributions in their work after graduation. This is facilitated through repeated exposure to these software packages. Hence, our goal is to use them consistently and appropriately throughout the curriculum, recognizing the strengths and limitations of each one. Students thus gain a solid working knowledge of each software platform by the time they graduate, and they know which tools work best for future problems that they encounter.

In the experimental domain, exposure to LabVIEW begins in the standard freshmen Classical Physics sequence. At this level, students interact with Vernier sensors using a LabVIEW interface. Although they do not write their own programs, they do learn LabVIEW's basic structures and functions. As sophomores, they modify existing LabVIEW codes in Modern Physics I and II, and then they gain a much more in-depth knowledge of LabVIEW and computer interfacing in Methods of Experimental Physics as well as Introduction to Electronics (the latter requirement is taught by our engineering department). In the Methods course, students follow nearly the entire *Advanced LabVIEW Labs* book written by John Essick[13]. The upper-level Optics course then reinforces students' knowledge of LabVIEW through its use in several experiments.

Mathematica is introduced in the Calculus sequence by our mathematics department. In our physics curriculum, we build upon this knowledge by incorporating *Mathematica* as a tool to help solve differential equations, integrals, and other analytical equations in the context in which they arise—usually in weekly homework problems. This is an ongoing process in our curriculum, with the goal of infusing the use of *Mathematica* throughout our courses, but with care to avoid an over-reliance on this tool at the expense of analytical skill development.

In the numerical or computational part of our triad, we use MATLAB as a common platform for three major reasons: its ease and versatility as a programming language, its readily accessible visualization tools, and its wide acceptance in both industry and academia. Students first learn MATLAB in a new introductory freshmen computer science course called Problem Solving in the Natural Sciences. Within the physics curriculum, MATLAB is introduced to sophomores in the Modern Physics and Methods of Experimental Physics courses. By this time, they are prepared to tackle the many computational modules we have developed and spread throughout our curriculum. These modules, described in section IV, give students valuable and repeated experience in translating material from a wide variety of courses into numerical simulations and then gleaning useful information from them. By graduation, students have enough experience and confidence with computational simulation in different contexts that they recognize—without assistance from a professor—when and how to use them.

Computational Competencies

In order for students to have proficiency in computation—that is, to be able to analyze a physical system in terms of a computer simulation that they create—students need to acquire several specific skills. These include simulation techniques and algorithm design, visualization, and validation of the results.

Simulation involves the representation of a physical system through a set of equations that are then solved with computational techniques. Pedagogically, our goal is for students to see the close connection between a physical system and its model, and to be able to construct these simulations for a wide variety of situations. Algorithm design involves the ability to formulate a procedure to numerically solve the equations underlying the simulation. While algorithm design is the subject of typical computer science classes, we have found that our students are much more motivated if the algorithm design process is closely coupled with system simulation, and if both are done in the context of an interesting scientific problem. Hence, it is natural to introduce these computer projects within our physics classes themselves rather than to simply expect that students will emerge from computer science requirements with extensive knowledge of the techniques involved. In the computational modules in our physics courses, we give students experience working with numerical integration and differentiation techniques as well as Monte Carlo methods, as described in the next section.

Visualization is an important component in the analysis of simulations, for it allows results to be interpreted much more easily than by simply sifting through numerical outputs [14]. Our computational projects therefore require the use of plots or other appropriate visual representations of the simulation. Students validate their simulation results through comparisons with either experimental data or information found in published articles. Having students read through research literature themselves gives them exposure to a discipline-specific style of written communication as well as the standards required to publish scientific results. It also gives students confidence that they have gained enough knowledge to contribute to, or at least understand, the scientific endeavor.

Specific Projects

This section describes the development of computational skills as students progress through our curriculum, highlighting the specific computational projects that they encounter. We describe these in the order that most students take them.

Freshmen

Along with the standard Classical Mechanics sequence, most of our freshmen physics and engineering majors take the new Problem Solving in the Natural Sciences class from our computer science department. This course ensures that they have a working knowledge of programming and algorithm development in both C and MATLAB, and it focuses on specific science and engineering problems that students might encounter later. In addition, students gain experience with *Mathematica* through the calculus sequence in the mathematics department.

Sophomores

Sophomore students typically take the year-long Modern Physics sequence in which they are introduced to computational methods with a Rutherford Scattering Module as a three-week experimental/computational lab. They also take the course Methods of Experimental Physics which focuses on the construction, analysis, and simulation of a chaotic pendulum.

The Rutherford Scattering Module has two components. The laboratory component is based on a fairly standard Rutherford scattering apparatus consisting of a scattering chamber, detector, and associated electronics[15]. Using a small vacuum chamber, an alpha particle source, a gold foil, and a particle detector connected to timer/counter, students measure the number of counts per time interval N as a function of scattering angle θ. We have the students perform this measurement between 0° and 30° inclusive, and verify the well-known analytical relationship between N and θ.

The computational portion of the module has three goals: introduction to the MATLAB computing environment, introduction to Monte Carlo techniques, and application of Monte Carlo techniques to Rutherford scattering. The MATLAB component is introduced through a tutorial and series of exercises adapted from Rudra Pratap's book *Getting Started with MATLAB*[16]. After these rudiments are mastered, the Monte Carlo technique is introduced through an exercise to determine the value of pi to a variety of precisions using different numbers of random points. The students are then encouraged to heuristically discover the relationship between the number of points used in the Monte Carlo simulation and the precision of the result (the error scales as $1/\sqrt{N}$, with N being the number of points). Having achieved a familiarity with the Monte Carlo technique and with MATLAB, students then simulate Rutherford scattering using a Monte Carlo to calculate the number of particles scattered as a function of angle. These results are compared to the experimental measurements. Moreover, the Monte Carlo analysis is used to obtain the predicted scattering at angles beyond 30°, which is the practical experimental limit in our current setup.

For many students, the Rutherford Scattering Module is their first exposure to numerical methods, and some are intimidated by the prospect. In our view, this discomfort substantiates our plan to provide repeated exposure throughout the physics curriculum. We also observe in this class the benefit of using a consistent computing platform (like MATLAB) on which to perform numerical computations. The time needed to obtain a working knowledge of the particular computing environment is considerable, and retention certainly improves after multiple encounters.

In the Experimental Methods course, students build on their computational skills by using MATLAB to analyze data collected from a chaotic physical pendulum. The pendulum incorporates a stepper-motor-controlled eddy drive, an electro-magnetic brake, and an optical encoder for sensing position. Students design and implement LabVIEW programs to control and read data from the apparatus. Data collected during non-chaotic operation of the pendulum is analyzed using MATLAB to determine physical characteristics of the system such as drive torque and frictional damping.

With this information, students develop numerical simulations of the system and use them to predict the chaotic nature of the pendulum. To probe their understanding of the physics, they compare their experimental data with simulation using MATLAB codes. The comparisons can be simple visual analyses of Poincaré sections at different phases of the simulated and experimental data, or they can be detailed calculations of fractal dimensions or bifurcation diagrams of the various states of the system. The project as a whole is complicated and involved—nothing is a "black box"—and it takes an entire semester to complete. By the end of the semester, students are competent at writing sophisticated codes and have confidence that serves them well as they move forward in the major.

Juniors/Seniors

Most of our junior-senior courses are taught every other year because of enrollment constraints. Consequently, our computational modules in these courses need to be self-contained, with only the experience gained during students' freshmen and sophomore years as a prerequisite. Specific computational modules in these upper-division classes are described below. Most of these modules are adapted from published articles and books because we want students to become familiar with the literature in our discipline and gain confidence that they can read, interpret, and make use of the information contained therein, with only broad guidance from the instructor. Consequently, the assignment of each module is accompanied by an information sheet that sketches the system to be modeled, the references to the relevant articles/books, and general guidelines on how to get started. In each case, students are expected to prepare a 4-6 page report describing their simulation and the background material needed to understand the simulation (e.g., Monte Carlo methods). Students are encouraged to talk to one another and emulate collaborative scientific research; however, projects must be handed in by each individual student.

In our Advanced Mechanics course, students are assigned a Three-Body Gravitational Simulation Module which demonstrates how an analytically difficult problem can sometimes be solved more readily with computational techniques. The particular problem explored here is a natural extension of the classroom discussion on central forces and orbital motion[17]. Students readily understand the basic concepts of this problem as well as the difficulties with approaching it analytically. With that background assumed, the goal of this module is to extend the study of orbital motion to the Sun-Earth-Moon system. Students are encouraged to use the "Euler method" of numerical integration[18] as a guide. Using initial conditions inferred from common

knowledge of the Sun-Earth-Moon system, students numerically integrate the equations of motion, plot the results, and compare them to observations in order to visually check whether the orbits of the Earth and Moon are stable. The actual numerical integration is straightforward, though care must be used (as students soon discover) to apply a sufficiently small integration step size if stable solutions are to be generated. With this module, students become aware of the subtleties involved in such a calculation—subtleties that would be very difficult to convey with classroom exercises using only analytical techniques. The project leads naturally to further classroom discussions regarding the difficulty in estimating future asteroid impacts on Earth given an initial set of observations. This latter topic can also make an interesting and relevant computational project in its own right [19].

The Electricity and Magnetism Module is an adaptation of work by Griffiths and Li modeling the charge density on a conducting needle [20]. While introductory physics students learn in electrostatics that any net charge possessed by a three-dimensional conductor resides on its surface, the static distribution of charge on a one-dimensional conductor is less simple to describe. For a finite straight wire, it is tempting to think that the analog of "surface" is "endpoints," but it is reasonably intuitive that the charge does not all gather at the endpoints. Instead, the charge distributes *uniformly* in the limit that a three-dimensional object shrinks to a one-dimensional finite line [21]. Griffiths and Li describe two approaches to numerically modeling a finite straight wire; but the model that interests us most for the purposes of this module is their fixed-position bead model, in which a number of point charges with varying magnitudes are fixed along a line segment. This approach yields a system of linear equations that is readily soluble computationally. The results obtained vary, not surprisingly, with the number of charges N used in the model. As N increases, the model yields a nearly uniform charge distribution, but it indicates greater charge densities at the endpoints. Because of the very slow convergence of the charge distribution as N approaches infinity, the increased density at the endpoints appears to be substantive; but the endpoints actually contain a decreasing fraction of the total charge[22].

This system is an appealing computational modeling exercise for a number of reasons. It is conceptually simple, but the results are not obvious. The modeling is straightforward and should not pose major difficulties. The slow convergence as N increases forms the basis for important lessons about modeling a continuum with discrete particles, and it highlights the caution one must take in interpreting model results. Perhaps most important, this module gives students experience in reading published scientific journal articles, following the arguments described therein (even if some of the mathematics are beyond them), and pulling out the relevant equations with some guidance from the instructor.

In our Optics course, we currently assign two computational modules, both of which deal with light scattering. The first module involves a Beer's law experiment where students shine laser light through a milk sample to determine its fat concentration by monitoring changes in transmitted irradiance. For single scattering, the transmission follows the familiar exponential decay with increasing fat concentrations. To build students' intuition about this phenomenon, we have them simulate the system by writing a simple MATLAB routine that performs a numerical integration and compares their results to the analytical expression. Then they perform the experiment using an adaptation of the apparatus described by DiLisi *et al.* [23]. The students investigate several milk products with different fat concentrations in order to observe the usefulness of Beer's law at low concentrations (single scattering) and its failure at high concentrations (multiple scattering) [24]. A numerical fit to the single-scattering region of a natural log plot serves as an apparatus calibration that can later be used to accurately determine the amounts of fat in "unknown" milk samples with fat concentrations less than about

13% by weight. We are currently upgrading this module to incorporate multiple Mie scattering into a more sophisticated Monte Carlo simulation and thus provide a single calibration curve for the entire range of fat concentrations in milk. The model will be used in simulations of laser-tissue interactions for biomedical applications, and it will be based on Monte Carlo routines written by Prahl and Jacques [25].

In the second optics module, students simulate the formation of rainbows. An initial homework assignment asks them to derive, using Snell's law and wavelength dispersion, analytical expressions for the angles at which sunrays emerge from a spherical water droplet to create primary and secondary rainbows. Then they use electromagnetic boundary conditions to calculate irradiances for those rays. The computer simulation incorporates a Monte Carlo routine that is similar to the ones suggested by Olson *et al.* [26] and by Hill [27]. Impact parameters for light rays and water droplets are randomly generated, as are several color hues that represent different wavelengths of light. A graphical display is created such that the rainbows are viewed relative to the antisolar point. In the future, we will ask students to also perform a more detailed analysis of the light's polarization and plot a map of the degree and orientation of polarization for the entire rainbow. When incident sunlight is unpolarized, light forming the rainbow is strongly linearly polarized tangent to the arc. According to the Fresnel coefficients, the degree of polarization depends on both wavelength and scattering angle.

Our Thermodynamics/Statistical Mechanics module illustrates the statistical basis of thermodynamics by having students explicitly model the statistics of an ensemble of dice, a system that they intuitively understand. We base this module on an adaptation of a microcanonical Monte Carlo technique to model the underlying statistics [28]. In this simulation, the values of each die correspond to the possible energy states of an atom. Initially, the ensemble is partitioned so that some atoms are at a very high energy state while most are at the ground state. As time progresses, these atoms interact, and energy is exchanged between them through a series of Monte Carlo steps. The simulation conserves system energy (or in our case, the sum of the dice), allowing quick generations of configurations with constant energy. From these configurations, the entropy can be statistically determined along with various thermodynamics quantities such as energy densities in each portion of the ensemble. By observing the change in entropy as the system progresses, students determine (1) entropy never decreases with time (consistent with the second law of thermodynamics); (2) energy densities in each portion of the ensemble converge with time; and (3) the statistical basis for the Maxwell-Boltzmann distribution. We have found that this module serves as a valuable complement to in-class lectures that may tend to be more mathematical. Students report that the simulation provides much of the intuition they need to fully understand the theoretical concepts.

Our Quantum Mechanics module has students visually characterize the orbital structure of the hydrogen atom for quantum numbers n, l, and m (through $n = 3$) using the von Neumann accept/reject technique [29]. The orbital structure in each case is visualized as a density function scatterplot, with dot density representing the probability of finding the electron at that location. In this implementation, randomly selected points are used to test whether the appropriate probability density function is large enough to represent that position with a dot. As with other Monte Carlo techniques, the multitude of randomly selected points leads to a plot that visually describes the hydrogen atom in that quantum state.

Assessment

The assessment of our computational modules and their integration within our curriculum

comprises both informal assessment by faculty working with students on research projects and more formal assessment through short- and long-term surveys. In our undergraduate research program, our faculty have been able to gradually assume a level of computational competency in students at the sophomore level and above. Previously, substantial investment in time was often needed to bring undergraduate research students to a sufficient level to be able to meaningfully contribute to research projects. In computation-intensive research projects, these modules have substantially reduced this ramp-up time, just as the introduction of the Methods of Experimental Physics course in the sophomore year has substantially reduced the amount of time needed for students to contribute meaningfully to lab-based research projects.

Our short-term surveys ask our junior/student students in each course to describe their experience with the computational modules contained therein. Specifically, we inquire about the degree of integration with the course material, the gain in computational skill development, and the general level of interest in the module.

Surveys of the thermodynamics/statistical mechanics, electromagnetism, and quantum mechanics modules have revealed that these projects do reinforce the relevant concepts from these courses while also building computational skills. Not surprisingly, survey results stress the importance of smoothly connecting the module to the classroom content. A typical response explains, "I found [the project] helpful, and liked actually doing something hands on to help learn the material. The assignment made you…really think about whether you understand the topics discussed in lecture."

We find it important to supplement these short-term surveys with long-term ones from alumni who have a few years' perspective on their education. Since our computational modules have been fairly recently incorporated, most of our alumni have not yet had the breadth of computation that our current students are experiencing. Consequently, we plan on formally introducing these long-term surveys during the next few years. The surveys will ask alumni to describe the usefulness of their computational experiences in their current endeavors and to suggest ways for us to fine-tune our curriculum.

During the last several years, we have informally queried our graduates about the original set of modules we developed. These graduates include many students who currently attend well-known graduate schools in technical fields (e.g., Caltech, Cornell, Wisconsin-Madison) and others who work for major corporations (e.g., 3M, Lockheed-Martin). They consistently stress the value of repeated experiences with computational projects as a way to build competency in this area. In addition, a number of our present students have remarked at the usefulness of their computational skills in the external REUs (Research Experience for Undergraduates) in which they have participated.

Conclusions

We have developed a curriculum that has infused computation throughout the student experience. This work has required a common vision among faculty in the department and close collaboration with faculty in engineering, computer science, and mathematics. Our philosophy is centered on the importance of building computational competency as one leg of the experimental, analytical, and numerical triad of skills that physics majors should develop. Observations of our students' achievements in external REUs, acceptances into highly regarded graduate programs, and industrial job offers suggest that this curriculum is giving our graduates the preparation they need to succeed in their future endeavors.

Acknowledgement

This work was supported in part by the National Science Foundation grant #DUE-0311432.

References

1. R. Ivie and K. Stowe, "The early careers of physics bachelors," American Institute of Physics publication no. R-433, http://www.aip.org/, August, 2002, (accessed January 2007).

2. W. Christian and M. Belloni, *Physlet® Physics: Interactive Illustrations, Explorations and Problems for Introductory Physics*, Addison-Wesley, Menlo Park, CA, 2004.

3. Advisory Committee to the National Science Foundation Directorate for Education and Human Resources, "Shaping the Future, New Expectations for Undergraduate Education in Science Mathematics, Engineering, and Technology," filename nsf96139.txt, http://www.nsf.gov, 1996, (accessed January 2007).

4. D. M. Cook, *Computation and Problem Solving in Undergraduate Physics*, *Vol. 1 and 2*, self published, 2004.

5. A. L. Garcia, *Numerical Methods for Physics*, 2nd ed., Prentice Hall, Englewood Cliffs, NJ, 2000.

6. R. H. Landau and M. J. Paez, *Computational Physics*, Wiley, New York, 1997.

7. R. A. Layton, "Using modeling and simulation projects to meet learning objectives in an upper-level course in system dynamics," *Computers in Education Journal*, vol. 14, no. 2, 2004.

8. R. L. Spencer, "Teaching computational physics as a laboratory sequence," *Am. J. Phys.*, vol. 73, no. 2, 2005.

9. C. A. Manogue, P. J. Siemens, J. Tate, K. Browne, M. L. Niess, and A. J. Wolfer, "Paradigms in physics: a new upper-division curriculum," *Am. J. Phys.*, vol. 69, no. 9, 2001.

10. H. Gould, "Computational physics and the undergraduate curriculum, "*Computer Physics Communications*, vol. 127, no. 1, 2000.

11. D. M. Cook, "Computation in undergraduate physics: the Lawrence approach," in Part 1 of *Proc. Int'l Conf. Computational Science (ICCS)*, edited by V. N. Alexandrov et al., Springer-Verlag, 2001.

12. M. E. Johnston, "Implementing curricular change," *Computing in Science and Engineering*, vol. 8, 2006.

13. John Essick, *Advanced LabVIEW Labs*, Prentice-Hall, Upper Saddle River, NJ, 1998.

14. E. R. Tufte, *The Visual Display of Quantitative Information*, 2nd ed., Graphics Press, Cheshire, CT, 2001.

15. We use the Rutherford scattering apparatus manufactured by Leybold Didactic GmbH. The equipment and experiment are described in several documents that can be downloaded from http://www.leybold-didactic.com, (accessed January 2007).

16. R. Pratap, *Getting Started with MATLAB 7: A Quick Introduction for Scientists and Engineers*, Oxford University Press, New York, NY, 2006.

17. G. R. Fowles and G. L. Cassiday, *Analytical Mechanics*, 6th ed., section 7.4, Saunders College Publishing, Philadelphia, PA, 1999.

18. R. A. Serway, *Physics for Scientists and Engineers with Modern Physics*, 4th ed., Saunders College Publishing, Philadelphia, PA, 1996.

19. D. R. Allen, "Earth's 'other moon': an exercise in computational dynamics," presented at the American Association of Physics Teachers Summer Meeting, Greensboro, NC, 2007 (unpublished).

20. D. J. Griffiths and Y. Li, "Charge density on a conducting needle," *Am. J. Phys.*, vol. 64, no. 6, 1996.

21. J. D. Jackson, "Charge density on a thin straight wire, revisited," *Am. J. Phys.*, vol. 68, no. 9, 2000.

22. O. F. de Alcantara Bonfim and D. J. Griffiths, "Comment on 'charge density on a thin straight wire, revisited'," *Am. J. Phys.*, vol. 69, no. 4, 2001.

23. G. A. DiLisi, C. M. Winters, L. A. DiLisi, and K. M. Peckinpaugh, "Got milk? A Beer's law experiment," *Phys. Teach.*, vol. 43, 2005.

24. V. P. Dick, "Applicability limits of Beer's law for dispersion media with a high concentration of particles," *Appl. Opt.*, vol. 37, no. 21, 1998.

25. Scott Prahl and Steven Jacques of the Oregon Health and Sciences University have written several Monte Carlo routines that are available for download at http://omlc.ogi.edu/software/ (accessed January 2007). Some of their programs are accompanied by tutorials.

26. D. O. Olson, C. Brozovich, J. Carr, H. Hatton, G. Miles, Jr., and G. Zwicke, "Monte Carlo computer simulation of a rainbow," *Phys. Teach.*, vol. 28, 1990.

27. D. R. Hill, http://mathdemos.gcsu.edu/ math demos/ (accessed January 2007). This Web site gives an explanation of the physics and a description of an example program that is available for download.

28. Koo-Chul Lee, "How to teach statistical thermal physics in an introductory physics course," *Am. J. Phys.*, vol. 69, no. 1, 2001.

29. V. M. de Aquino, V. C. Aquilera-Navarro, M Goto, and H. Iwamoto, "Monte Carlo image representation," *Am. J. Phys.*, vol. 69, no. 7, 2001.

Biographical Information

Paul Ohmann is currently an Associate Professor of Physics at the University of St. Thomas. He received his Ph.D. from the University of Wisconsin-Madison in 1994. After postdoctoral fellowships in high-energy physics at DESY and the University of Oxford, Dr. Ohmann worked at Lockheed-Martin Corporation for several years on radar and sonar applications. He joined the faculty at the University of St. Thomas in 2000, where he has worked on a variety of interests related to computation.

Adam Green has been an Assistant Professor of Physics at the University of St. Thomas since 2003. His Ph.D. work at the University of Nebraska focused on spin-dependent scattering of electron beams from chiral molecules. His present research deals primarily with polarimetric thin-film analyses of iridescent insects and with spectropolarimetric imaging through turbid media such as fog, murky water, and human tissues.

Martin Johnston is an Associate Professor of Physics and serves as departmental chair. He obtained his Ph.D. in 1993 from the University of California–Riverside working on the ionization of metastable neon. He joined the University of St. Thomas in 1995 following a postdoctoral appointment at the University of Nebraska. His current efforts center on understanding nonlinear systems, specifically chaotic magnetic pendulums and timing structures in sonoluminescence.