Computational Physics and the Undergraduate Curriculum

Harvey Gould

Clark University, Department of Physics, Worcester, MA 01610

Abstract: The advantages of a separate undergraduate course emphasizing computer simulations are discussed. Also discussed are the advantages of Java and the timeliness of integrating aspects of computational physics into the undergraduate curriculum so that the curriculum reflects the way physics is currently done.

PACS: 01.40.Gm, 01.50.Ht

1. Introduction

There is much interest in incorporating developments in computational physics into the undergraduate and graduate physics curriculum. We believe that these developments should be guided by the following general principles:

- Computation has led to important conceptual advances and new ways of thinking about physical systems.
- Changes in the curriculum and innovations in educational methods should be guided by developments in physics and physics education research.
- Our goal should be to incorporate computational methods into the curriculum rather than computers in the classroom.
- New courses in computational physics emphasizing computer simulations, numerical methods, and symbolic manipulation have been developed and many new texts are available. Now our goal should be to integrate computational physics into the entire physics curriculum so that the curriculum reflects the way physics is done by physicists in academia and industry.
- Computational physics does not yield instant gratification as is found in many other computer applications. We have to provide opportunities for students to understand that computing does not lessen the need for thinking deeply and that such thinking has its own rewards.

As computational physicists, we know that there have been many exciting developments in algorithms, computer-based models, and their applications. In the following, I discuss how these developments might be incorporated into the physics curriculum.

2. Undergraduate Level Textbooks

Interest in computational physics has led many physicists to write textbooks that emphasize some aspect of computational physics. According to our count, there were four textbooks published before 1980, six textbooks in the 80's, and at least 24 undergraduate level texts published in the 90's.[1] (We have omitted textbooks that emphasize the use of computers in areas such as the real-time control of laboratory experiments.) At this rate, we will soon have more textbooks than students to read them. These texts can be roughly grouped into three classes depending on their emphasis: simulation, numerical methods, and symbolic manipulation.

If we accept the assumption stated in Section 1 that our goal should be to incorporate computational methods into the curriculum, what should be our first priority? (There is frequently pressure from administrators to introduce computers into the classroom so that it looks like the faculty is doing something new.) Ideally, we should incorporate computational methods into every course in the physics curriculum. And while we are thinking of the ideal, we also should incorporate laboratory experiments into every course. However, we are far from this ideal and the next best situation is to offer a separate course on computational physics taken by students as early as possible in their studies. Just as students take calculus during their first year and use it in almost all their physics courses thereafter, students should take a computational physics course also. In general, a beginning course in computer science is not a good substitute because it usually is too general and teaches programming without a meaningful context.

So given our assumption that a computational physics course should be taken by students as early as possible, what should be its emphasis? Our second assumption, which has been tested by us and many others over the past fifteen years, is that the course should emphasize computer simulations. The advantages and disadvantages of such a course are discussed in the following.

3. Computer Simulation Laboratory at Clark University

At Clark we offer a course entitled *Computer Simulation Laboratory* to both undergraduates and graduate students. The advantages of such a course include:

- Computer simulations provide an opportunity for involving students in open-ended problems and letting them do physics closer to the way research is done. (There are no answers in the back of the book.) Students who are outstanding in the usual classroom are not necessarily the best students in this more open-ended environment. The course is an excellent predictor for how graduate students will do in research.

- Doing simulations provides a way of reaching a deeper understanding of fundamental physical concepts, particularly by writing programs with graphics and user interaction. The act of converting an abstract model into a working program makes the model more meaningful. As researchers, we do not understand an algorithm until we have written a program that implements and tests it.
- Numerical methods are more meaningful when part of a simulation, than when taught only as a tool. As an example of the difference in emphasis in a first course, we discuss the order of the algorithms used to convert Newton's equation of motion to finite difference equations, but do not derive the result. Instead we let the students find the order of the various algorithms empirically.
- The approach is similar to laboratory experiments, is project oriented, with a minimum background in either physics and programming required. Students write laboratory reports, an excellent vehicle for improving their writing.
- Simulations allow open-ended questions and encourage creative thinking in contrast to memorization and routine problem solving.
- Simulations encourage a broader vision of physics than is usually seen in undergraduate courses. Students can study models of interest to geologists, biologists, material scientists, and social scientists, and the course can attract non-physics majors. However, because of the open-ended nature of the course, the course is only for motivated students.
- Students might reform the curriculum. Once students know how to write and test their own programs and know some important algorithms, they will use the computer in meaningful ways in their other courses even if their instructors do not encourage it.

Of course there are disadvantages associated with any change. These disadvantages include the following:

- If we add a computational physics course, what course do we eliminate? Should the course be required of all physics majors? Our answers are that if the course can be structured to appeal to students in other departments, then during this time of decreasing numbers of physics majors, we need to do all we can to attract students to our courses, especially if they are intellectually challenging. Ideally, all physics majors should take a computational physics course so that more advanced courses can incorporate related assignments. However, at a liberal arts institution such as Clark where students receive a B.A. rather than a B.S. in physics, we need to limit the number of required courses. So instead we require one laboratory course, either Electronics or Computer Simulation. Almost all students take both anyway, because of their interest, faculty encouragement, and because they learn skills that are useful in the marketplace.

- Open-ended laboratory courses are time consuming, and the same is true of the Computer Simulation Laboratory. The help of a teaching assistant is essential. Although such courses are labor intensive, we do not foresee them being replaced by a Web-based course.

- It is more efficient to incorporate computation into the entire curriculum than to do so in a separate course. The difficulty is that such a reform assumes that the faculty can act coherently, an assumption that applies to only a few departments. Even so, a course in computational physics would be desirable.

- An argument against introducing computation too soon is that it would encourage students to neglect the development of their analytical and problem solving skills. Although it is true that today's students have weaker analytical skills, we have not observed that a laboratory-based simulation course leads to this problem. For example, students quickly find that although they can gain an intuitive understanding of chaotic systems, they cannot gain a deeper understanding unless they learn more physics and mathematics.

- Frequently students do simulations for which the answers are not known to them or the instructor or maybe not to anybody else. Teaching simulations in an open-ended context forces us to change the way we teach. We will leave it to the reader to decide whether this change is an advantage or disadvantage.

One area in which the use of computers has been a double-edged sword is graphing. We used to have students plot a graph by hand and do a visual fit before we let them do least squares fits on the computer. But we have given up and students would fit an elephant by a least squares fit if we let them. It takes all semester to get some students to think about their graphs and the significance, if any, of their fits.

4. **Typical topics**

As mentioned above, one advantage of simulations is that there is a wide variety of possibilities that are accessible and interesting to students with varying backgrounds. As an example, we will briefly survey some of the simulations that students have done this semester at Clark.[2] As usual, students enjoy simulating simple chaotic systems. We introduce Monte Carlo methods in some simple physical contexts such as the approach to equilibrium and random walks, and then introduce importance sampling in the context of estimating one-dimensional integrals. Random walks illustrate the difference in approach. The dynamics of ink in water could be modeled by the diffusion equation $\partial P(x,t)/\partial t = D\,\partial^2 P(x,t)/\partial x^2$ and implemented on a computer using finite difference methods. The treatment of boundary conditions is nontrivial. Instead, the approach we use is to simulate a random walk on a lattice using Monte Carlo methods which students readily understand. The treatment of boundary conditions is simple using this approach. That is, we can solve the diffusion equation numerically or use random walks to simulate diffusion on a lattice.

Once students understand random walks, it is not that difficult for the more advanced students to do quantum Monte Carlo in simple contexts and diffusion-limited aggregation, etc. A junior in economics is doing a simulation of a percolation model of the stock market.[3] Several computer science majors are using genetic algorithms. Other projects include the simulations of the planar model and use of the relaxation method to obtain numerical solutions of Laplace's equation.

5. **Programming Languages**

The choice of a programming language is of interest to many people. The glib answer we used to give was that students' first language should be "anything but Fortran" (Fortran 77). However, Fortran 90/95, or better F,[4] is an excellent first language. The main points to emphasize are that there are many languages, none of which is perfect, and that once students learn one language well, they can learn others quickly. The features of a programming language which are important to us include portability, low cost, simple syntax, built-in graphics, and a library for numerical functions. We also want the language to be structured, preferably object-oriented, allow event-based programming, to be useful outside of physics so that the language will be maintained and improved and provide a marketable skill for students, allow bit manipulation capability, and have parallel programming capability or an easy route to a language which does.

We made the choice of using True BASIC many years ago when most computer scientists were using Pascal in beginning courses and most physicists were using Fortran 77. True BASIC has a clean, easy to learn syntax, and makes it easy to do graphics in a platform independent way. We have learned how to teach it in the context of learning physics so that students have very little trouble with the syntax, and most importantly, students have no difficulty making the transition to C or F.

However, True BASIC does not have the same appeal as it once did, and it is important to keep up with developments in programming languages. So our choice of programming language now is Java, especially beginning with Java 1.1.[5] We are finally using the same language as the computer scientists. Java has all the features listed above, although most students still find it more difficult to learn than a procedural language. Our challenge is to learn how to teach Java so that its object-oriented nature is easy to grasp. In the meantime we have found its object-oriented nature to be very convenient for doing graphics and interactive programming.

6. Integrating Computational Physics into the Curriculum

Now that there are many computational physics texts available, our challenge should be to incorporate computational physics throughout the curriculum. What should we do?

At the introductory level, we can have students solve Newton's second law numerically and do some simple Monte Carlo sampling, for example, random walks. However, such add-ons would be as ineffective as many traditional labs are now. We need to make fundamental changes in the introductory course, a change that will not happen soon.

In upper division courses, the nature of the supplementary material is more obvious. For example, in classical mechanics, we can simulate nonlinear systems and few body motion, and in electricity and magnetism, we can discuss relaxation methods and random walk solutions to Laplace's equation and numerical solutions of Maxwell's equations. However, our experience is that these applications work only if the students already know the answers or if they have taken a computational physics course. So the argument for the latter is compelling.

One area in which simulation can make a big difference is thermal and statistical physics, probably the area of physics which has been most affected by computer technology. The very nature of probability, which requires a series of measurements, is made more meaningful by making many measurements in the context of a simulation. We can simulate simple models which illustrate the approach to equilibrium and the increase of entropy, do microcanonical simulations (using molecular dynamics or the demon algorithm) to compute subsystem probabilities and motivate the Boltzmann probability, compare different ensembles by doing various Monte Carlo simulations and molecular dynamics, and compare time averages to ensemble averages. We also can use the broad histogram and other new Monte Carlo algorithms to improve student understanding of the meaning of the density of states in systems other than the ideal gas. Equally importantly, we can consider applications we would not be able to consider otherwise. Examples of student interest include maximum entropy and image enhancement and traffic flow.

All of the above simulations and applications can be discussed without using the computer at all. Just talking about the implementation of the demon algorithm makes the microcanonical ensemble more meaningful. A good test of the meaningfulness of a possible application is whether it is useful to discuss it without actually implementing it on a computer.[6]

We recently have obtained a NSF grant[7] to develop Web-based curricular materials for such a course. We expect to have many Java applets, QuickTime movies, and hypertext with many applications. All of our material will be "open source" so that users will be able to adapt it to their own needs and make improvements. Our immediate goal has been to finish editing a theme issue on thermal and statistical physics for the American Journal of Physics.[8] Interestingly, few of the submitted manuscripts involve computation.

7. Summary

Although we have ambitious plans to enhance the curriculum in thermal and statistical physics, our most important goal is to help develop a community of teachers and students to generate course materials and exchange ideas in an open source environment. In this way, our individual contributions as teachers can be more coherent and make an impact beyond our own institutions.

In our efforts to incorporate recent developments into the physics curriculum, we might ask the question, "How can the computer be used to teach physics?" We think a better question is, "How can we teach the student to 'teach' the computer?" Our goal should be to give students the tools and the background so that they can learn physics by reading, studying, solving problems, doing experiments, and doing simulations. Nobody said that learning physics is easy.

Acknowledgements

I thank Jan Tobochnik for many useful conversations on the topics discussed here and for colloborating with me for many years. This work was supported in part by National Science Foundation grant PHY-9801878. This article is based on an invited talk given at the March APS meeting, sponsored by the Division of Computational Physics and CCP99.

References

[1]A list of computational physics books is available at http://sip.clarku.edu/books.html. The SiP acronym stands for "Simulations in Physics."

[2]The textbook used for the course is H. Gould and J. Tobochnik, *An Introduction to Computer Simulation Methods*, 2nd e., Addison-Wesley (1996).

[3]R. Cont and J. P. Bouchaud, cond-mat/9712318. I learned about this model from D. Stauffer, "Interdisciplinary Applications of Computational Statistical Physics," Amer. J. Phys. **67**, 1207 (1999).

[4]F is a subset of Fortran 90 which discards the latter's obsolete elements. Information on F can be found at http://www.imagine1.com/. A tutorial on F can be found at http://sip.clarku.edu/tutorials/F_tutorial.html.

[5]See for example, J. Tobochnik, "Teaching Students How to Write Computer Simulations in Physics," *Comp. Phys. Comm.*, **121-122**, 562 (1999).

[6]Thinking about how a problem can be formulated on a computer can be very rewarding. See for example, K. G. Wilson, *Rev. Mod. Phys.* **55**, 583 (1983).

[7]Information about the STP project can be found at http://stp.clarku.edu. The STP acronym stands for "Statistical and Thermal Physics" or "Standard, Temperature, and Pressure."

[8]Theme issue on thermal and statistical physics, December, 1999.