TEACHING COMPUTATIONAL PHYSICS AS A LABORATORY SEQUENCE

Ross L. Spencer
Department of Physics and Astronomy
Brigham Young University
Provo, Utah 84602

Abstract

The undergraduate curriculum in most physics departments is so full that adding new courses is difficult, but the training of an undergraduate physics student is greatly enhanced by learning computational methods. The standard method of addressing this need at most universities is to offer one 3-credit hour computational physics course, either early on as an introduction, or later, when it can address more advanced problems. In the Department of Physics and Astronomy at Brigham Young University we have chosen to use the standard 3 credit hours allotted to computational physics in a different way. Computational methods are taught as 3 separate 1-credit hour laboratories, one for sophomores, one for juniors, and one for seniors. (The usual semester load of a physics student is 14-16 credit hours.) Students are introduced to symbolic methods using Maple when they are sophomores, and to numerical methods using Matlab beginning in their junior year. This helps prepare the students for their upper division courses, prepares them for the research they will do for their senior project, and spreads computational methods throughout the undergraduate curriculum.

I. INTRODUCTION

Physics, like other mathematically developed sciences, has a very full curriculum because students need to learn challenging material dating back at least to the 1700s. These old techniques are still important, so ancient material can't just be removed from the course list to be replaced by the most recent advances. So even though computational methods have been important in physics for more than 30 years, the physics departments that teach them (about half in the United States, according to an Internet search of a wide sample of departments) only offer one course on numerical methods. A particularly well-developed example of such a single course using a laboratory setting is the course developed over a span of nearly 20 years by Gould and Tobochnik[1,2]. (Two notable exceptions are the interdisciplinary computational physics degree at Oregon State University, in which computational methods are employed throughout the curriculum.[3,4], and the CCLI Project at Lawrence University, which integrates computation with undergraduate coursework[5–7].)

At Brigham Young University we began teaching computational physics the way most departments do, by teaching computational physics late in the curriculum when the students' experience allowed advanced problems to be tackled. We offered a 3-credit hour course in numerical methods at the senior/first-year graduate level with an emphasis on grid methods for solving partial differential equations, using the excellent book by Alejandro Garcia which taught us, among other things, the value of Matlab as an introductory programming language.[8] We learned two things by teaching this course. (i) It came so late for our undergraduates that little connection between these methods and either their course work and or their research projects was possible. (ii) Lecturing and then sending students off to do their homework mostly created frustration, mostly because many of our students have poor programming skills when they start our program. Students studying this subject simply need lots of help from the instructor to be able to develop a good programming style and debugging skills. The lecture/homework style also made it difficult to survey enough methods to give the students a broad background.

To solve the second problem we started using a laboratory format so that while the students were working at computers doing their homework the instructor and some teaching assistants were present to help them with debugging and to answer questions. (A survey of computational courses around the United States, and the world, shows that other departments use this solution too.) In this format the instructor gives a short introduction to the subject for the day, after which the students go to work. Occasional mini-lectures are also given during the lab period to enliven the course, to anticipate difficulties, and to provide breadth.

We were pleased to find that teaching this subject as a lab creates an active learning environment in which students are exposed to problems, help each other solve them much of the time, and are ready to listen when the instructor has something to say[9]. This use of an active learning environment worked so well that we decided to use it to solve problem (i) as well by abandoning our 3-credit-hour lecture course and putting in its place three 1-credit hour lab courses: Physics 230, 330, and 430. (The usual semester load of a physics student is 14-16 credit hours.)

Physics 230 is designed for sophomore physics students and teaches them basic computational skills (with emphasis on symbolic methods) using Maple. In the process the students review the mathematical methods they learned in their first year and apply them to physics problems from their first and second year courses. The textbook for the course is a Maple worksheet titled *Introduction to Maple for Physics Students*. This course will be discussed in Sec. II. (A similar electronic tutorial in Mathematica is included in Dubin's textbook[10] and a printed Mathematica tutorial with similar physics examples is in De Jong's book[11].)

Physics 330 is designed for juniors and focuses on ordinary differential equations with applications mostly from classical mechanics (and especially nonlinear dynamics). The students start by learning to solve differential equations in Maple, using *Introduction to Maple for Physics Students* again (we skip this topic in Physics 230 because the students typically haven't studied differential equations yet.) While they are enhancing their Maple skills, they are also learning the basics of Matlab, culminating in learning how to use its

differential equation solvers. Along the way the students review classical mechanics and are introduced to some of the basic ideas of nonlinear dynamics, as discussed in Sec. III. Other textbooks which cover the same material as Physics 330, but not in laboratory format, are listed in the bibliography[10,12–21].

Physics 430 is designed for seniors, and is discussed in Sec. IV. The emphasis is on methods for partial differential equations and we concentrate on grid methods, using Matlab. These methods are illustrated with examples involving wave motion, diffusion, Schroedinger's equation, and two-dimensional electrostatics. Other textbooks which cover the same material as Physics 430, but not in laboratory format, are listed in the bibliography[10,12–16,19,20,22–24].

These courses are relatively new and are still being refined, but our students give them good marks when they evaluate the courses. The students are especially pleased with Physics 230 because of the power it gives them early in their studies to solve difficult problems in their other physics courses. In fact, most of our students now use Maple as their standard mathematical handbook. When students become involved in research (now a graduation requirement in our department) they often use both Maple and Matlab in their work. We have not as yet, however, performed any systematic study of student outcomes to assess the quality of these courses. The texts for these courses are freely available at http://www.physics.byu.edu under the Courses link.

To compare the approach described here with what is being done elsewhere see the extensive listings of articles and books on computational physics at Kalamazoo College[25] and Clark University[26].

II. PHYSICS 230: MAPLE AND SOPHOMORE PHYSICS

The goal of the Maple worksheet *Introduction to Maple for Physics Students* is to help the student develop the ability to solve difficult problems with a symbolic computing engine (Maple in our case), and to review the basic mathematics of their first two years of study.

The text is awkward to use in printed form; it is designed to be read by someone sitting in front of a computer on which Maple is installed.

For example, when the students learn to integrate in Maple they are given problems involving integration over a known charge distribution to find the electrostatic potential and the electric field, and they also integrate over mass distributions to find moments of inertia. When the students learn to use Maple to do sums they review the basic ideas of wave interference, including 2-slit interference and beats, using Maple's powerful graphic capabilities. There is also a rather long exercise on wave packets in which the students produce Maple animations of packets in dispersive media, illustrating the ideas of phase and group velocity. In this unit they also work through the basics of the uncertainty principle, allowing them to see that the limitation on the product $\Delta\omega\Delta t$ is not a quantum principle but a wave principle.

We also introduce them to some of the more difficult mathematical ideas and functions that they will encounter in their third and fourth years. They plot Bessel and Legendre functions and are introduced to the idea of orthogonal functions and Fourier analysis as they learn to make plots and perform integrations in Maple.

The classroom environment consists of a room filled with computers and staffed by an instructor and one or more teaching assistants. The lab period lasts for 3 hours. Typically each student sits in front of their own computer, but we encourage them to work as lab partners because with this helps both students to learn more effectively. The instructor and the student teaching assistants roam the room answering questions and helping the students overcome programming and conceptual errors. The instructor also gives mini-lectures about the physics that will be covered that day and anticipates difficulties. When the students discuss what they are doing with a partner, read the text aloud to each other, and frequently call the instructor or an assistant over for help, the classroom becomes an active learning environment in which doing difficult mathematics for three hours is a lot more fun than it sounds like it would be. The students don't print their results because printing wastes paper and slows them down. Instead, when they finish an assigned problem, the instructor

or a teaching assistant sits with them for a few minutes listening to them explain what they did, asking questions, and correcting errors or misconceptions. During the semester three examinations are usually given and the students also take a final examination, often in oral format. A brief list of the topics in the nine chapters of the Maple text are given in Table 1.

Table 1: Overview of Physics 230

Chapter	Topics
1	Introduction and using the Maple interface
2	Plotting (and review of elementary functions)
3	Calculus: limits, derivatives, integration, sums
4	Complex arithmetic and functions, including branch cuts
5	Linear algebra: linear systems, eigenvalues, vector calculus
6	Solving equations, linear and nonlinear
7	Differential equations (postponed to Physics 330)
8	Logic, loops, and procedures
9	Review of symbolic algebra commands in Maple

In addition to helping the students handle difficult problems, Physics 230 is also an excellent preparation for our standard mathematical methods course on solving partial differential equations using separation of variables and orthogonal functions. Because our students already know how to use Maple, the mathematical methods course uses it extensively. Note, however, that Physics 230 is not a mathematical methods course. The emphasis is on learning to write and debug Maple code, and the mathematics and physics ideas in the course are mostly a review of material previously covered in other courses.

The Maple text *Introduction to Maple for Physics Students* is publicly posted at the Physics 230 link under Courses at http://www.physics.byu.edu. Solutions to the problems in the text are available from the author upon request.

III. PHYSICS 330: DIFFERENTIAL EQUATIONS AND DYNAMICS

There are three texts for this course: *Introduction to Maple for Physics Students*, *Introduction to Matlab* (88 pages), and *Computational Physics 330* (49 pages), the last two posted in PDF format at the Physics 330 link under Courses at http://www.physics.byu.edu. The first is the same Maple worksheet used for Physics 230, but in this course the material on differential equations that was skipped in Physics 230 is covered. The second text is a blend of a Matlab tutorial and a brief reference manual, and the third is a laboratory manual which gives the students the assigned tasks for each lab period, and explains some of the physics ideas that will be encountered.

The classroom environment is the same as that discussed in Sec. II for Physics 230, except that the instructor has to lecture a little more because the physical and mathematical content of the course is more advanced. Most of the labs begin with a 15-minute lecture by the instructor. Perhaps the best way to show how the course is organized is to list each lab with a short description

Table 2: Overview of Physics 330

Laboratories	Topics
1	Differential equations (Maple) and start learning Matlab
2	Differential equations (Maple), more Matlab
3	Driven damped harmonic oscillator (Maple)
	Linear algebra and polynomial fitting (Matlab)
4	Phase space (Maple) and loops and logic (Matlab)
5	Numerical differentiation and integration (Matlab)
6	Fast Fourier transform, uncertainty principle (Matlab)
7	Parametric instability (Mathieu's equation) using Maple
8	Pendulum (Maple) and solving nonlinear systems of equations (Matlab)
9	Chaos: entrainment, intermittency, period-doubling (Matlab)
10	Coupled non-linear oscillators (entrainment) using Matlab
11	Dynamically stabilized pendulum (two-time scale analysis) (Matlab)
12	Gravitational motion, Kepler's laws (Matlab)
13	Hysteresis and nonlinear resonance (Maple and Matlab)
14	Hysteresis and nonlinear resonance continued

As they work through these projects the students use the power of Matlab graphics to see the self-similarity of fractals and to be introduced to the basic ideas of dynamical chaos. Each of the last 5 labs is a long single project, including the non-linear synchronization of two weakly coupled oscillators (Huygen's famous neighboring-clocks on the wall problem) and the stabilization of a vertical pendulum by rapidly vibrating its point of support. The students also use graphics and the numerical solution of differential equations to illustrate and review all of the standard analytic machinery of 2-body gravitation, including a numerical verification of Kepler's laws. Finally, they spend two weeks working through the hysteresis that occurs when nonlinear damped oscillators are externally driven.

This list of topics may seem overly ambitious for a class that only meets for three hours

each week, but the goal of the course is for the students to learn to write and debug code, not to master the intricacies of dynamics. We assume that the students either have already taken, or are currently enrolled, in our junior-level classical mechanics course, so the physics topics in each lab are either a review of material already covered, or a supplement to it.

Those who have taught this course have found that it is especially important for the instructor to carefully monitor the progress of the students during the first 4 labs, to help students not to become discouraged. Students who have no programming background struggle at the beginning and many just give up unless they receive special attention. If it becomes clear that some students are not going to be able to finish a laboratory, the instructor often helps the students catch up by doing part of a laboratory as a class project by lecturing at the board and having the students write the code as it is being explained. Lots of help from the instructor and from teaching assistants (almost exclusively alumni of the course) is essential for all of the students in the course to have a successful experience.

The last two weeks of the course involve a long and difficult project in nonlinear dynamics. Alternatives to this final project might be to either stretch out the first 4 labs into 5 or 6, to help students with a weak programming background come up to speed, or perhaps to have the last two labs involve projects of the students choice.

Solutions to the lab problems for Physics 330 are available from the author upon request.

IV. PHYSICS 430: PARTIAL DIFFERENTIAL EQUATIONS

The text for this course is a lab manual titled *Computational Physics 430* (88 pages) which contains both physical explanations and descriptions of computing techniques. The emphasis is on finite-difference methods using uniform grids and time stepping. All programming is done in Matlab, and the same active learning environment described in the previous two sections is used in this course. Because extensive programming is required in this course several Matlab scripts are provided to the students for them to use as a template as they

program. Key sections in these templates are often left blank so that the student can program the most important parts of the algorithms being used, but details like building grids and making plots are often supplied. The topics covered in each lab are listed in Table 3.

Table 3: Overview of Physics 430

Laboratories	Topics
1	Spatial grids in one and two dimensions
2	Finite-difference approximations to first and second derivatives
3	Converting differential equations into linear algebra problems
4	Resonance and steady state for the wave equation
5	Vibrations of a hanging chain and quantum bound states
6	Time-stepping: the wave equation and the staggered leapfrog algorithm
7	Two-dimensional wave equation via staggered leapfrog
8	The diffusion equation, explicit method
9	Diffusion using an implicit algorithm (Crank-Nicholson)
10	Animating Schroedinger's equation using Crank-Nicholson
11	Two-dimensional electrostatics using Successive Over-Relaxation
12	Two-dimensional electrostatics using Successive Over-Relaxation
13	Advection and introduction to one-dimensional gas dynamics
14	Advection algorithms applied to solitons (Korteweg-deVries)

This course is a companion to electrodynamics, quantum mechanics, and thermal physics. We assume that students either have had courses in these subjects, or are currently enrolled, so that the physics ideas can be reviewed and illustrated rather than fully developed. The unifying idea in this course is the power of linear algebra to effect numerical solutions of difficult problems. This turns out to be a challenge to most of our students because linear algebra seemed rather remote and abstract when they saw it for the first time. But it comes alive for them when they use it to find the vibration frequencies of a hanging chain and

to compute quantum bound state energies in potential wells for which no analytic solution is available. They then use linear algebra to develop implicit algorithms for the diffusion and time-dependent Schroedinger equations, culminating in an animation of a wave packet tunneling through a barrier and an animation of solitons forming and "colliding" with each other.

As with Physics 330, the involvement of the instructor in each lab is crucial, especially at the beginning where some students become discouraged and drop out. Each lab begins with a lecture, and extensive coaching by both instructor and teaching assistants is required throughout the lab period. Because of the more advanced nature of the problems in this course, the instructor usually requires the students to have read through the assigned laboratory before class, an activity that is encouraged by giving a short quiz on the reading at the beginning of each class period.

The last two labs are interesting applications of the grid methods developed in the course, but are not crucial to flow of the course. Some instructors might want to stretch the first few labs out and eliminate these last two, or perhaps replace them with labs that introduce other computational topics such as Monte-Carlo techniques and particle simulations, or allow these last two lab periods to involve student-designed projects.

A drawback of this lab format is that it is difficult to fully cover the theory behind the algorithms that are presented. Students who are interested in a more complete treatment are encouraged to take a course on numerical analysis from the Department of Mathematics. Another concern is that this course focuses exclusively on finite-difference methods using uniform grids, ignoring finite-element methods, non-uniform grids, Monte-Carlo methods, particle simulation methods, etc.. We hope, however, that with this introduction behind them students will be able to better learn other methods on their own as they encounter them. Finally, there is also the potential problem that programming in the Matlab environment is different from programming in a compiled language. We feel, however, that Matlab is an excellent preparation for learning a compiled language, and find that knowing Matlab is a good introduction to other computer languages.

Solutions to the problems in these labs are available from the author upon request, and the lab manual is freely available in PDF format at the Physics 430 link under Courses at http://www.physics.byu.edu.

V. CONCLUSION

Computational physics is taught as a sequence of three 1-credit hour labs instead of as a lecture course at Brigham Young University. We find that this allows us to introduce computational methods earlier in the curriculum and to give a broader overview of this large field. It also allows computational methods to be emphasized over a period of three semesters instead of just one, which helps the students to become more proficient at programming and debugging. The students use the symbolic computational skills they learn in these courses to do their homework in our traditional physics courses, and they use both symbolic and numerical methods in their research, gaining a set of skills that will help them succeed after they graduate.

We do not regard this lab sequence as a substitute for a rigorous course in numerical analysis. Students interested in developing true expertise in computational methods are encouraged to take one, or more, traditional courses in this field. But we find that these labs allow all of our students to develop basic skills in this important area.

VI. ACKNOWLEDGMENTS

The author thanks Branton Campbell, Grant Hart, Bryan Peterson, S. Neil Rasband, and Jean-Francois Van Huele for helpful comments and suggestions in the development of these courses.

[1] H. Gould, Computer Physics Communications **127**, 6 (2000).

[2] J. Tobochnik and H. Gould, "Teaching Computational Physics to Undergraduates," in *Annual Reviews of Computational Physics IX*, D. Stauffer, ed. (World Scientific, Singapore, 2001).

[3] Rubin H. Landau, Oregon State University, Computational Physics for Undergraduates program, http://www.physics.orst.edu/CPUG/

[4] R. H. Landau, H. Kowallik, and M. J. Perez, "Web-Enhanced Undergraduate Course and Book", Computers in Physics, **12**, 240 (1998)

[5] D. Cook, "Computers in the Lawrence Physics Curriculum - Part I", Computers in Physics **11**, 240 (1997)

[6] D. Cook, D. Cook, "Computers in the Lawrence Physics Curriculum - Part II", Computers in Physics **11**, 331 (1997)

[7] D. Cook, *Computation and Problem Solving in Undergraduate Physics*, (self-published, http://lawrence.edu/dept/physics/ccli/).

[8] Alejandro L. Garcia, *Numerical Methods for Physics* (Prentice Hall, New Jersey, 1994, 2000).

[9] Bonwell, Charles C. and James A. Eison, *Active Learning: Creating Excitement in the Classroom* (ASHE-ERIC Higher Education Report No. 1, Washington D. C. : The George Washington University, School of Education and Human Development, 1991).

[10] D. Dubin, *Numerical and Analytical Methods for Scientists and Engineers Using Mathematica*, (Wiley-Interscience, New Jersey, 2003)

[11] M. L. De Jong, *Mathematica for Calculus-Based Physics*, (Benjamin Cummings, 1999).

[12] G. Baumann, *Mathematica in Theoretical Physics*, (Springer-Verlag/Telos, Santa Clara, CA 1996)

[13] F. Cap, *Mathematical methods in physics and engineering with Mathematica*, (Chapman and Hall/CRC, London, 2003)

[14] P. L. DeVries , *A First Course in Computational Physics*, (Wiley, New York, 1994)

[15] R. N. Enns and G. C. McGuire, *Nonlinear Physics with Mathematica for Scientists and Engineers*, (Birkhauser, Boston, 2001)

[16] R. Gass, *Mathematica for Scientists and Engineers*, (Prentice Hall, New Jersey, 1998)

[17] H. Gould and J. Tobochnik, *Introduction to Computer Simulation Methods*, (Addison Wesley, 1995)

[18] R. L. Greene, *Classical Mechanics with Maple*, (Springer, New York, 1995)

[19] T. Pang , *An Introduction to Computational Physics*, (Cambridge University Press, 1997)

[20] F. J. Vesely , *Computational Physics: An Introduction*, (Kluwer Academic/Plenum, New York, 2001)

[21] R. L. Zimmerman and F. I. Olness, *Mathematica for Physicists*, (Addison Wesley, Boston, 2002)

[22] N. J. Giordano, *Computational Physics*, (Prentice Hall, New Jersey, 1997)

[23] R. H. Landau and M. J. Paez, *Computational Physics, Problem Solving with Computers*, (Wiley, New York, 1997)

[24] S. M. Wong, *Computational Methods in Physics and Engineering*, (World Scientific, New Jersey, 1997)

[25] J. Tobochnik, http://www.kzoo.edu/sip/articles.html

[26] H. Gould, http://sip.clarku.edu/books/