Technology in Calculus

– A Simple Approach

Draft[1] – October 17, 2002.

Todd Young

DEPARTMENT OF MATHEMATICS, OHIO UNIVERSITY, ATHENS, OH 45701
E-mail address: young@math.ohiou.edu

[1]Dilbert cartoons in drafts used by permission of United Media.

Preface

It is universally acknowledged that a new technology is in want of an educational application. Software companies, computer manufacturers, and college administrators all want us to use technology "in the classroom", but why and how we are to do so are questions about which we have not reached consensus.

I am a technology skeptic who has become convinced that computational software is an innovation that we should incorporate in the teaching of undergraduate mathematics. Many others have come to this same conclusion, but for a wide variety of reasons. One aim of this book is to promote reasons that are based on mathematics rather than pedagogy. It is my hope that these arguments for use of technology will appeal to a wide cross-section of mathematicians.

Even among those who agree that computational software has a role to play in undergraduate math, opinions vary widely about how it should be used. This book promotes ways to use computers in a simple, efficient manner. If computers are to be used widely in our courses, resource constraints demand that we be efficient. To do so we must take a simple approach.

Acknowledgment

Thanks go to several people who took the time and initiative be the first to incorporate technology into the mathematics curriculum at institutions where I happened to be studying or working, and whose hard work started me thinking about these issues. These include Jim Wells at the University of Kentucky, Fred Andrew and Tom Morley at Georgia Tech and Leonard Evans of Northwestern University.

At Ohio University I have received much invaluable help and constructive criticism from my colleagues, as well as general support from the Department of Mathematics and Ohio University. Thanks go to Steve Chapin, Jeff Connor, Klaus Eldridge, Barbara Grover, S.K. Jain, Winfried Just, Gene Kaufmann, Sergio Lopez, Jim Shultz, Larry Snyder, Mary Ann Swardson, and Paul Szeptytcki for many good ideas. Special thanks go to Michael Saum and Tsun-ho Liu for solving all my technical problems, Lindsay Eynik for proofreading and assistance, and to Heather Krugman, Debbie Pack, Ennice Sweigart and Jan Williams for administrative support.

I have discussed these issues with a number of colleagues at professional meetings. These professors have mostly been research mathematicians with strong interests in teaching. They are at a wide variety of institutions, and many of them have taken part in introducing software in their own departments; some are even authors of textbooks or supplements for calculus with computational software. These people have both contributed to and corroborated the ideas I will present.

The manuscript was completed while I was a visitor at the Institute for Physical Sciences and Technology at the University of Maryland. Thanks go to Jim Yorke and the staff of IPST for their hospitality.

Finally and most importantly I would like to acknowledge the contribution of my wife, Mary Beth Brookshire Young. Many, if not most, of the ideas in this book originated with her. They were first tried in her classes and were regularly discussed at dinner. This book should rightly have been jointly written, but she has left mathematics for a career in Law. While I am confounded as to why anyone with her talent for mathematics would choose to do anything else, but I am still happy to have her as a partner.

Contents

CHAPTER 1

Introduction

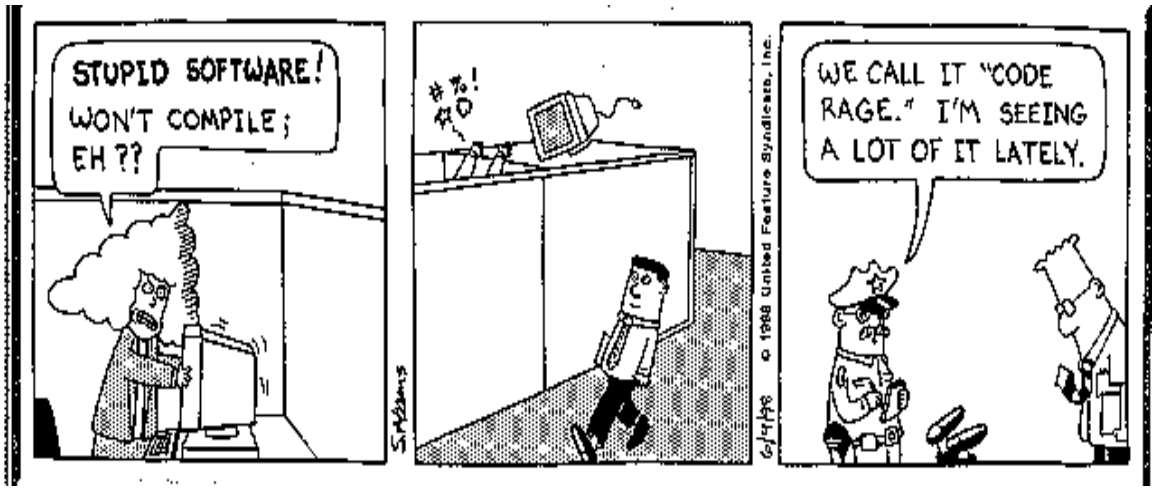
> "I believe that the motion picture is destined to revolutionize our educational system and that in a few years it will supplant largely, if not entirely, the use of textbooks."
> – Thomas Edison, 1926, quoted from [**Kob96**, p. 13]

1. A dilemma

Many mathematics professors find themselves in a dilemma. For one reason or another they feel pressured (internally, as well as externally) to employ some sort of computer technology in teaching calculus. Yet the prospect of doing so is not at all attractive to them. These mathematicians (particularly the ones who are the most prolific researchers) believe that using technology in teaching, while perhaps worthwhile in theory, would be too time-consuming in practice. They accordingly view introducing technology as an unwelcome distraction from other duties, particularly, their research. Their apprehensions are well founded. Many of those who have pioneered the use of technology in teaching have done so only with very large investments of time and effort, often with little recognition or reward. A colleague of mine gave a talk a couple of years ago with a very revealing title: "How to teach with technology and still have time to eat, sleep and enjoy the other pleasures of life" [**Jus98**]. The belief that using technology in teaching would be a large time-drain is probably the biggest reason that technology has not been more widely used.

A primary goal of this book is to alleviate this fear by suggesting ways to simplify and improve the use of technology in the teaching of undergraduate mathematics. I will explain how to incorporate technology in undergraduate math courses with very little time investment on the part of the instructors, and I will argue that to do so is worthwhile. The type of technology that I have found can be incorporated in teaching math in an *easy and worthwhile* way is computational software (programs such as AXIOM, DERIVE, MAPLE, MATLAB or MATHEMATICA) and the use of such software is the emphasis of this book.

While fear of excessive time-commitment is likely the most common reason mathematicians shy away from using technology, there are other concerns as well, each of which this book hopes to address. For example, many professors are simply not adept enough with computers to feel comfortable using them in front of an audience. Believing that to use computational software, one must use it in the classroom, either by holding class in a computer lab or by having a projector in the classroom that the instructor uses for demonstrations, they do not incorporate computers in their classes at all. This



situation is reflected in the unfortunate phraseology “technology in the classroom.” But technology in teaching is *not* limited to technology in the classroom. Indeed, I will argue that the best use of technology is outside of the classroom. Everyone agrees that simply doing demonstrations for the students is inadequate, and that they must also do things themselves to learn. At most universities holding all calculus classes in computer labs is not possible, so one is left with the only practical alternative: having the students do computer work on their own, i.e. as homework. I propose that for most instructors, just giving the homework alone is the easiest *and best* alternative. Some faculty (especially those who have taught in a computer-equipped classroom) will object that students would not be able to use the software on their own unless they have supervision by an instructor in a lab, or at least in-class demonstrations. My colleagues and I have actually given computer homework without any lab or demonstration, and I hope to convince the reader that it is possible, provided one takes the right approach to the homework.

There are also many mathematicians who simply don’t think that incorporation of technology in undergraduate education is very important. These mathematician often are skeptical about the hype surrounding computers in education, or even critical of the sometimes exaggerated claims of technology’s proponents. They feel that the content of the existing calculus sequence is very important and that using time for technology implies omitting parts of the existing course. I agree with them that the contents of calculus are important and often hard to cover in the available time. I will argue, however, that using a simple approach to technology can take very little class time and thus does not require any major deletions from the existing syllabus. I will further seek to convince this group that incorporating technology can have positive and important benefits. My arguments will be based on mathematical, not pedagogical, considerations. In fact, I am very skeptical about any actual pedagogical advantages of using any technology. Regardless of whatever advantages a technology may or may not have purely as a means of instruction, I am convinced that computational technology and the ideas required to use it well are themselves mathematically important.

There are still other mathematicians who feel that the introduction of technology is actually detrimental to the educational process. At one time I was a member of this set. Although most of these people probably won't read this book, I intend to address their concerns as well. Many in this group bemoan their students' lack of computational skills and blame the use of calculators for these shortcomings. One of my colleagues at Ohio University recently wrote an email addressed to all the faculty to this effect. He stated that he was going to ban the use of calculators altogether because the machines hindered the students' ability to do computations by hand. Another colleague, Jim Shultz, in the College of Education at Ohio, had a very nice reply to this objection: "Aren't you going to ban pencil and paper as well? They are also forms of technology and no doubt hinder your students ability to do computations in their head". Ultimately, however, I do not believe that the use of technology is justified solely by the unsatisfying explanation that computer technology is a fact of life and so using it is inevitable. I will argue that computational software should not be used in our courses simply because it is there, but because using it has affirmative mathematical value and because it actually requires the student to develop more mathematical sophistication than they otherwise would.

Finally, there is a whole other class of mathematics professors who have already incorporated technology, in some form or another, into their teaching. Some of them are very satisfied with what they are doing, some are not. One colleague of mine spent a good deal of time on an in-class technology effort, but now is regretting that he has to go to the trouble of teaching the technology-based course year after year. Whatever these mathematicians' current use of technology in teaching, I hope that the ideas in this book will help these people in some way to improve their approach. Perhaps this book can help them to make the process simpler and easier for themselves and their students. Perhaps it can help them to make the content of what they do more significant. In addition, this book may equip these technology enthusiasts with the tools to convince their foot-dragging fellow instructors to join the computer age, if only in a minimal way. Or, at least it may foster some patience for unenthusiastic colleagues. I hope that the enthusiasts will adopt more of a view that the when it comes to incorporating technology in the curriculum, *anything worth doing is worth doing gradually.*

2. What this book is about

In short, I will propose ways to incorporate computational software in undergraduate mathematics courses as **simple homework** that requires no in-class demos, no interaction of the instructor with the technology and almost no knowledge or time commitment on the part of the professor. In addition to explaining the concept of simple homework I provide numerous examples in the text and in Appendix C, and provide links to simple homework assignments using a variety of programs. The simple approach that I present does not make any demands whatsoever on the classroom style of professors who use it. Anyone can employ the homework and continue teaching with their present style.

I am neither a mathematics education expert, nor a computer expert. But the very lack of these credentials is a credential given the Book's goals, as experts in education or computers are much less

likely than someone like me to fully appreciate the need for a simple approach. What I consider myself to be is what most math professors are: a research mathematician with a genuine interest in teaching well. I recognize computers as a useful tool and would even go as far as to say that their use is an intrinsic part of the subject of Mathematics itself, but I honestly don't like to deal with them directly more than necessary. I also have a healthy respect for research in mathematics education, but my interest in educational theory, like my interest in computers, is as an end user. This background, along with some very fortunate circumstances and associations, are what enabled me to develop a simple approach to using computational software. I hope that this background will also allow me to better relate this approach to a large cross-section of mathematics instructors.

The ideas presented in this book come both from observing others and from my own efforts to implement a simple approach. From my experiences at various institutions, I have seen that the introduction of technology can go very smoothly if done well, and that otherwise, it can be a very frustrating experience. I hope to outline factors that make the process succeed and that give students mathematically significant training in use of the software.

Here are some major themes of the book.

- **Computational technology is very important to the subject of mathematics itself.**
- **Using computational technology appropriately requires mathematical sophistication.**
- **We have a responsibility to society and to ourselves to teach people to use computational software appropriately.**
- **Simple homework using computational software can teach students valuable lessons about using technology intelligently.**
- **Most educational uses of technology are too burdensome for faculty.**
- **Simple homework using computational software can be easy for the instructor to implement.**
- **Simple computer homework can be used without major changes in the instructor's existing teaching style.**
- **Many uses of technology are too burdensome for students.**
- **Simple homework using computational software can be done in such a way that is not burdensome to students.**
- **I am not promoting the use of computers for pedagogical reasons.**

The skeptical reader will recognize that I am proposing that the math community can "have it's cake and eat it too." My experience tells me that we can. We can introduce mathematically substantial computer technology into our teaching without a lot of fuss.

CHAPTER 2

Why I hate computers

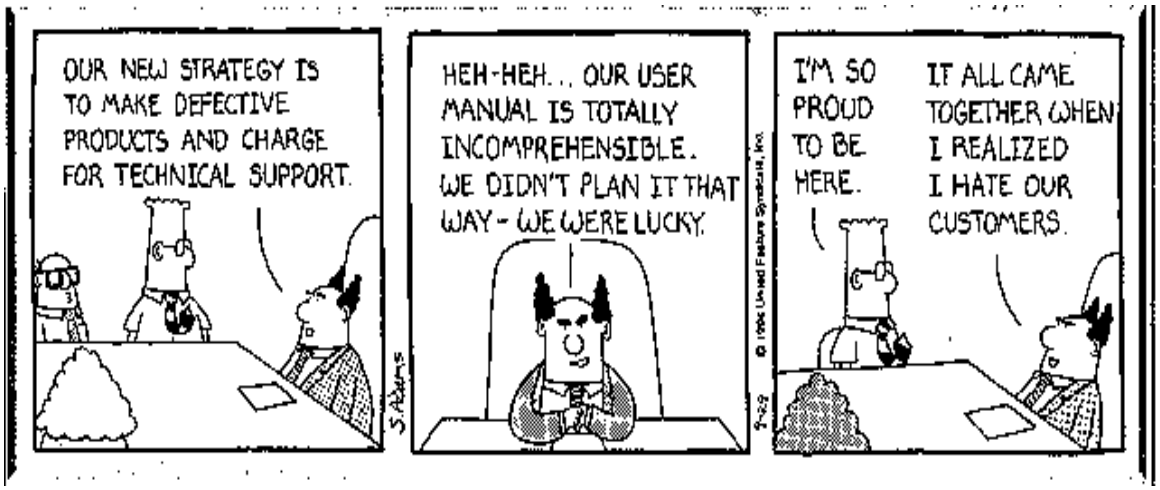
"64K is enough memory for anyone." – Bill Gates, 1978

1. The making of a technophobe

I grew up at the dawn of the computer age and in my educational career took many classes involving computers. According to conventional wisdom this should have made me "computer literate" and "comfortable" with computers. In fact, quite the opposite is the case. Instead I classify myself as "technically challenged" and maybe even a "technophobe." What happened? I report some of my personal history, because others in mathematics have had similar experiences, and the technical enthusiasts should have some understanding of why we would want a "Simple Approach." I mostly hope to save some students from similar experiences.

In my junior year of high school (1979), I took a computer programming course at a local college. I was motivated by the widely accepted idea that computers "are the future." I was told that, "Knowing computers will be very important in any career." So, I signed up for a class in which I learned to write programs in BASIC on a TRS 80. This enterprise was simple and self-contained and I enjoyed the course.

When I went to the University of Kentucky as an undergraduate majoring in Electrical Engineering, the first semester programming course used PASCAL on a mainframe computer, with commands input by punch cards. In addition to this, I was in an experimental section of Calculus in which we were



using the first programmable TI calculators. I had to learn the TI programming language entirely from the user's manual. During my sophomore year I had to learn FORTRAN and by this time we had advanced to a system where commands were input to the mainframe via a terminal. I had to learn to use a line editor (`vi`), as well as some operating system commands. Usually, no technical help was given by the instructors of the courses. By this time I was already tired of learning a new language for every course, and was sure I did not want to be a Computer Engineer.

In my Junior year I changed majors to Mathematics. Having an applied bent, I took the Numerical Analysis course, that mercifully used FORTRAN, a language that I already knew. However, the operating system and editor had changed, so I had to learn those. Also, still being interested in engineering I took a Circuits course in which we had to learn and use an early version of MATLAB, which was only on the computers in a special lab. I also had to learn to use an HP calculator. I got a "B" in the Numerical Analysis course because I could never get my programs to compile. By this time I was aware that even though I could handle quite easily the logic required to program, I was really bad at technical aspects of computing. For instance, I make a lot of typing mistakes and have a hard time finding them; I don't remember commands well; and I often forget to do things like saving my work. At that point I decided not to be a Numerical Analyst, even though I really liked the mathematics involved.

After graduating with a B.S. in Mathematics, I entered a two-year Master's degree program in Engineering Mechanics. This program was heavily research oriented and so I spent most of my time writing a thesis. The word processing in that department was all done using Word Perfect, so I had to learn that and also the basics of MS DOS. PCs at that time were of course way too slow for calculations, so I did all the numerical work on the mainframe in the Computer Science department. The data had to be made into graphics, and I learned some SAS for that purpose. There was no way at the time to export graphics to the Word Perfect environment, so I had to (literally) cut and paste the graphs into my thesis. To top it off, time on the mainframe was expensive and was charged to my advisor's account. So experimental data had to be handled on the PC's, for which purpose I learned to use Lotus. Somehow I suffered through the production of a 200 page thesis using this menagerie of software, but I was really sick of computers before it was over. I had spent about as much time dealing with computer problems as I had doing actual research.

Upon finishing the M.S. degree, I went to China for two years. During this period I had time to rethink my relationship with technology. When I returned to the U.S. and began pursuing a Ph.D. in mathematics, I had committed myself to be very selective about my computer use. I decided to have two criteria that any software had to pass before I would invest any time in it. These two criteria were that the software must be both:

(1) **Productive**, and
(2) **Stable**.

By “Productive” I mean that the software allows me to do more research and publish more papers than I could without using it. By “Stable” I mean not only that the software is reliable, but also that it is not changing rapidly, and is widely used and supported.

As a result of my criteria for technology, during the first two years of graduate school, *I did not use computers at all*. Since I was only doing classwork, teaching, and preparing for comprehensive exams, I did not find that any computer use could be truthfully deemed “Productive.” During the next two years I began doing research and writing papers. By that time TeX and LaTeX seemed to have become almost universally used in the Mathematics community and people assured me that the software would not change much, so I began to use LaTeX. I also began to use email to correspond with researchers and friends. During the last two years of graduate school I added MATHEMATICA to my repertoire for teaching purposes (more about that later) and started to use a web browser for the purpose of looking at job advertisements on E-math.

I am extremely satisfied about how this strategy served me during graduate school. I was able to submit five papers before graduation, and had a couple more near completion. Believe it or not, during this period my two children were born, and I never worked during the evening and almost never on weekends. I attribute a lot of my productivity to staying away from computers as much as possible!

After graduation, I spent two years at Northwestern University at which time I learned to use MAPLE and a little bit of C for research purposes. I don’t really know anything about C, other than how to take someone else’s working program and modify it for my own purposes.

When I took a position at Ohio University, I switched to using a PC running the Windows operating systems, because that was the only platform that was best supported in our department. I still only used LaTeX, email, and a web browser, and write simple C programs. There seemed to be lots of other things on my computer, but I purposely avoiding learning anything them. Luckily, the switch to a PC from Unix workstations did not require much effort, although this is surely due to the fact that I choose to do only simple things.

Until accepting a tenure-track position, I purposely avoided learning html and having a webpage. Being in a somewhat more stable position, I decided I should finally have a webpage for the sake of public relations. The department’s administrative assistant was offering to set up webpages for faculty and I took advantage of his offer. When my webpage needs to be updated I generally ask someone to do it for me, unless it is just a matter of changing a word or two in the text.

To this day I am still not good with computers. At least twice during the writing of this book I have neglected to “save” often enough and lost a couple of hours worth of editing. I dislike dealing with systems issues. I plan to continue limiting myself to stable, productive computer use. A good rule I have for anyone wishing to avoid technical trouble is to begin using a technology only after it has become completely standard. One should stay away from the cutting edge, unless one doesn’t mind getting cut.

2. Problems with computers

My own bad experiences aside, Here are some of the reasons i have found that should lead one to be wary about computers.

- **They are distracting.** I have personally witnessed lots of young mathematicians waste lots of time that they could not afford on all kinds of amusements on computers. People will argue that recreation is good and good recreation is by nature distracting. Granted. I am just saying that one should consider the matter carefully before deciding how to spend one's limited recreation time (or especially to spend work time on recreation). To all the graduate students out there let me say that I have now served on a hiring committee and I can state with assurance that when it is time for you to get a tenure-track job, the quality and quantity of your research will be much more important than having a picture of your cat or dog posted on your homepage.
- **It is hard to get them to work right.** This of course means, to get yourself to use them correctly, but the results are the same. Computers as we know them today are completely inflexible. You must generally type the command exactly the right way or the computer won't do what you want it to do. The "point and click" revolution was supposed to alleviate the necessity of remembering commands, and it has to some extent, but with it, new complications have been introduced. The thing one wants to do never seems to have an icon in view. Another difficulty, which relates to the problems I had as an undergraduate, is that there is no compatibility or standardization among different manufacturers and software writers. When one starts learning a new system, one must start from scratch. Further, even with the best preparation, new technology has glitches.
- **Computer documentation is impossible to decipher.** Manuals are typically hopeless. Help utilities are improving, but still have a long way to go. I find that even the learning guides for the few things I use, like MAPLE and LaTeX, are filled with lots of stuff that don't seem relevant to the average user. For instance in LaTeX manuals there is a lot of fuss over what is or isn't a "fragile" command. I have been using LaTeX for several years now and to this day I have no idea what "fragile" means. This makes it very time consuming to figure out what to do. Unless someone has a bent for such things, and I don't, it is far better just to learn the minimum amount necessary to do your work. The best way to learn is from other people's working examples.
- **Computer people aren't like the rest of us.** This is probably the reason behind the problem with documentation, and often lies behind the problems of getting computers to work. From my experience, people who are really good with computers (and thus are in control of them) are often more interested in making the computer do cool things than they are in doing productive things; thus the proliferation of screen savers, for example. Computer people are

often attracted to doing very advanced things which are prone to cause more problems then they are worth, at least in the estimation of non-computer people. For instance, I recently loaned the use of my office and computers to a visitor who is a computer person. Not satisfied with my one Windows machine and one Linux machine, he decided to run Linux on the Windows machine as well. In the process he accidentally erased my entire hard drive and spent days recovering my files. As a non-computer person, I can scarcely imagine how frustrated I would be to waste so much time.

- **Computers as educational tools are very overhyped.** In later chapters I will discuss some of the scholarly literature about the pedagogical advantages and disadvantages of technology in undergraduate mathematics. For the moment, I mean only to point out the unrealistic and overblown characterizations of the educational value of computers, a point to which I now turn.

3. Computers and education

When it comes to the educational use of computers, a large dose of skepticism is in order. Popular (over)enthusiasm about the educational value of computers is great for the computer industry, which spends hundreds of millions of dollars each year promoting the idea that the Internet is the gateway to all learning. I recently saw an ad that said that soon you could prepare for any career on earth over the Internet. First, I don't believe it. Second, I certainly hope not, as I think there is great value in the educational process as it now occurs. Another advertisement I have seen is particularly annoying. It is the one where an elementary student sends and receives e-mail from an astronaut on the space shuttle. There are a number of good reasons to be cynical about this commercial:

(1) The number of students who could participate is such a game is so limited, it would be hilarious if so many people didn't take it seriously. At most an astronaut could answer about 10 e-mail messages an hour. Given that putting an astronaut into orbit is extremely expensive and their time in space to crammed full of other activities, I just don't see a lot of correspondence being possible.
(2) If we insist on spending this much money for students to communicate with astronauts, couldn't we let the students talk to the astronaut on the phone? There is nothing unique about computers that makes it possible to communication with people in space.
(3) What makes anyone believe that an astronaut has anything relevant to say to a fifth grader?

Perhaps more information, at less expense, can be available over the Internet than through a traditional library. But students are not learning all the information available to them now, so we have no reason to think that what they need is increased availability of information. Indeed, it seems more likely that students are in fact overloaded with information and what they lack is time and guidance to assimilate and understand the information they have.

Like others, I fear that equipping schools with Internet access only distracts from much more important issues like class sizes and teacher pay and training. Frankly, learning to use the web does not require years of training. My own mother learned to surf the web without difficulty at age 60, without the benefit of having had computers in her childhood school.

As we have moved around for academic jobs, whenever we could, we have sent our children to Montessori schools which, as a rule, do not have computers. In these schools students work with their hands and minds. For instance, they learn mathematics through handling of "manipulatives", they learn reading by reading good books and they learn about science by hands-on experiences and by reading ordinary old-fashioned books. They interact constantly with teachers, teacher's aides, and classmates (live human beings), developing language and social skills. There is in fact growing evidence that dealing with the physical world is very important for children's intellectual development. It is theorized that humans are hardwired to develop logical thinking through experiences with the physical world. A computer screen cannot provide this kind of learning environment. However, I feel certain that any person who has developed basic intellectual skills can master any uses of a computer that they will ever need. Computers in classrooms definitely do not promote good communication and social skills, which I would argue are more important than computer skills, both for career advancement and for a happy life.

The overhype concerning the educational value of computers is not limited to primary and secondary schools - at Colleges and Universities around the world it is very attractive to administrators to put a lot of emphasis on computers, albeit with only a fuzzy notion of how the computers are to be used. One of our responsibilities as mathematicians is to educate these administrators about uses other than simply "being connected to the web." At my University, computers were recently installed in all freshman dorms. This event was advertised with a lot of fanfare with appropriate mentions of phrases like "information superhighway" and "new millennium". However, these computers do not have access to any computational software, although they do have a network-interactive version of one of the most popular video games. The math department has lobbied for over two years to get this situation changed, but computational software is still not available in the dorms. There simply wasn't any money budgeted for mathematics software. Eventually money from an internal competitive grant bought a campus-wide site license for **MATLAB** which was supposed to put the software into all labs and dorms. So far, the administration has only put it into labs. The results of a study on the student's use of computers may have provided some impetus for getting the computational software at all. The study, which was not widely publicized, found that the largest use of the dorm computers was downloading popular music. The second biggest use was downloading pornography. These results are disappointing, but not too surprising. The information superhighway has more than its share of questionable roadside joints, that students are likely frequent given unlimited, free access. The only real surprise is that more universities have not realized the fallacy of mere access as a significant educational advantage.

CHAPTER 3

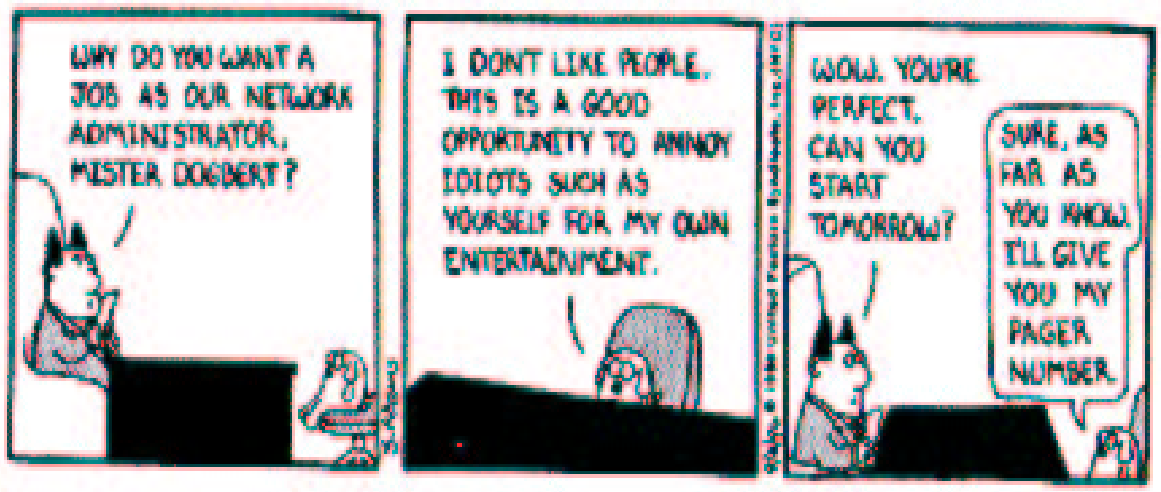
How I ended up teaching with computers anyway

"One of the main challenges in life is to grow wise, but not cynical." – James Wells, my freshman calculus professor at the University of Kentucky

1. MATHEMATICA projects at Georgia Tech

At about the time that my wife, Mary Beth, and I started as graduate students at the Georgia Institute of Technology (1991), a group of dedicated professors obtained a rather large grant for the purpose of exploring ways to integrate the program MATHEMATICA into the calculus curriculum. With the grant money they equipped a laboratory of about 20 Apple computers and began writing MATHEMATICA-based projects for use in Calculus classes. These projects contained more difficult applications of the calculus than are normally considered, they involved hard calculations that required a computer, they involved group work and they had a large writing component. The projects had many similarities with design projects used in upper level engineering courses. The leaders of the initiative soon began to use these projects in classes and encouraged other faculty and graduate TA's to do the same.

Mary Beth and I were reluctant to use the projects in our own courses. Having been the victims of several educational trends ourselves over the years, we questioned the value of adding computers to the calculus in the first place. Further, it seemed that those involved with the projects were dedicating a really large amount of time to them and as new parents we viewed our own time as scarce and valuable.



Accordingly, we watched with removed cynicism. At that time if you had told me that I would eventually write a book about using computers in teaching I would have said you were crazy. Then a couple of things happened to change my mind.

2. Mary Beth tries the projects

The first thing to happen to knock us out of our pre-technological bliss was that after completing a Master's degree, Mary Beth decided to not pursue a Ph.D. in mathematics, but rather she wanted to eventually go to Law School. She began looking for a job for the time until I finished my graduate work and the Georgia Tech mathematics department offered her an instructorship, which she gladly accepted. In the hiring process she was particularly encouraged to use MATHEMATICA, so with that encouragement she decided to try using the computer projects.

During her first quarter using the projects she did the following: learned some MATHEMATICA herself, took the students to the lab regularly, tried to answer a lot of technical questions, spent lots of time explaining and re-explaining the complicated projects and even more time trying to grade the final products. Her conclusion at the end of the quarter was that it just wasn't worth it. Despite her huge investment in MATHEMATICA, she concluded that the students:

(1) Did not really gain much experience using MATHEMATICA.
(2) Did not learn calculus any better.
(3) Developed a hatred for MATHEMATICA and for her.

In fact, she suspects that the students learned even less calculus because the time-consuming nature of the projects led them to spend less time on regular study and homework, and because the ill-will generated by the projects deteriorated the learning environment. This was especially exasperated by her presence in the lab while students were struggling through the projects. By generating student frustrations and making herself a handy target for them, she felt that she was sacrificing the student-teacher relationship and she was not willing to let this happen.

Her first quarter using the projects was her last. However, being a very conscientious person, still feeling an obligation to the department to use MATHEMATICA, and *having seen that the program itself was indeed quite remarkable*, she decided to invent her own way of employing the software with her classes.

3. Mary Beth charts her own course - Simple and Simpler

As she planned for the next quarter, Mary Beth identified three main problems that she hoped to correct.

(1) The projects had been too long, too complicated and not clear enough.
(2) Documentation on using MATHEMATICA was bad.
(3) It is counterproductive to be in the lab while students are working.

To alleviate the first problem, she decided to write her own greatly simplified projects. These will be discussed below.

The inadequacy of documentation on MATHEMATICA pervaded the user's manual, the help command and the instructions in the projects. Nowhere in any of this could she find a concise or readable introduction to the basics of using MATHEMATICA. As a result, students had a lot of difficulty figuring out how to use MATHEMATICA to do what they needed. Further, they would forget all the syntax and commands used in the previous assignments and their only remedy was to search through the previous projects (of course, the smart students wrote commands down). To deal with this she compiled three pages of simple, clear instructions on the basics of MATHEMATICA. This document, the "MATHEMATICA Info" sheet is shown on pages 14-16.

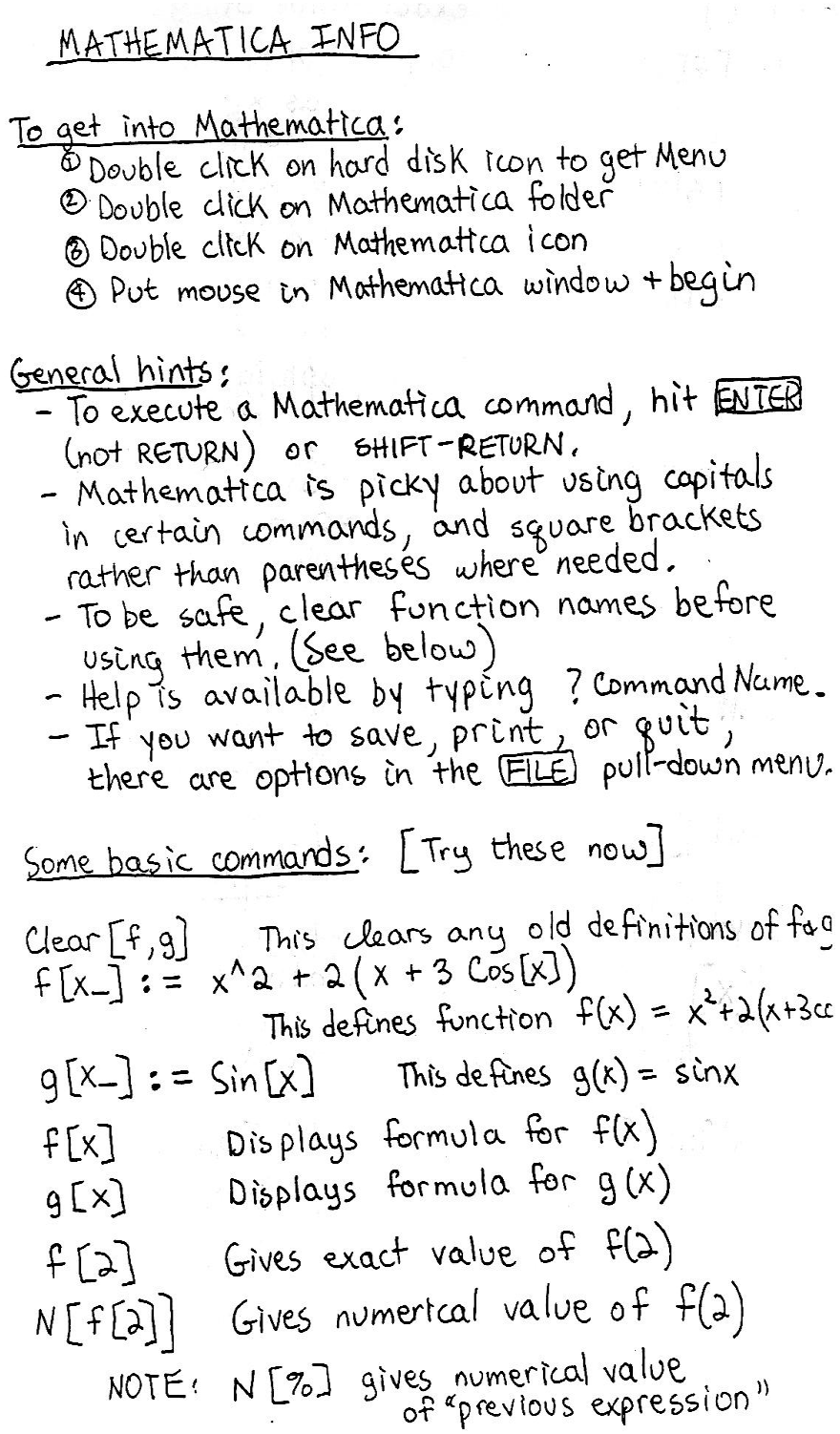
She hoped to alleviate the last problem by arranging everything so that assignments could be completed as homework, without her assistance in the lab. It was partly with this in mind that she made the instructions of the projects very clear, so that with the aid of "MATHEMATICA Info", they could reasonably be expected to do the work without supervision. Then, she simply did not schedule any lab times with the students except for one hour during the first week. In this hour she handed out "MATHEMATICA Info" and asked them to read the first two sections and then try out the commands contained in them. After that, with the help of "MATHEMATICA Info", and assisted by the relative ease and clarity of her projects, the students encountered very few technical difficulties that they could not work out among themselves and her presence in the lab was not missed.

Mary Beth's first project of her own is on pages 17 and 18. It emulated the existing projects by exploring an application for which MATHEMATICA could be used to obtain the final answer. This project still took too much time. Basing the project on an application not only took a lot of time for Mary Beth to write, it caused problems for many students, problems which they brought to her. In the end the students still spent more time setting up the problem than they did using MATHEMATICA.

On the other hand, several things about this assignment did work well. The explicit instructions allowed the students to use the program without technical help. Note that the instructions even include how to edit the file before printing. Mary Beth had learned from experience to not take anything for granted. All in all the actual use of MATHEMATICA went much better than before.

Mary Beth's second attempt at MATHEMATICA homework is on page 20. Notice that she did not typeset the assignment and the assignment is not about an application. It is simply about using MATHEMATICA to do some basic calculations. In retrospect, Mary Beth thinks that this one went a little too a far in its minimalism, but not by much. The students had no trouble with this assignment and from that point on the "project" format was abandoned in favor of the "simple homework" format. This was the major turning point in the evolution of the ideas of this book. Her subsequent assignments were of the style of the second one, but required a little more analysis on the part of the students.

The results of Mary Beth's ventures in writing her own MATHEMATICA assignments were overwhelmingly positive. Basically, the problems mentioned above seemed to have been solved. Students



MATHEMATICA INFO

To get into Mathematica:
① Double click on hard disk icon to get Menu
② Double click on Mathematica folder
③ Double click on Mathematica icon
④ Put mouse in Mathematica window + begin

General hints:
- To execute a Mathematica command, hit ENTER (not RETURN) or SHIFT-RETURN.
- Mathematica is picky about using capitals in certain commands, and square brackets rather than parentheses where needed.
- To be safe, clear function names before using them. (See below)
- Help is available by typing ? Command Name.
- If you want to save, print, or quit, there are options in the FILE pull-down menu.

Some basic commands: [Try these now]

Clear[f,g] This clears any old definitions of f & g
f[x_] := x^2 + 2(x + 3 Cos[x])
This defines function $f(x) = x^2+2(x+3\cos$
g[x_] := Sin[x] This defines $g(x) = \sin x$
f[x] Displays formula for f(x)
g[x] Displays formula for g(x)
f[2] Gives exact value of f(2)
N[f[2]] Gives numerical value of f(2)
NOTE: N[%] gives numerical value of "previous expression"

FIGURE 1. Mary Beth's MATHEMATICA Info sheet, page 1. This was handed out at the first of the course as a "simple, understandable user's reference."

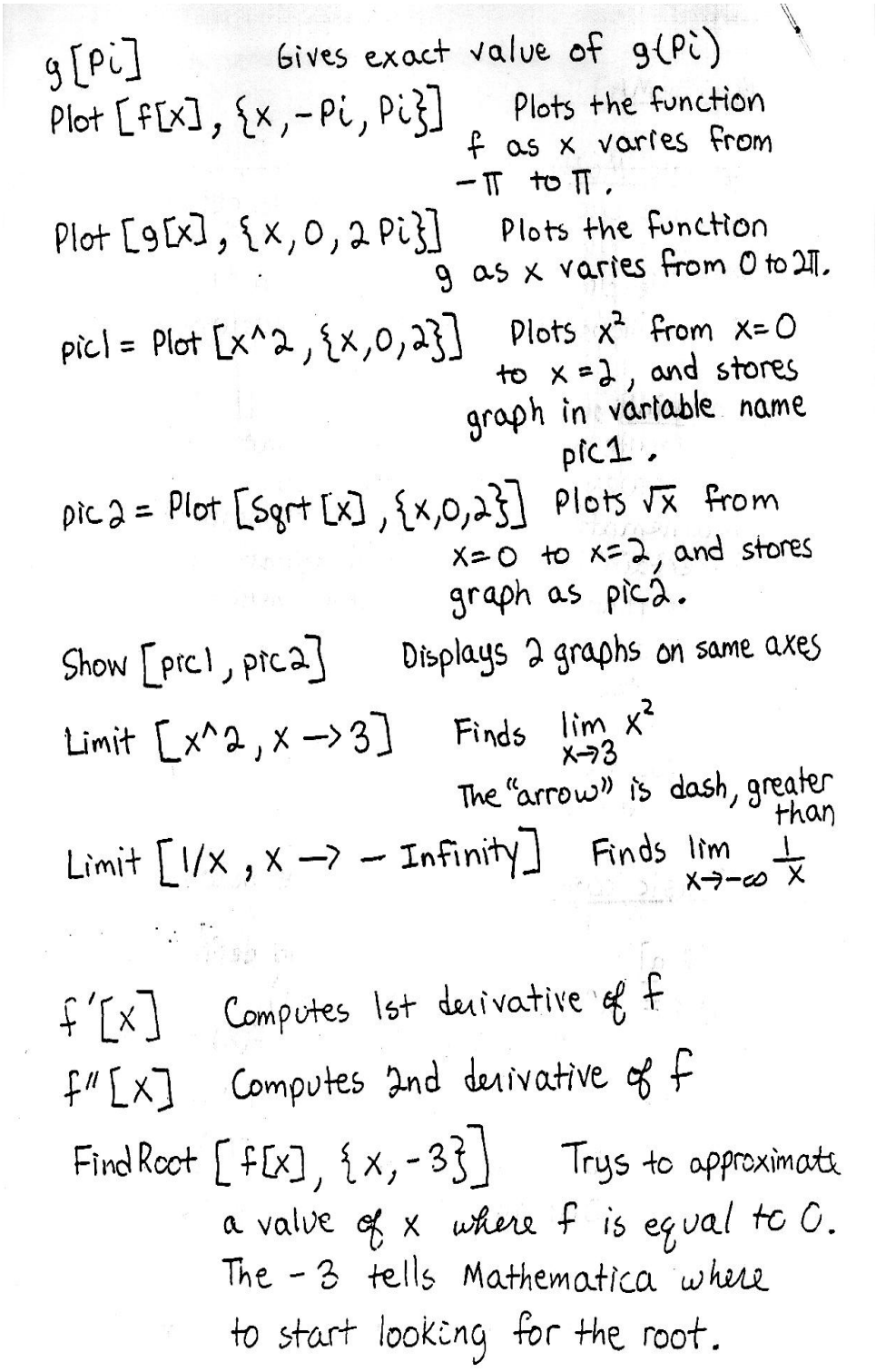


FIGURE 2. Mary Beth's MATHEMATICA Info sheet, page 2.

The command

Integrate [expression, var] gives an indefinite integral. (without a constant)

For example

Integrate [2x+1, x]

yields $x^2 + x$.

The command

Sum [expression, {var, varmin, varmax}]

is used to evaluate a summation.

For example

To evaluate $\sum_{i=1}^{100} (-1)^i \cdot \frac{1}{i^2}$, type

Sum [(-1)^ i / i^ 2, {i, 1, 100}]. This yields a messy fraction. To see a decimal approximation, type

N [%] which yields

-.822418

FIGURE 3. Mary Beth's MATHEMATICA Info sheet, page 3.

Mathematica Project I You may turn in individually, or work with one other person. (If you work in a pair, be sure both names are on the printout you turn in.)

General Hints:

– To execute a Mathematica command, hit ENTER or SHIFT-RETURN. (Just RETURN will not work.)

– Mathematica is very picky about using capitals in certain commands, and about using the correct kind of parentheses, brackets, etc.

-- Help is available by typing ? followed by a command name.

-- To save, print, or quit, choose the correct option from the FILE pull-down menu.

Mathematica is able to solve linear systems. The example below illustrates how to use Mathematica to solve the system:

$$\begin{aligned} w + 3x - y + 6z &= 1 \\ 2w + 4x - z &= -1 \\ -2x + 3y - z &= 2 \\ -w + 2x - 5y + z &= 5 \end{aligned}$$

```
In[1]:=
  m={{1,3,-1,6},{2,4,0,-1},{0,-2,3,-1},{-1,2,-5,1}}
Out[1]=
  {{1, 3, -1, 6}, {2, 4, 0, -1}, {0, -2, 3, -1},
    {-1, 2, -5, 1}}
In[2]:=
  MatrixForm[m]
Out[2]//MatrixForm=
  1    3    -1   6
  2    4    0    -1
  0    -2   3    -1
  -1   2    -5   1
In[3]:=
  b={1,-1,2,5}
Out[3]=
  {1, -1, 2, 5}
In[4]:=
  LinearSolve[m,b]
Out[4]=
  {-(209/9), 305/27, 73/9, -(7/27)}
```

FIGURE 4. Mary Beth's first attempt at a "simple project", page 1.

Your Assignment: Use the circuit diagram below to set up a linear system. Then use Mathematica to solve for the various currents. I want a printout showing your work, as in the example given. Before printing, please remove all incorrect or irrelevant information from your file. To do this:

1. Use the mouse to highlight the side bar of the cell to delete.
2. Choose cut from the edit menu.

Do this for each cell you want to remove. When you have your file like you want it, choose Print from the File menu.

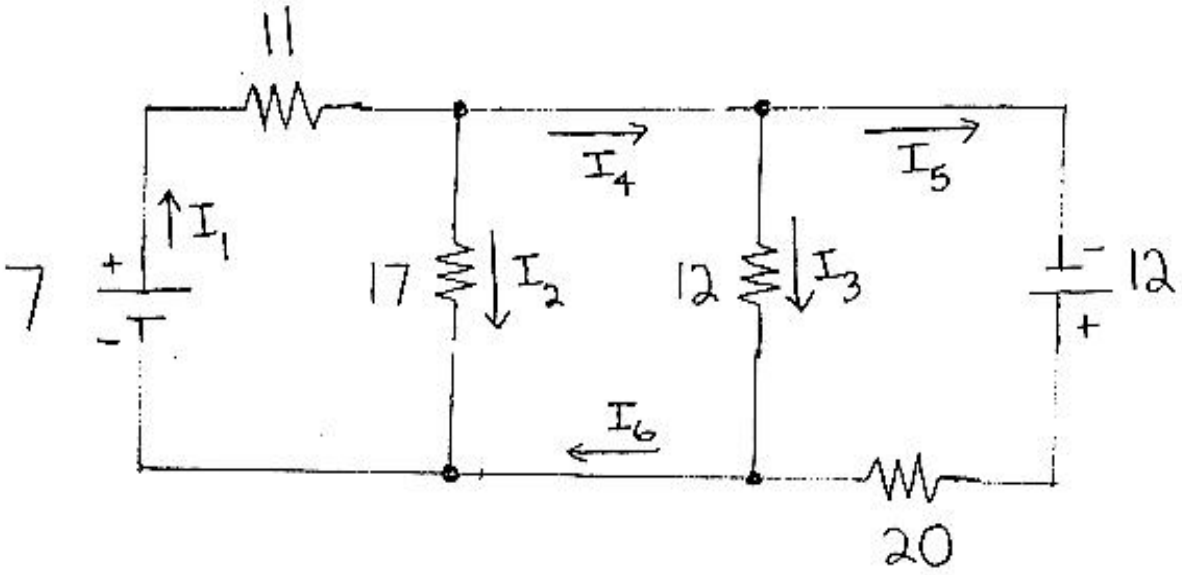


FIGURE 5. Page 2 of Mary Beth's first attempt at a "simple project."

were positive about the assignments, did them well and seemed to learn from them. Despite the time spent writing her own assignments, Mary Beth actually spent less time on the computer homework than in the previous quarter. The time spent with the new simple homework was also less frustrating. Most of the remaining criticisms about the homework were aimed at the documentation and difficulties of

the program itself. Many of the students who had been in sections using the more complicated projects expressed their appreciation for the simple homework approach. Not only were they less frustrated, but they felt they had learned more.

4. Teaching with technology becomes a professional obligation

At about this time, I decided to begin giving MATHEMATICA homework. One factor in my decision was Mary Beth's experience that it could be done without costing all my precious time. I had followed the progress of her experiments closely and we had spent many hours over dinner discussing them. The other reason I started was that I became convinced that it was the right thing to do. There were several factors that led me to this conclusion and I will discuss them in more depth in the next chapter, but one particular event was "the straw that broke the camel's back."

We were in the office one day when one of our fellow TAs entered the office and exclaimed "Look at this neat graph that MATHEMATICA made." The graph was labeled x vs. $\sin x^3$ and was obviously a case where the program had under-sampled and the graph was garbage. A copy of the actual graph is shown in Figure 7. I assumed that this TA was excited by the graph because it showed the limitations of the program. But in fact, this TA thought the graph was accurate and was excited because the program had produced something that was too complicated to obtain by hand. This TA had even made 35 copies of the graph as hand-outs for the class to show them what the graph of $\sin x^3$ should look like.[1]

Appalled, I attempted to explain to my fellow graduate student that the graph was inaccurate and why, but to no avail. "The computer did it, it must be right", was the response. My fellow TA was in fact incensed at my arrogance in believing that I was smarter than MATHEMATICA.

This event cemented my conviction of the following facts:

(1) Technology is out there and people will use it whether they understand anything about it or not.
(2) Computational software is dangerous in the hands of novices.
(3) It is the responsibility of the Mathematics community to teach intelligent use of such software.

Misuse of computational software in classwork endangers the reputations of our students and ourselves. More importantly, if such misuse occurs in industry, it is not out of the question that public safety could be jeopardized.

5. I begin using computer homework

Finally motivated to teach with technology, I began writing my own MATHEMATICA homework. I would have taken the really easy route and just used Mary Beth's assignments, but I was not teaching a course that she had done before. The ideas behind my projects were basically the same as hers. My assignments contained very clear instructions and did basic computations. I gave out copies of Mary

[1]Not surprisingly, this graduate student did not survive long at Georgia Tech.

Mathematica Project II Names (1 or 2)

(A Gift for You . . .) _____

① Input the matrices

$$m = \begin{pmatrix} 1 & 3 & -1 & 6 \\ 2 & 4 & 0 & -1 \\ 0 & -2 & 3 & 1 \\ -1 & 2 & -5 & 1 \end{pmatrix} \quad \text{and} \quad n = \begin{pmatrix} -1 & 3 & 3 \\ 2 & -1 & 6 \\ 1 & 4 & -1 \\ 2 & -1 & 2 \end{pmatrix}$$

by typing

m = {{1,3,-1,6},{2,4,0,-1}, etc... }

n = {{-1,3,3},{ ... etc.}

Display using MatrixForm[m] and MatrixForm[n]

② Find the determinant of m using

Det[m]

③ Find the inverse of m using

Inverse[m]

④ Multiply m by n by typing

m.n

⑤ Think about how easy this was, and how nice computers are.

⑥ Get a printout to prove you did it.

FIGURE 6. Mary Beth's second MATHEMATICA "project." The simple homework concept is born.

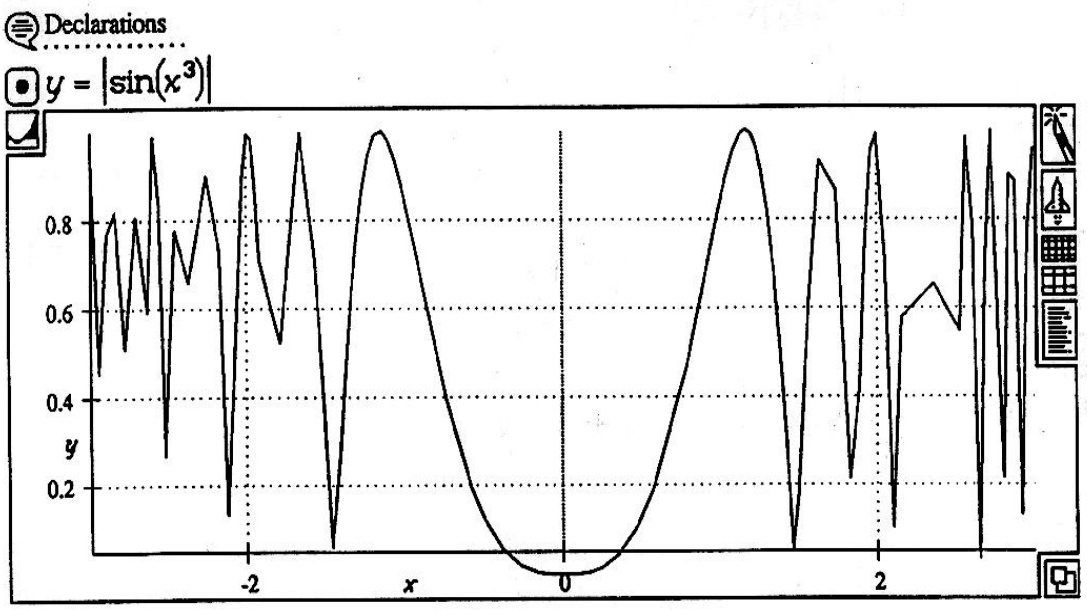


FIGURE 7. Graph produced by MATHEMATICA and distributed by a TA to a calculus section. The TA believed the graph to be a correct representation of $y = \sin x^3$.

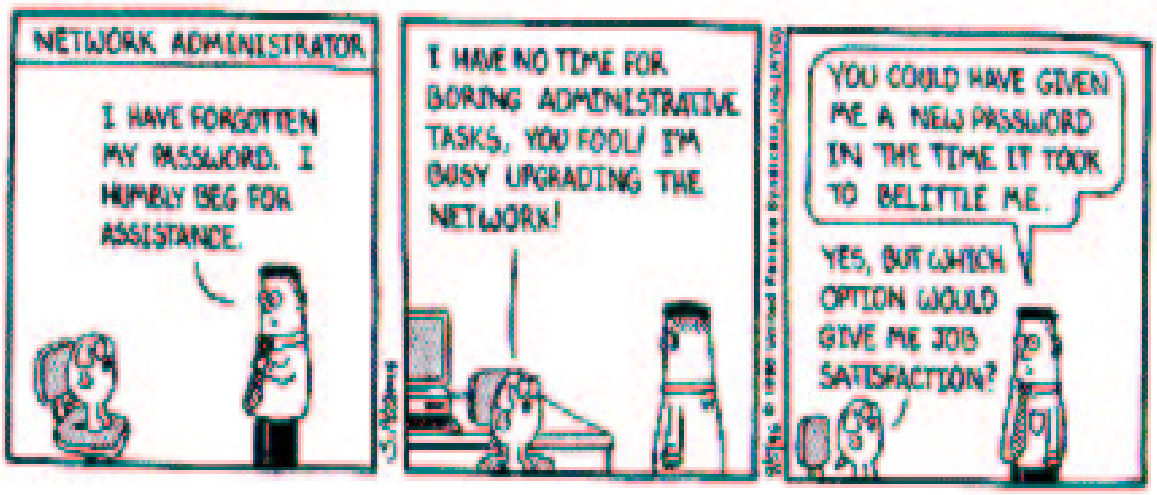
Beth's "MATHEMATICA info sheet" at the beginning of each quarter, never went to the lab with the students, and assigned about four assignments per quarter, often as extra credit. I encouraged students to work together on the homework, but usually required each student to write their own summary. For about two years I used this approach at Georgia Tech in several sections of Calculus, Differential Equations and Linear Algebra. During this time I had very few problems with technical difficulties and the students had very few complaints about doing the homework.

During this time the incident involving the graph of $\sin x^3$ in Figure 7 not only served to motivate me to begin using technology, it was a guiding point for "what to teach." As I wrote my first assignments I sought to include examples where the software failed for one reason or another. These examples then led to discussions with the students about the algorithms the computers use. During this time period I developed the view that computer homework could contain good mathematical content while at the same time being simple.

Some of the homework I assigned during that time are on pages 48, 49, 50, and 51.

6. At Northwestern

After graduation from Georgia Tech, I went to Northwestern University as a visiting assistant professor in the fall of 1995. Up until that time, technology had been used in teaching by only a handful of math professors at Northwestern. During that year I took part in a year-long discussion about



introducing technology in the calculus sequence. The discussion was fueled partly by pressure from outside the department, particularly from the engineering school, which was threatening to teach its own calculus courses. In the end, the University and the Math Department agreed on a course of action. The University purchased a site license for MAPLE, that included an agreement that students in classes using MAPLE could download the program to their personal computers (most students at Northwestern own computers) over the web. Further, MAPLE was installed on the computers in most of the labs on campus. Beginning in the fall of 1996, use of the software was *mandated by the College Dean* for every calculus class. With the mandated use and campus-wide access to MAPLE, professors in other courses could take for granted that students were familiar with the software. During the summer, Professor Leonard Evans, the undergraduate chair of the department, wrote excellent worksheets[2] for almost all of the calculus courses. These assignments could be downloaded off the web. At the beginning of each course students were handed instructions on how to download MAPLE and the homework. Attendants at campus computer labs were given basic instructions on MAPLE and on downloading homework. The whole thing went over without a glitch; neither professors nor students complained. I taught two sections of calculus during the first quarter that MAPLE was introduced and I received precisely zero questions about how to obtain or use the software and almost no complaints. All I had to do was hand out the instructions, tell students when assignments were due, collect the homework, grade it (which was easy since a large majority was perfectly done) and discuss with the students the mathematical content of the assignments.

One point is worth special note. I did not know MAPLE at the time I was giving MAPLE assignments at Northwestern. I did learn some from grading the homework, but did not sit down at a computer and use MAPLE myself. This I think illustrates just how easy computer homework can be for instructors, if implemented in the right way.

[2]A "worksheet" is the basic MAPLE document type. It is opened using MAPLE, much as a "Word" document is opened using MS Word.

My experience at Northwestern was so pain free that I was forced to address the question of which format is preferable, to assign simple homework as Mary Beth and I had done, or to give prewritten worksheets as homework. I ultimately came to the conclusion that there are some definite advantages of the simple homework approach. In Chapter 8, Section 5, I will discuss the pros and cons of each method.

CHAPTER 4

Implementing technology in teaching: Lessons about simplicity

"Keep it simple stupid" – Anonymous

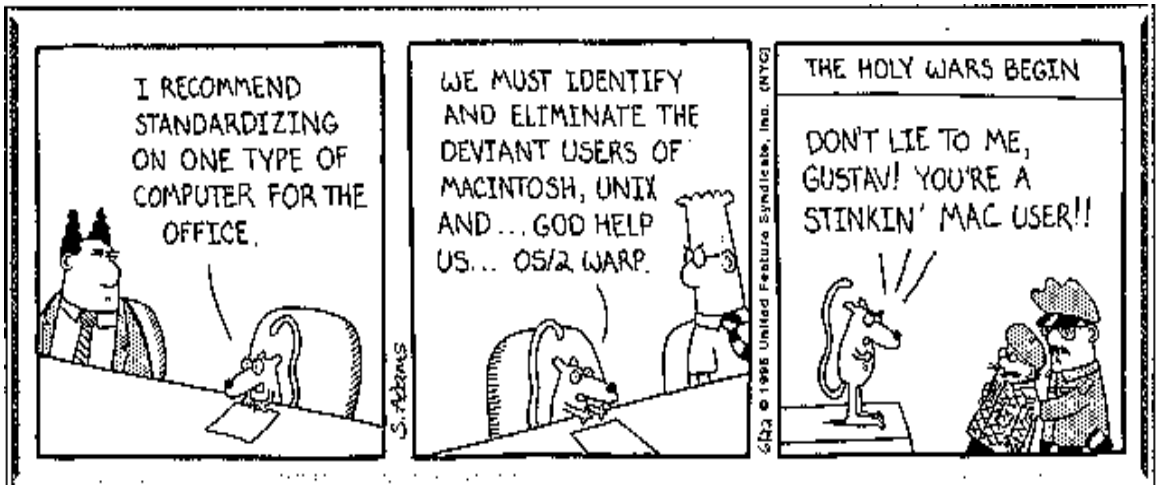
1. Widespread availability and widespread use are crucial

For a software package to be truly simple to employ in your calculus course, it should be available on most computers on campus and it should be used in enough classes that all science, math and engineering students must encounter it multiple times. That is, the software must be widely available and widely used. The obvious advantage of widespread availability is that students can use the software when and where they choose. This eliminates problems like crowding of specific computer labs when a project is due. As work becomes more distributed, pressure on individual labs decreases. Another clear advantage of widespread availability and use is that in subsequent Math classes and in math-using courses, professors are able to assume knowledge of the software. This leads to a kind of snowball effect: widespread use encourages more widespread use. Once this begins to take place, and provided the mathematics department is teaching students to use the software intelligently, computational software can become a useful tool for mathematics throughout an institution.

A less obvious, but perhaps more important, benefit of widespread use is that a culture or collective knowledge about the software develops. When this happens, the need for technical advice plummets, because students are usually able to obtain immediate help with problems from classmates, friends and neighbors. I have definitely witnessed this phenomenon and it is one of the keys to keeping homework simple for both students and faculty. Not only does the collective knowledge decrease technical problems, it increases the likelihood that the software will be used well.

During the initial "MATHEMATICA project" period at Georgia Tech, MATHEMATICA was only available on machines in the math department lab funded by the grant. This no doubt exacerbated the students' frustrations. Not only were the assignments hard, it was inconvenient to do them, and they could not expect the hard-earned experience to be transferable to later classes. Later at Georgia Tech, all students were required to own computers with standard software, including both MATLAB and MAPLE. Now the one or the other of these computational programs is used in most calculus sections with very little trouble. Professors in other departments are also using the software more and more because they can assume a certain amount of basic knowledge.

Another advantage of making software widely available is that the Mathematics Department does not need to have or maintain lots of labs. At Northwestern, the Mathematics Department actually had



no computer labs available for calculus students at the time MAPLE was introduced and that was not a problem. Instead, students could use Maple at any lab on campus or in their dorm rooms.

2. Uniformity is good.

In regard to many issues, uniformity is bad. For instance, we obviously should not all teach in the same way and we should not all teach with computational software in the same way. But when it comes to choosing software, uniformity within an institution is a good thing. That is, it is much better for students to use the same software package from course to course than it is to use different software in different courses. First, using the same software from course to course allows students the time to develop proficiency with the software. Learning new software always requires an initial investment of time and energy, and these resources are often taken away from actual study. Second, when students can rely on the software to be the same from course to course they are much more motivated to learn how to use it and much less likely to resent the investment of time.

Just as widespread availability and use decrease demands on faculty, having a single software package is better for faculty as well. Uniformity facilitates the building of both individual and collective knowledge of the software, thus reducing the need for technical help from instructors. It also allows mathematics professors and our colleagues in other departments to take knowledge of the software for granted and to move on to more substantial matters.

Ideally, a mathematics department, in consultation with all interested parties on campus, should decide on a software package and commit to use it throughout the undergraduate curriculum. This approach maximizes both the simplicity and the benefits of the initiative. Of course in many departments this may not be politically feasible. This decision can be made much easier if the proposed use of the software is along the line of what I am suggesting in this book, i.e., a simple approach as a minimum standard. Otherwise, many professors will justifiably object if they perceive a proposal to be a threat to their time or their way of teaching. Other departments have faced this decision and put the matter

to a vote (see [**Can95**] and [**Lew95**]). From documented cases, when departments have followed majority rule and required a certain technology on a large scale, the results have been good. Others have observed the wisdom of uniformity and widespread use. Krantz says,

> We have had catastrophic experience, at my own university, trying to introduce MATHEMATICA or MAPLE or some other software into isolated courses. Students are not stupid. They catch on right away to the fact that this software is specific to the particular course, and as soon as the semester is over they are unlikely to see it again. They resent having to learn a whole new language .... If we are going to introduce serious software into the lower-division curriculum then we should do it globally instead of locally. ...It makes sense to tell students from day one that their entire lower-division mathematics curriculum will depend on MATHEMATICA ...and that they will need to master it right away. Having understood this dictum, they will comply ...and the software will become part of their *lingua franca*. They will (we hope) carry it (or the analytical skills attendant to it) on to the rest of their education, and their lives.

Krantz adds that this is tough to do. I hope to convince the reader that introducing software globally is not only the right way to do it, but that it is not as tough as it may sound.

One of the keys to the easy success of the introduction of MAPLE at Northwestern was that the software was universally used (as well as widely available). To a large extent, the fact that the Dean mandated use of a software package saved the Mathematics Department a lot of time it would have spent dealing with technical problems from several different software packages. Further, it probably saved the professors a lot of energy that would have been wasted on internal arguing.

At Ohio University, the engineering department all individually chose to use MatLab as their main computational tool in classes. When the math department decided to also chose MatLab, we did so with a lot of encouragement from the college of engineering.

3. Use of the software can and should be simple

Especially in an introductory course, an instructor should not be ambitious in using computational software. It is assumed that a student's first contact with the software will be in the context of a math course. It is an undeniable fact that these courses are difficult in themselves. Further, these courses are major career events for students interested in the Sciences and Engineering. To overburden students in these courses with additional difficulties is unhelpful and unwise, and I would go as far as to say that it borders on irresponsibility.

A key to making the assignments simple is to avoid the project format and instead, to write assignments so that the students can go directly to a computer, spend half an hour dealing directly with the software and then half an hour thinking about what was happening and writing that up. To achieve this

it should be very clear to the student what they are to do and how they are to do it. I do not want to devalue challenging assignments in which the students must really think for themselves. However, the most effective way to introduce the software is not within the context of difficult problems or projects, but within the context of simple computations.

4. Technical instructions should be very explicit, clear and concise.

Very explicit technical instructions are necessary because of the fact that computer documentation is simply not useful to the beginner. It is not helpful to give a lower level student an exercise and say "Use MAPLE to solve this." A typical student who encounters such an assignment, if she does not have a knowledgeable friend, will probably pick up the learning guide to MAPLE [**HHR98**] and spend several hours wading through irrelevant material before reaching the commands needed. Next she will go to a computer lab where the software is available and spend a few hours overcoming system and syntax problems. In my view this process is a terrible waste of the student's time.

When I suggest that the students be provided with very clear technical instructions I mean that you should tell them everything they will need to do to complete the assignment. This should include how to get to the software on at least one readily available platform, the specific commands to be used, syntax needed to use the commands, how to get help, and even how to print and exit. By giving students simple and clear assignments, we allow them to bypass technical problems and to go directly to the tasks of gaining experience with the software and understanding the mathematical ideas.

As mathematicians, most of us have a natural aversion to anything we feel is "spoon feeding" the students. We want students to work things out themselves. This inclination is quite appropriate with regard to mathematical concepts, but should not be applied to technical information such as the commands and syntax of a program. Tracking those things down oneself from a reference manual has no conceivable educational benefit and wastes student time that could be spent more profitably.

One part of giving clear technical information is to make the assignments themselves technically very clear. Another helpful step is to provide students with a brief technical information sheet, such as was described in Chapter 3 and will be discussed in more detail in Chapter 5.

5. For many of us, using technology directly with the students is bad anyway

Several potential difficulties surround the use of in-class demonstrations and/or faculty-supervised labs. First, many of us cannot avoid technical glitches ourselves, or at least fear that we cannot. This fear is not misplaced and it is not just a matter of giving every professor adequate training in use of

technology. As Bill Gates once demonstrated[1] public demonstrations of technology involve a significant risk of failure, even if one knows what one is doing. When an instructor tries doing something on the computer in front of students and things go wrong, students invariably respond internally, and often verbally, with "If the prof can't get it to work right, how does he expect us to do it." The appearance of being ill-prepared or of expecting too much causes students to lose respect for us. Respect is a valuable commodity in the classroom and to squander it needlessly is unwise.

Further, preparing for in-class use of technology can be extremely time consuming. One must plan the actual material, put it on the technological platform, and then try it out in the actual environment. All this can take much more time than simply preparing a good lecture. For most of us spending more time preparing lectures would be more beneficial to the students.

Another pitfall of using technology directly with the students is that it takes up valuable class time. In my experience, most faculty find it difficult to meaningfully cover all the topics they desire in a calculus course, much less to have extra time for additional activities such as demonstrations or supervised computer use. Further, at many institutions, the "technology classroom" is not a sustainable model on a large scale anyway. Not only is the initial cost of out-fitting the classrooms prohibitive, the prospect of maintaining those facilities is daunting.

Of course, using technology directly with students is not a bad idea for everyone. Some mathematicians no doubt make good use of technology in the classroom. Some of them are competent enough that technical problems rarely occur, and some of them have the ability to maintain teacher/student rapport when problems do happen. These individuals should realize, however, that there are many of us who cannot do what they do, not merely because of lack of effort, motivation, or enlightenment, but because of lack of talent. I fear that some should also heed Bressoud's advice to "...be wary of presentations that are too slick. Remember that your job is not to entertain but to get students to think [**Bre99**, p. 180]."

6. Using the software can be easy for instructors

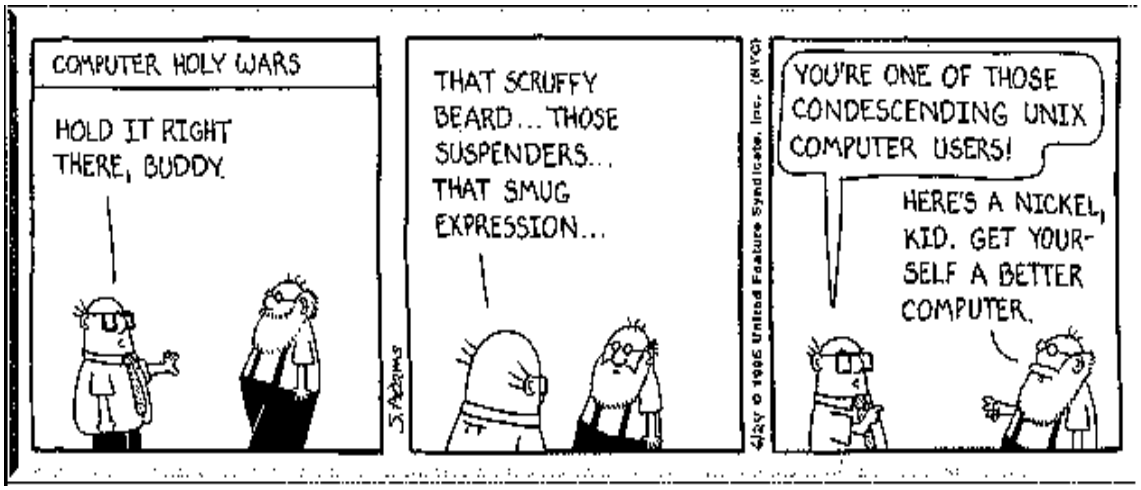
This statement is a simple corollary of the preceding sections and is central to this book. Unfortunately, many infusions of technology in mathematics education that I have witnessed have required far too much time and effort on the part of instructors to be practical on a grand scale. If a professor uses a technology that is not widely available then she must take the time to answer the students' inevitable questions on how to use it. Or, if the instructions are not clear and simple, then she will pay the price both in terms of clarifying the assignments or in grading misconceived papers. This can not only be

[1] I am referring to the technical show in the spring of 1998 when Windows 98 was unveiled. Bill Gates himself was giving the demonstration and the computer crashed. I am told that it was actually a hardware problem, not a problem with the operating system, but that is beside the point. Computers are still a relatively new and rapidly changing technology with not infrequent glitches.

bad for the learning process, but also consume a lot of time and energy. On the other hand, if instructions are clear then the instructor almost never has to deal with technical problems. For instance in the Fall quarter of 2001 I taught a section of Calculus I with 40 students, I had only one student come to me with a problem about MATLAB. It turned out that the student had made a simple typo, was embarrassed by the mistake and that was that. If the software is only used for homework and the homework assignments are centrally managed then most instructors almost never have to deal with the technology directly. This has been the case at Ohio Univ.Three of us wrote a selection of assignments for use in calculus, differential equations and linear algebra and posted them on the web. Other professor use the assignments by simple pointing students to the web site, grading the assignments and then discussing the mathematical issues therein.

If we are to keep the present Researcher/Teacher model that exists in our universities (and I absolutely think we should!) and if at the same time teaching is to evolve to include new technologies, then the technology must be incorporated in a way that isn't time-consuming for professors. The good news is that this is an attainable goal. With the combination of widespread availability of the software, simple assignments, and clear technical directions, demands on professors are minimal. In particular, professors can give homework and not have to worry about questions of a technical nature. Also, with widespread availability and use, the individual instructor can give assignments using the technology without holding class in a computer lab or giving demonstrations.

I taught a course in Numerical Linear Algebra at Ohio shortly after MatLab was adopted. Unwisely, I ignored some of my own advise and suffered the consequences. First of all, since MatLab was relatively new at Ohio, only a few of the students (math undergraduate and graduate students) had happened to have used it before. They did however all have programming experience, so I took for granted that they could figure out MatLab. Wrong! I basically, had to demonstrate the key commands needed in each assignment for most of them in office hours. This was time-consuming and frustrating for the students. The next time I teach the course, more students will have used MatLab before, and I will also give the students the sheet "MatLab for Linear Algebra" shown on page 92.


COMPUTER HOLY WARS
HOLD IT RIGHT THERE, BUDDY.
THAT SCRUFFY BEARD... THOSE SUSPENDERS... THAT SMUG EXPRESSION...
YOU'RE ONE OF THOSE CONDESCENDING UNIX COMPUTER USERS!
HERE'S A NICKEL, KID. GET YOURSELF A BETTER COMPUTER.
J. Adams
4/27 © 1995 United Feature Syndicate, Inc. (NYCI

CHAPTER 5

A computer hater's apology - why technology shouldn't be simply ignored

> "If you owned a company, would you hire a Mathematician who could not compute?" – James A. Yorke, 1995.

The fact that computer use can be simple is not in itself justification for its introduction into the calculus curriculum. Having examined some particulars of using computer technology simply in teaching undergraduate mathematics, it is well to step back and reflect on some more general questions before proceeding to the details of the simple homework approach. In this chapter I hope to discuss some of the broader issues surrounding computers and math education, and at the same time to comment briefly on some of the literature.

1. Do computers help students learn mathematics?

While writing this book I undertook to read a reasonable cross-section of what has been written about the role of computers in mathematics education. I was surprised by how little research I found on the subject. One helpful review article, [**KT94**], explains the lack of technology related research in traditional mathematics education journals as caused in part by the mathematics community's "lack of technological engagement" and in part by the development of more specialized venues for a technologically-oriented audience. It appears that much remains to be done, both in considering how and whether computers aid learning, and in disseminating the results to the mathematics community at large.

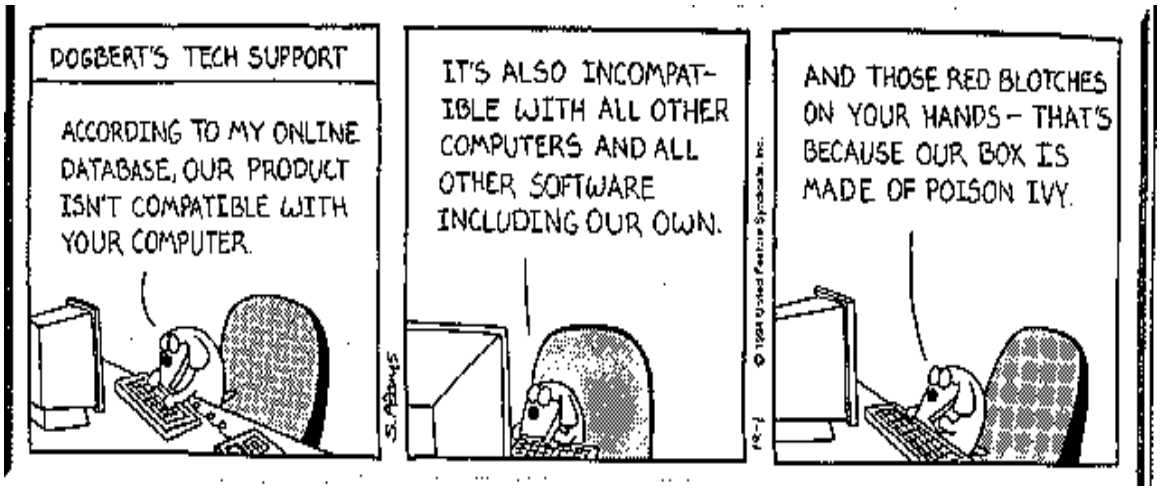
For anyone interested in reading some of the literature, I would suggest starting with the articles by Koblitz [**Kob96**] and Dubinsky and Noss [**DN96**] for some provocative opinions and some very entertaining reading. Koblitz argues against the widespread use of computers in mathematics education. He organizes his discussion around the claims that computers: 1. drain resources, 2. involve bad pedagogy, 3. have anti-intellectual appeal, and 4. corrupt educators. Dubinsky and Noss argue in favor of some uses of computers. For instance, concerning point 3., Koblitz writes that, "Computers are usually used in a classroom in a way that fosters a Golly-Gee-Whiz attitude that sees science as a magical black box, rather than as an area of critical thinking." Dubinsky and Noss counter that, this concern is not unique to computers, which they argue are no less conducive to critical thinking than "having students listen to lectures, follow worked-out problems in the text, or read notes written by a mathematician

who is not paying attention to how learning actually happens." Despite the surface appearances, the authors of these two articles are in agreement on most basic issues. For instance, Koblitz writes that, "What students need in order to become mathematically literate citizens in the computer age is not early exposure to manipulating a keyboard, but rather wide ranging experience working in a creative and exciting way with algorithms, problem-solving techniques and logical modes of thought" [**Kob96**, p. 14]. Dubinsky and Noss agree; "If anybody thinks otherwise, it must be an advertising agent of a computer company, whom we join Koblitz in despising" [**DN96**, p. 20]. Almost all of us would agree, but yet we have to admit that such a cult of "computer literacy" persists in our society and that the advertising agents too often find willing allies among politicians and educators.

Two articles, [**Mee98**] and [**PT96**], that I found enlightening, compared freshman in sections of calculus using the technology-integrated textbook Calculus&Mathematica [**DPU94**] with students in traditional sections. In [**Mee98**], Meel examined differences in the students understandings of limits, differentiation and integration. This article also contains a very good review of several comparative and non-comparative studies. While Meel's study was too small to draw conclusive results, the methods of this study seem to provide a promising model for future research. Results from the author's study and reviewed studies were mixed. Generally speaking, it was found that computational proficiency was not compromised by the use of technology, technology had minimal impact on student achievement on conceptually oriented tasks and Calculus&Mathematica students in Meel's study were found to be more flexible in their approaches to problems. The studies found that students in computer sections had better attitude toward the courses. Park and Travers [**PT96**] reach similar results, concluding that students in the Calculus&Mathematica section obtained a slightly higher level of understanding than students in the traditional course, without loss of computational proficiency.

Other studies I found focus on the influence of computers on narrowly defined aspects of student learning, such as on the concept of functions [**HDP96**] [**O'C98**] [**Zbi98**] and may be too technical to be of much use to the non-expert in mathematics education.. All these studies found that the technology in question had a small positive effect on students learning. Many other writings focus not on the learning outcomes but on the attitudes of students toward the technology, and the effects of the technology on the students' attitudes toward the class [**Edw98**] [**GH98**] [**NH96**]. These studies also report small positive effects.

Other authors have used computers to supplement the teaching of a certain concept or method and report favorably on the results. For instance, Cannon reports that calculators helped with Newton's method and Riemann sums [**Can95**] and Krantz agrees that "Computers are indispensable for Newton's method, Runga-Kutta and Simpson's method [**Kra99**, p. 22]." I think there is probably validity in these claims. It is of course natural that these topics would be enriched by the use of computer, not just because the speed of the computer allows students to carry out multiple iterations of these method, but also because the proliferation of high speed computers make some of these topics relevant to a calculus student in the first place. Even though one can teach these methods effectively without actually using



any technology, I doubt if they can really be appreciated by the students unless their place in the context of computer capabilities is at least discussed.

In summary, there seems to be some evidence that using computers in some ways may have a positive effect on student learning. However, much work is left to be done in verifying and clarifying these results. In the meantime, pedagogy should not, in my opinion, drive large-scale adoption of technology in teaching mathematics. In fact, I think that the red-herring of pedagogy has prevented many mathematicians from seeing the real reasons to incorporate technology into their teaching. In the following sections I hope to outline some of the reasons that should, in my opinion, drive computer use.

2. Why computational software should effect the curriculum

For a long time, some have viewed technology as inevitably transforming the curriculum (see [**Hei88**], [**BK95**, p. 92]). One author puts it: "(A) Introduction of technology is inevitable. (B) Technology forces one to rethink the curriculum, (C) Therefore .... [**Can95**]." While I do agree that we should constantly reevaluate the curriculum, I don't quite agree with this particular argument for doing so. It may well be that our students will use whatever technology is available to them. But, for now, the mathematics community is the guardian of the curriculum and it will not change unless we cause (or at least allow) it to do so. If and how we use technology as part of our teaching is up to us.

While it is important to remember that we control how technology will change the curriculum, I doubt that many mathematicians would argue that changes in technology should have no effect at all. I believe these effects should not be merely pedagogical, but substantive. The point for us should not be the hypothesis that computers may assist student learning in yet to be determined ways. Rather, the point for us should be that the development of new computational technologies, both numerical and symbolic, is very significant mathematically. The speed of computers has fundamentally changed the kinds of computations that are possible and this changes the kinds of mathematics people do [**KT94**,

p. 676]. Since the advent of computers, the ability to do numerical calculations has increased astronomically. In response, fields of numerical mathematics have blossomed in recent years and the use of numerical methods in the sciences and engineering has become ubiquitous. In response to this proliferation, it seems that we should give our students some exposure to the important mathematical concepts, such as accuracy, stability and efficiency, involved in numerical computations. These basic concepts are easily introduced in the calculus sequence, through a few examples of numerical methods.

The invention of symbolic manipulators adds a whole new dimension to the interaction between mathematics and computers. An obvious and commonly cited example of how the advent of symbolic manipulators should affect the content of calculus courses is that student can now find the integral of many functions without learning lots of "tricks" that were a big part of traditional calculus courses. I think that this example only scratches the surface of the issues involved. Research mathematicians, scientists and engineers can now perform many difficult, and even deep and abstract, operations on the computer, something that a generation ago was only a dream of a handful of people. Both applications and pure mathematics that are in effect driven by technology have already emerged. For instance, one of my fellow graduate students used MAPLE (and a couple of really ingenious ideas) to find "piecewise polynomial wavelets" [**DGH99**]. At a recent AMS regional meeting I attended, 2 of the 4 invited lectures involved computation of topological characteristics of the object under study and the computations were performed by computer.

The future of Mathematics is intimately tied to computers and the relationship is two-way (see [**SG94**]). Field Medalist Steven Smale has said that one of the great achievements of the Mathematics community in the twenty-first century was to axiomatize the notion of a manifold and to begin studying manifolds as mathematical objects. In the new century he predicts that the study of algorithms will be a similar phenomenon [**Sma00**].

Returning to the more mundane topic of calculus, the existence of computers should, I think, shift some of the emphasis in the curriculum. By this, I *do not* mean simply shifting away from calculation by hand to calculations by machine. One thing I mean is that it is appropriate to put more emphasis on numerical methods. Specifically, I think that Newton's Method, numerical integration, and Euler's method should be a standard part of the calculus sequence. Further, we should not just teach the mechanics of these methods, but we should emphasize the issues of accuracy and efficiency.

I also think that computers should affect our discussion of many other topics not usually associated with numerical analysis. For instance, sometime during the first quarter of calculus I discuss in my lectures that calculating derivatives numerically on a computer is a tricky business because of division by a small number. I also discuss with my students the general principle that any function that can be written down in terms of elementary functions can be differentiated in terms of elementary functions and it is possible to program all the algorithms to do so. Another example of how the discussion in our classes can be influenced by computers is in the topic of Taylor series. I think that students studying this topic should be aware that calculators and computers use methods related to Taylor series to calculate

numerical values of functions. They should understand that this is because Taylor polynomials involve only elementary arithmetic operations. I begin such a conversation by drawing a triangle and asking the class how a computer could determine a value for the sin and cos functions. Obviously the computer cannot measure the sides the triangle and take ratios. Finally, I think that students should be given an appreciation that Taylor's theorem and similar concepts are necessary for guaranteeing the calculator's numbers are sufficiently accurate. When covering Taylor series in calculus, I give exercises that ask students to use Taylor's theorem to determine how many terms must be used to guarantee a certain number of accurate decimal places.

That mathematics and technology are intrinsically linked is one reason why technology should affect the curriculum. Another way new technologies should affect the mathematics curriculum is by triggering increased attention to concepts necessary for computing well. In sections 4 and 5 of this chapter, I will discuss the topic in more detail, but let me say here that this has hardly been addressed. For instance, among questions raised in [**TOC96**] about directions in research in mathematics education, most questions relating to technology address how computers affect the learning of traditional topics. One, however, is an example of how technology may affect substantive concepts. Namely, "How does technology relate to student's concept of equality vs. approximation?" I think this is a very central issue to the novice learner of mathematics, but one that we take for granted. Indeed my experience suggests that most of our students' understanding of equality vs. approximation is much shallower than we imagine. Technology may exacerbate this by masking the fact that approximation has occurred. The concept of approximation is one of the basic example of concepts needed by students to use technology well. Closely linked to the issue of approximations is the issue of numerical vs. symbolic computations. Since both types of computations are now possible by computer it seems to me that students should understand the difference at an early stage.

Introducing computational software into the curriculum need not mean eliminating existing topics or modes of instruction, though some tension may occur. For example, some view introducing technology into the curriculum as a threat to the teaching of proofs. In [**PP96**] high school teachers presented an anecdote about using a graphing calculator to demonstrate the double angle formula. Several authors interpreted this article as evidence that the proof of the double angle formula was being replaced by a demo [**Wu97**], [**PP96**, p. 165]. While I am not sure that the high school teachers were not undermining the proof of the double angle formula, I am also not sure that the teaching of proofs in high school is something that exists on a large enough scale to be undermined or that proofs *should* play a large role in high school mathematics. While I would agree that proofs should play a role in undergraduate calculus, I do not believe that they should be the primary focus. Calculus was a tool before it was a rigorous system and the vast majority of our students need to *use* calculus much more than they need to prove any of it.

I think it is helpful to keep in mind that calculus itself is by nature just a tool, i.e. technology, for calculating, thus the name calculus. We should not necessarily discount the role of computers as a tool, just because they are relatively new.

All this said, many of our colleagues believe strongly that proofs should be a primary focus of calculus. Whatever one's views of the relative importance of proofs, I find no reason why computational software should make proofs any less important. The only logical implication of computational technology is perhaps in the consideration of which proofs are important. For instance, I think that the proof of Taylor's theorem is more important today than ever before, because Taylor's theorem is a way to estimate the validity of approximations. If I knew an easy proof of the quadratic convergence of Newton's method, I would be inclined to include that in the first quarter of calculus, because speed of convergence of an algorithm is something I want my students to consider as a mathematical problem. Thus, in my view, while proofs are one part of the calculus, understanding the meaning of derivatives and integrals is another, applying these tools to concrete problems is another, and understanding the mathematical concepts necessary for using computational software intelligently is (or should be) another. There is no reason that the learning of one aspect should distract from the learning of others. On the contrary, we should expect students to gain deeper understanding through considering many aspects.

3. Should students learn to do calculations?

Whether or not students should learn to do calculations by hand is a central question in considering the role of computers in mathematics education. On one side, there are some who feel that, in general, learning to do calculations is a waste of time. A strong argument in favor of this view can be found in [**BK95**]. The authors there argue that most of what is taught in the name of mathematics is "to calculate reasonably accurately and quickly." In their opinion this amounts to trying make people into computers. They argue that people do not make good computers and this leads to a high failure rate. They go on to explain that,

> "Calculating is not thinking . . . Thinking while computing destroys accuracy . . . People should spend more time reasoning, making connections and reducing situations to mathematics." [**BK95**, p. 92]

Many of us would respond that teaching people to do calculations should *not* be aimed at accuracy and speed, but at understanding. However, accuracy and speed are easier to test than understanding, and I fear that many of us, myself included, are often guilty of propagating accuracy and speed as the *de facto* goals. We would do well to take warning from the arguments in [**BK95**].

On the other side there are those who argue that students need to learn to calculate just as they have been doing, or used to do, and that the introduction of technology hinders the mastery of computations. One author writes that

> "While our brightest may gain some benefit from doing MATHEMATICA projects, we must be vigilant lest the B-/C+ students substitute skills with button pushing. . . . If you suggest that the computer has obviated the need for facility in arithmetic, algebra and trig, then please provide firm evidence for your views. Part of the reason the New Math floundered was its unyielding contempt for anything remotely resembling "mere rote learning." If "mere rote learning" were renamed "essential drill," we might give it the respect it deserves [**And99**, p. 158]."

As usual, the truth of the matter probably lies somewhere in the middle, but my own tendency is to lean toward the teaching students to do calculations. Many techniques, though far more easily and accurately accomplished by computer, contribute to students' understanding if done the old-fashioned way. For instance, I think most mathematicians agree that Krantz is right when he asserts that "If a student spends an hour with a pencil – graphing functions just as you and I learned . . . the student will also learn to *read* a graph [**Kra99**, p. 27]" and perhaps with the statement "Integration by parts should not be left to the computer." (See also [**Bre99**, p. 180].) In graphing a function by hand, the student is in effect inferring an infinite amount of information from a finite amount of data. That is big stuff. If the students are made to realize that they are doing this and that a computer is incapable of such a leap, they will be more likely to appreciate the process.

Similarly, while symbolic manipulators do allow students to do derivatives and integrals of many functions at the push of a button, if students don't learn a few basic tricks of taking derivatives and finding integrals, they will have difficulty appreciating the intellectual step forward that the topic represents. Students now more than ever need to understand the meaning of the integral in order to be able to use it in applications. For the sake of convenience in class and later classes they need to know how to calculate integrals by hand for some simple functions. More than this, since those using mathematics today have at their disposal technology that not only can do integrals, but can produce misleading errors in doing so, I think that they need to understand the integral even better than previous generations. In order to know what the computer is up to, they need to know about, if not master, the basic tricks of symbolic integration. The technology also requires that they have some basic understanding of numerical integration and its relationship to the definition of the integral. I find that making this connection gives many students the motivation to appreciate the meaning of Riemann sums for the first time.

Another practical reason why we should continue to teach calculations by hand is that our students will need to do simple hand calculations in their subsequent classes. I am currently the coordinator for our Calculus sequence. The corresponding coordinator for the Physics sequence recently met with me to discuss what the Physics department want us to teach students coming into their classes. The basic message he had for me was that students need to be able to do simple symbolic calculations by hand and nothing more. He wanted no technology, no useless theorems and certainly no proofs, only hand computation of derivatives and integrals of elementary functions. While this view is conspicuously

absurd, it is worthwhile noting that many of our colleagues in other departments hold it. We should realize that the ability to do hand calculations makes the educational process more efficient. Even if a student could take out a calculator and determine that $\cos x$ is the derivative of $\sin x$, it is much faster to simply know the fact.

Finally, doing calculations requires mental discipline, itself a valuable goal. Some complain that mathematics courses weed out those who do not master calculations. I am not convinced that weeding out students who do not muster the mental focus to do calculations is entirely bad. I would not be alone if I asserted that a lot of students who fail our classes simply don't work hard enough [**And99**, p. 157] [**Zuc99**]. I do not see why certifying that students are able to discipline themselves and learn to do some calculations should not be part of our jobs.

I believe that there are historical circumstances that have caused the mathematics community to miss the true implications of the development of symbolic manipulators, namely, the fact that calculators became available roughly a generation before computational software. At one time the slide rule was an indispensable tool for the practice of mathematics. Its existence and widespread use made it necessary for students to have complete mastery of logarithms and exponents. In contrast, the calculator takes no particular skill to use, other than the obvious need to know "when to do what." As a result, it is easy to conclude that the effect of calculators on the educational process has been to replace knowledge (facility using exponents and logarithms) with button pushing. Most of us have assumed that computational software would just replace more skills and many of us have concluded that it is our duty to oppose that replacement. What mathematicians in general have not considered, however, is that computational software actually requires a step back toward the need for skills due to the nature of operations such software performs. A chief difference between calculators and symbolic manipulators is that calculators only do operations for which there is a fail-safe algorithm. Computational software does things for which a fail-safe algorithm does not exist. Both symbolic and numerical integration are examples of this.

To illustrate, Jim Shultz [**Shu99**] has proposed the following list of things we should know about **square roots**:

- What the concept of **square root** means.
- How to find simple **square roots** mentally.
- How to find harder **square roots** using technology.
- How to estimate **square roots** (as a rough check on the technology).
- How to solve problems involving **square roots**, especially *when* to take square roots.

In particular, students do not need to learn techniques to find square root by hand as calculators can always be employed to find the square root with great speed and perfect precision to some prescribed (and adequate for almost all purposes) number of decimal places.

One could substitute any number of mathematical operations into this list, including any of the usual functions performed by a non-graphing scientific calculator. What these functions have in common is that they have an algorithm that works every time to any degree of accuracy. If one were to create a similar list for integration, then one would need at least to add:

- Roughly how a computer finds a definite integral numerically and ways in which this might go wrong.
- Not all functions have indefinite integrals in terms of elementary functions.

Admittedly, some calculations are destined to be dropped. Learning to calculate square roots by hand has silently disappeared from the educational process since the time I was in high school, and the loss has not had any significant consequences as far as I can tell. In the end, the Mathematics community must decide what, if any, algorithms are important for students to learn to do by hand. It may happen that we make these decisions over time without realizing that we are doing so. Processes that are the best candidates for discarding are those that: (a) have a fail-safe algorithm that can be programmed, (b) the algorithms are not instructive in themselves, and (c) are not central to the mathematics at the particular level. Perhaps the reason the loss of computation of square roots by hand from the curriculum has happened so quickly and silently is that it satisfies all three of these criteria.

4. Teaching people to use software intelligently is important

Since computational software exists and is powerful, people will attempt to use it. Since using it well requires some mathematical sophistication, it naturally is up to the mathematics community to provide people with that sophistication.

Precisely because the new computational software is so remarkable, to use it properly requires training. Consider again the example of a calculator. Although a calculator alleviates the need for most people to do long division, a calculator is useless to the person who does not know to when to add and when to divide. That knowledge can only come from an understanding of the meaning of addition and division. The user of computational software must understand a larger number of things. First, since both numerical and symbolic operations are possible, the user must understand the advantages and disadvantages of each, or, at the very least, know what each means and be able to identify which the box has performed. Thus, not only must the user understand what she wants to calculate, but must also have some understanding of how the computer is going about its work. Krantz [**Kra99**, p. 22] argues that one does not need to know everything about how a car works in order to drive and that, similarly, one need not know how a computer works to use it. While it is absolutely true that most of the workings of the computer are irrelevant to the user, when one uses computational software it is often the case that one needs to understand at least something about the algorithm the computer is using. Without such an understanding it is quite possible to believe that the computer is doing what is desired, when in fact it is doing something completely different. Results of these computations can be

entirely misleading or even wrong. As in the example of the bad graph, Figure 7 in Chapter 3 shows, even something as simple as plotting a function can go wrong when technology is in the hands of someone not prepared to use it. Fortunately, most problems of this nature can be avoided if students master a few basic mathematical principles about the software. Even mathematicians unenthusiastic about use of the software as an educational tool should recognize their obligation to ensure that our graduates have the mathematical maturity to use computational software when they encounter it in the workplace. Mathematicians should not try to sell students on the value of computational software in the workplace, but rather we should teach them the mathematics necessary to use the technology correctly.

Many people in the general public and some in universities have the view that people need training to be comfortable with computers. According to [**BK95**] one of the goals we should be striving toward is to produce "People who are interested in and willing to apply and use technology to explore mathematical ideas and compute mathematical conclusions." I don't think that getting people "interested in" and "willing to" use technology is even an issue. From my experience, people are very glad to employ technology whenever they think it will make their work easier. The problem is getting these people to think when they employ technology. This is not accomplished by mere exposure to the software, but by grappling with the underlying mathematical concepts.

If mathematicians do not teach people to use computational technology correctly then, in all likelihood, no one will. Professors from other departments are traditionally not interested enough in mathematical subtleties to bother with these issues. On the other hand, they quickly detect the erroneous results when they encounter them in a student's work and they quickly, and correctly, place the blame for these mistakes squarely on the Mathematics Department. The teaching of mathematical understanding is our business, whether it expresses itself in hand computations or in computer computations.

Returning to the car analogy, in [**Kob96**], Koblitz denounces the "inevitability argument":

> "Perhaps the most frequent argument for computers in the schools – and perhaps the most illogical – is the inevitability argument: 'Calculators/computers are going to be everywhere, so we might as well incorporate them into math class. One can't fight against the tide.' Using the some reasoning, one could say that, since automobiles are everywhere in our society and play a crucial role in all of our lives, therefore cars should be used as much as possible in education, and driver education should be regarded as a centrally important subject in school, much more than such relatively useless subjects as music and art. The inevitability argument for computers in the schools is exactly the same sort of anti-intellectual *non sequitur*."

While I agree with much of this statement when applied to many uses of computers in education, when it comes to computational software, this argument breaks down. Namely, there are plenty of opportunities for young people to become safe drivers, including drivers' education courses in schools. In the

same way there are also plenty of opportunities for students to learn to use the web, gain experience with different operating systems and use various kinds of software. There are not many opportunities for students to become "safe" users of computational technology, other than in our classes. Further, as I argued in the previous sections, technology is an inherent part of doing math. There is no parallel principle in the automobile analogy; automobiles are not inherently tied to art and music, as computing is to math.

Finally, even if students never use computational technology directly in their careers, there are still reasons for them to gain an understanding of it from our classes. The analytic skills learned from understanding computational software are valuable to prepare our students for the task of self governance is a world in which careful scientific thinking (as well as an awareness of the limitations of technology) is increasingly important. I think that we are especially well positioned to propagate a healthy skepticism of technology and it is our responsibility to do so.

5. Simple computations can contain good mathematics

The only reason I *like* using computational software is that I can use it to teach interesting mathematical ideas. Presumably, all of us share the joy of teaching mathematical ideas.

There are many notions of what qualifies as interesting mathematics, but I find that the mathematical notions needed to use computational software intelligently are in themselves quite interesting. For instance, students leave my courses having seen examples of problems where symbolic solutions are impossible, where symbolic solutions are obtained but are practically useless, and where numerical solutions go wrong. Many students (the interested ones) leave knowing why these examples happen. All kinds of examples like these can be achieved with simple computations.

Mathematical results regarding nonexistence of solutions are not generally associated with computations, but are in fact very important in understanding how to compute. That symbolic solutions are not always possible is something that most students do not know when they come into my classes. If symbolic solutions were always possible (as was thought in some fields of mathematics up until the last century), how different would mathematics be today? Numerical Analysis would not be a field! I think that unsolvability is so important, I often tell my classes that the unsolvability of the quintic is the "Fundamental Theorem of Applied Math." If such an ideal and simple situation requires numerical methods, how much more does an "applied problem" with non-integer coefficients.

I personally find that the mathematical issues of computations themselves are so interesting that I most often emphasize those issues in the computer assignments. I do, however, sometimes use the computer assignments to highlight other mathematical issues and I am certain there are enough good mathematical ideas out there that any math instructor can find plenty of material to that will interest both themselves and their students. For instance, when I teach sophomore level differential equations, I like to use computational software assignments to demonstrate to students the difference between linear and nonlinear equations. In linear algebra courses I like to have students plot two sets of polynomials

on the unit interval, one set the monomial, the other a set of orthogonal polynomials. I find that seeing such an example with polynomials gives students their first understanding of the usefulness of abstraction. (I would never teach a Linear Algebra course without polynomials as a core example of a vector space. Without such an example, the abstraction of vector spaces is pointless!)

When examples of simple homework assignments are presented later in the book, care will be given to point out mathematical ideas contained therein.

CHAPTER 6

Simple Homework

> "..., it has become increasingly evident that the technology altered the nature of the activity using it." – James Kaput and Patrick Thompson [**KT94**]

1. General principles for simple homework

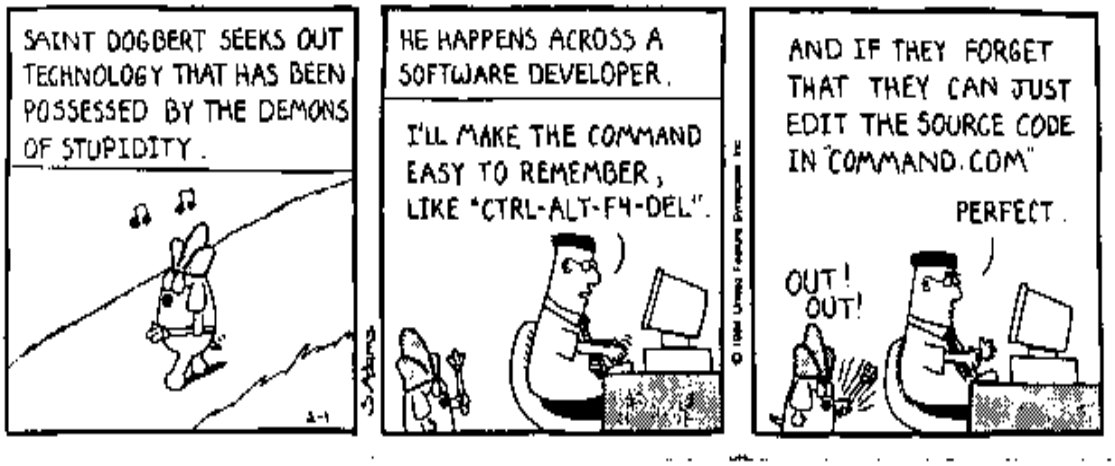
A place to start in making simple homework assignments is to set simple goals. Others have also found that simple goals contribute to success in incorporating technology in the mathematics curriculum [**Can95**]. I adopt the following set of goals for computational software homework in freshman calculus courses:

(1) The students will become familiar with some basic commands and syntax of the program.
(2) The students will realize that care should be taken in using mathematical software and will understand a few of the mathematical ideas needed to use the software intelligently.
(3) The students will come to understand the difference between numeric and symbolic computations.

I would suggest that goals 1 and 3 from this list are essential to an effective "simple approach." Goal 2 might be deleted, appended to, or replaced by other goals that suit the instructor's personal taste. Mathematical enthusiasm can make it difficult to avoid aiming for too much from the assignments. One should exercise caution. Once you have arrived at simple goals, the process of making assignments is much easier.

Here are some guidelines for making assignments simple:

- **Just do basic calculations.** Good candidates for assignments are calculations that they are most likely to use later. Generally, one should only do computations that the students could, at least in principle, do by hand. This allows them to understand what the computer is doing.
- **Keep it short.** Don't do too many calculations and don't try to cover every aspect of the commands. One should definitely resist the temptation to introduce all the "options" of a command.
- **Keep explanations to a minimum.** Unfortunately, students don't really read or understand a lot of explanations anyway. Let the computations themselves demonstrate the mathematics and the software.
- **Include good mathematics in the basic calculations.** In my opinion, if you don't do this, you really miss the main point of doing computer work. The best way I know to include good



mathematics is to include examples where things go wrong and require the students to think about why. These examples are often very instructive in that they illustrate the process the computer is using. Finding good examples often requires experimenting with the software yourself. Understandably, the documentation of the producer never includes examples where things go wrong.

Here is an incomplete list of things one can include to inject good mathematics into simple computer homework:

(1) The same thing symbolically and numerically. The difference is often instructive.
(2) Approximation vs. equality.
(3) Unsolvable problems; integrals without closed form, quintic equations, nonlinear ODEs, etc..
(4) Numerical approximations that are inaccurate.
(5) Problems whose complexity is beyond the present computing power or iterative processes with too many steps, etc..
(6) Misleading graphs.
(7) Effects of grid or sampling size.
(8) Convergence of graphs.
(9) Parameter dependence.
(10) Linear vs. Nonlinear behavior.

- **Provide very clear instructions.** Tell students exactly what commands to use and give clear simple examples.
- **Provide very clear, concise and complete technical information.** Most of this is covered in the technical info sheet distributed to students at the outset, but, do not be afraid to be too explicit in the assignments themselves. For instance, at least for the first two assignments of a term I include in the assignment the phrase "type at the prompt and press Enter."

Of course, as the term progresses you may begin to do more complicated assignments and assume more on the part of the students. This is also true if your students are more advanced and it is particularly true if all your students have already been in classes using the same software. In an ideal situation where all the calculus students at your university are using the same software throughout the calculus sequence, and the first year was done well, you could assign very advanced things in the final course of a calculus sequence, if you are so inclined.

If you write assignments yourself or obtain them from a source that may not be up to date, you should always perform the calculations yourself on a machine with about as much computing power as students are likely to have available. I learned this lesson the hard way at Georgia Tech. I once assigned an example directly from the Mathematica Book [**Wol96**] that crashed on the machines in the Math Department's lab. Although I sometimes make machines crash on purpose in an assignment, the students were not amused that I had not tried the example myself and did not anticipate the problem. Unless you are very certain about the source, there is a chance that the software has changed since the assignments were written. Some examples may not work the same way they were intended. The only time I ever got away with not trying the assignments myself was at Northwestern when MAPLE was being introduced. The same assignments were required of every section and the person who wrote them was extremely reliable.

Some examples of simple homework that I gave to students at Georgia Tech are on pages 48, 49, 50, and 51. Because I did not anticipate using the exercises repeatedly, I wrote them by hand to save myself time. This is another example of technological minimalism which saved me a great deal of time.

2. Technical Information Sheets

The technical info sheet was an integral part of Mary Beth's original plan and is an important part of making a simple approach possible. Her "MATHEMATICA Info Sheet" contained in Chapter 3 was the prototype for this concept. Other examples of technical info sheets that we have used are contained in the Appendix B.

The Technical Info Sheet should serve as a very succinct users' guide to the software. It should address almost every problem the student might encounter in using the software throughout the course. If it is done well, this can be accomplished in about two pages. In fact, for most software packages, it is possible to compile on one page most of the commands that most users will ever use. This is an approach that is seemingly foreign to documenters of software.

The Info Sheet should be handed out at the first of the course and you should emphasize to your students that they should rely on it throughout the term. Students often lose them anyway, so keep some extra copies on hand.

There are three components that should be included in the Technical Info Sheet:

(1) How to access the software in at least one place at your institution.

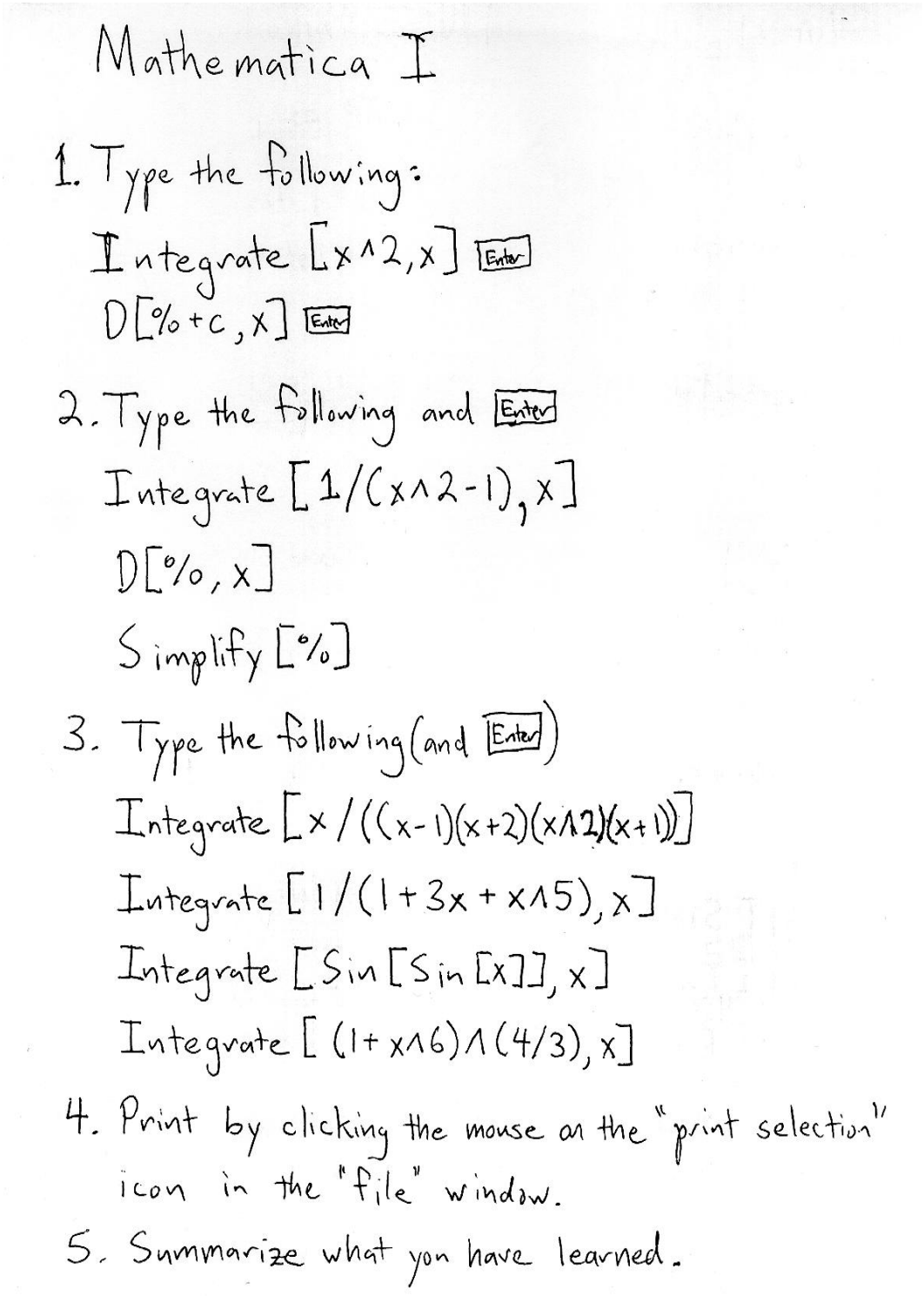


FIGURE 1. A simple homework assigned by the author to an integral calculus class at Georgia Tech. The goals of this assignment are to familiarize students with doing integrals and to lead to a discussion of the fact that some functions cannot be integrated in term of elementary functions.

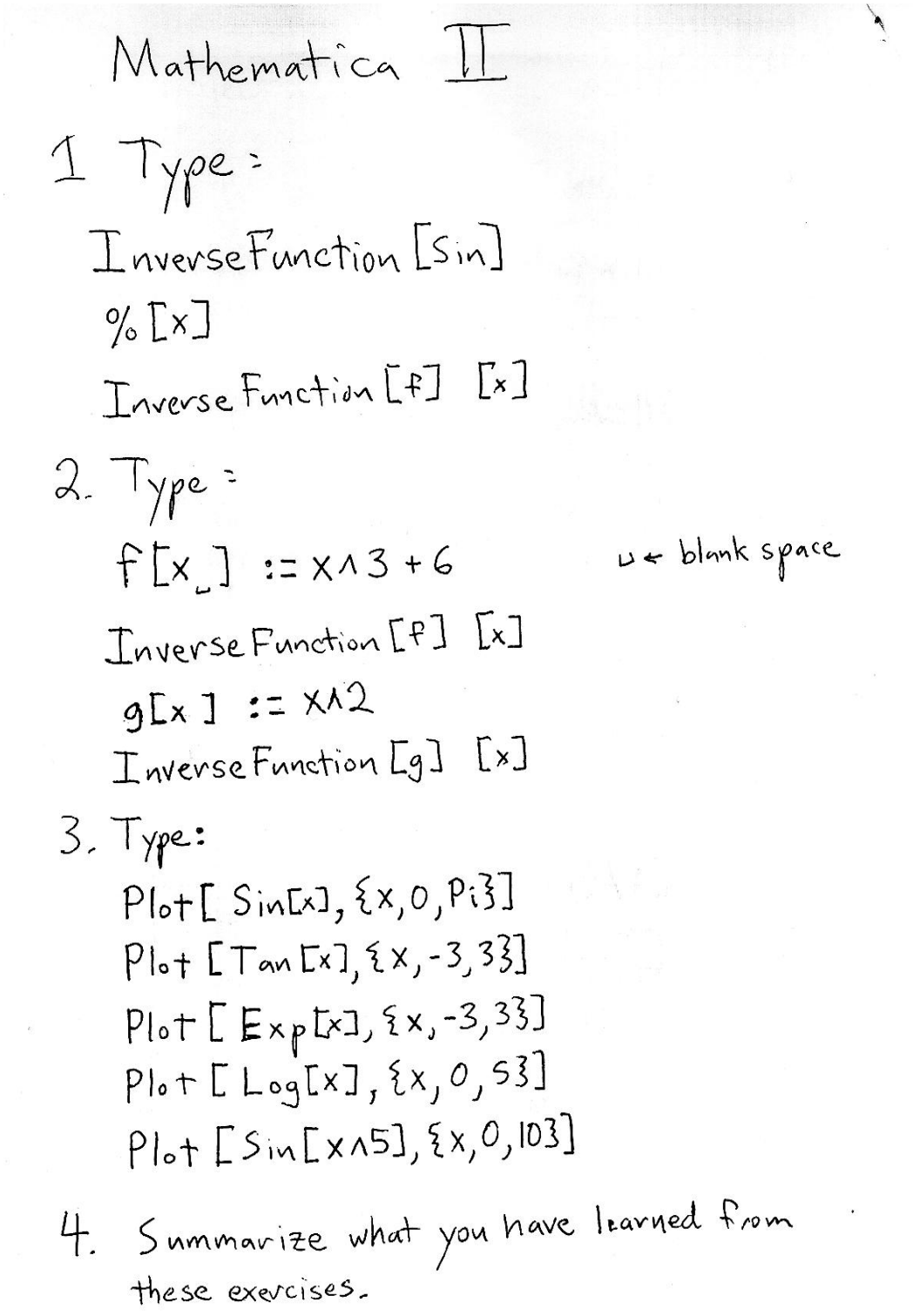


FIGURE 2. The second homework for the same integral calculus class. One of the goals of this assignment is to lead to a discussion of why one of the plots is bad and how computers plot functions.

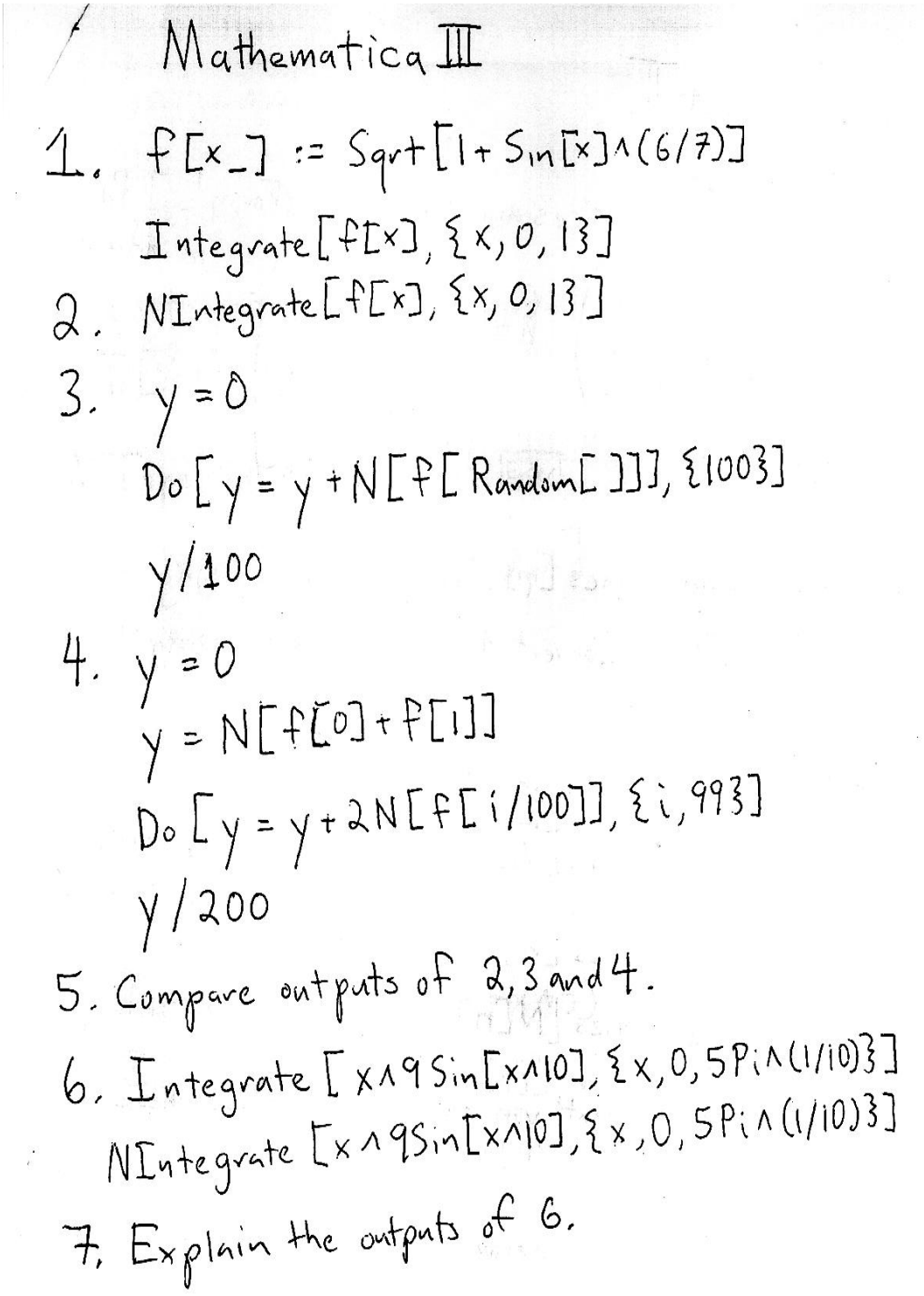


FIGURE 3. The third homework for the integral calculus class. The mathematical ideas are the difference between symbolic and numerical integration (NIntegrate). Part 6 demonstrates that numerical integration may fail. Monte Carlo integration is also introduced and students are generally very surprised by this concept.

Project II

1. Input the matrices and type MatrixForm[ ] for m and n.

$$m = \begin{pmatrix} 3 & -1 & -1 \\ -1 & 0 & 2 \\ 1 & 1 & -3 \end{pmatrix} \quad n = \begin{pmatrix} 4 & 8 & 0 & -3 & -1 \\ 3 & -2 & -2 & 4 & 4 \\ -1 & 1 & -1 & 0 & 1 \\ -2 & -1 & -3 & 0 & 1 \\ 1 & 4 & 1 & -3 & 0 \end{pmatrix}$$

2. Type Eigenvalues [m]. Try to simplify using Simplify[%]. Convert to numerical data using N[%].

3. Type Eigenvectors [m].
   Type Eigensystem [N[m]]. (This make m a numerical matrix, then calculates)

4. Type Eigenvalues [n].
   Type Eigenvalues [N[n]].

5. Summarize what you have learned from these exercises.

FIGURE 4. A project given to a Linear Algebra class at Georgia Tech. A goal of this assignment is to lead to a discussion of the fact that the eigenvalues of an $n \times n$ matrix with $n \geq 5$ cannot in general be found exactly.

(2) General principles of using the software.
(3) Basic commands, including at least all the commands that will be used during the course.

Including information about accessing the software is essential for simplicity for students and instructors. It was usually the case in my education that this information was not included. When it is not, students spend lots of time trying to obtain the information and will often come to the instructor in the end anyway. You might as well give it to them in the beginning. Lately we have begun giving this and other information on a separate information sheet entitled "**MATLAB** homework at Ohio University". A copy of this sheet is included in Appendix B.

General principles of using the software should include at least the following:

- How to enter commands.
- Any peculiarities of syntax.
- Any particularly useful things.
- How to include text (if possible).
- How to edit.
- How to save and print.
- How to get help.
- How to exit.

An example of peculiarity of syntax in MAPLE is that all commands must end in "`;`" (or in "`:`" if one wants to not see the output). An example of a particularly useful thing would be how to refer to output of previous lines. In MATHEMATICA and MAPLE, the symbol `%` is used for this purpose.

One can also include facts that might guide users to more advanced topics. For instance, I mention in my introduction to MAPLE that it can be used as a programming language. I also mention the existence of "packages" that can be used for many advanced (and some not so advanced) calculations.

Some mathematicians have objected to my info sheets on the grounds that I tell students the commands for things that have not yet been covered in class. For instance, at the beginning of differential calculus courses I hand out my sheet which happens to contain commands for integration. My answer is that I view the info sheet as a universal basic users' guide that students can take with them and they can use in most situations they will ever encounter. With this as my goal, including commands that are advanced - relative to the students - is unavoidable. All software documentation necessarily includes material beyond the range of the beginning reader.

If you are technically gifted, *avoid the temptation to put too much in the info sheet*. There are many things about the software that might be fascinating to you, but that are not necessary for the purpose of basic use. For instance, someone accustomed to programming in MAPLE probably feels that the difference between local and global procedures is of crucial importance, but these concepts are not relevant to the average user and not appropriate for the technical info sheet.

Keep in mind also that many things are more easily understood by example than by explanation. For example, in MAPLE an object can be either a function or an expression. I mention this fact in the technical info sheet and give an example in which the command `unapply` is used to change from an expression to a function. I do not, however, try to explain the functionalities of expressions and functions. Students pick that up from the names and from using them. In my opinion, detailed explanations of such concepts are not effective anyway. Learning a computer language via detailed explanations is not unlike learning a spoken language by discussing it's grammar, instead of listening to and speaking the language.

3. Using simple homework

I would describe my own teaching style as employing traditional lectures with some group work added. From what I can ascertain, most professors in the United States employ some type of lecture format with small perturbations. Simple homework can be added to this teaching style with no difficulty at all. In fact, any teaching style could be easily supplemented by the use of simple computational homework. In adding the simple computer homework to the lecture format I take the following steps:

(1) Hand out the Technical Info Sheet, "Where to obtain " info sheet at the beginning of the course.
(2) Hand out four or five assignments during a quarter.
(3) Collect, grade and return the assignments.
(4) Discuss in class the mathematical content of the assignments.

The procedure has proved sufficient to achieve the basic goals of the assignments and also has been extremely easy to implement.

From my experience, this last step, discussing the assignments in class, can lead to making computational issues a part of the rest of the class. For instance, in a Linear Algebra course, after giving the students a homework assignment involving determinants of matrices that are too large, the question of size of the matrix and practicality of methods can be raised over and over again. As another example, when I teach Newton's method in the first calculus course, I do not assign any computer homework related to the method, but my entire discussion of the topic is inundated with references to computer implementations. Having done computer homework on other topics makes this discussion possible. Typical questions I ask the students in class are: "If you were programming the next generation of TI calculators to use Newton's method for such and such problem, how many iterations would be enough?" and, "If you were programming this, how would you make the program come up with a good initial guess?" We always discuss the speed of algorithm in this context. I find that students are always fascinated by these discussions.

To implement simple computer homework throughout a department can be easy. A few professors can coordinate the production or procurement of simple assignments and the distribution of information

throughout the department. Most professors can simply take the steps mentioned above, possibly without any knowledge of the software themselves. Professors can assign homework by giving students a web address, such as `www.math.ohiou.edu/~simple` or an address set up internally by a department. We have such a page which may be viewed at `www.math.ohiou.edu/~matlab`. Alternatively, and I like this because it is low-tech, a professor can simply hand out photocopies of assignments from the web for this purpose or from a selection of hardcopies of assignments can be kept in a central place in a department

One additional idea is to conduct large demonstration sessions on the software at the beginning of each term. This was found to be helpful at Northwestern University and elsewhere [**Can95**]. While I am skeptical about how much students can actually learn from such sessions, the sessions may be beneficial in alleviating some student anxieties about the software. Holding sessions might also act as an advertisement to the administration of what the math department is doing to improve undergraduate education. Holding the sessions at various locations on a campus might also help to make professors from other departments aware that their students are learning to use the software. While such sessions may sometimes be helpful, they are by no means essential. These sessions will only be effective if led by someone with both technical savvy (to avoid the fear-and-disrespect inducing public glitch) and good communication skills (to avoid producing more confusion than clarification.) At Ohio University we have not held any such sessions, mostly because we lack someone qualified to lead them. However, we have not found the lack of public demonstrations detrimental.

4. Training your colleagues to use simple homework

Once a department has decided to use simple computer homework in calculus, it must face the problem of training it's member to implement the plan. This too can be done in a simple way. We have found that the training can be accomplished with the right approach and materials in a one hour introductory session on giving the homework. The main key to this working well is to focus the session not on software itself, but simply on how to give assignments to students. In fact I make it a point to my colleagues that the assignments can be given with no actual knowledge of the software on the instructor's part. A few have taken me up on the challenge that they could give and grade the assignment we have written without learning anything about the software themselves. They have been satisfied that it is indeed possible.

If the instructors do not have to know how to use the software, what do they have to know? What we give them in the training session is a one page sheet "How to give MATLAB homework", which is shown in Appendix B. We talk them through the sheet. This usually raises a few predictable questions like "how can I grade the assignments if I don't know how to use MATLAB?" My answer is that the mathematics behind the commands is all they need to know, the assignments themselves are self

explanatory to any professor of mathematics. Then we let them do the assignment “Defining, Evaluating and Plotting Functions” to see how easy and self explanatory it is. There are usually one or two problems with typos and we point those out.

CHAPTER 7

Some practical details and peripherals

Teaching should be such that what is offered is perceived as a valuable gift and not as a hard duty" - Albert Einstein

1. Group work

Allowing students to do the computer homework, (or any other type of work) in small groups of two or three students has two distinct advantages: it promotes cooperation among students, and it makes grading easier. It is now fairly widely accepted that promoting cooperation between students is beneficial to the learning process. The American Association for Higher Education lists cooperation among students as the second principle in its Principles for Good Practice in Undergraduate Education (Appendix A). Group work promotes a better social atmosphere, which has been linked to improved retention and to an increased likelihood of choosing to major in mathematics (see [**LK96**] and [**Lei96**]). As to grading, it is surely agreed among research mathematicians that time spent on grading is time that could be spent in more enjoyable and/or productive employment. I would much rather allocate my time to preparation for teaching or helping students during office hours than to grading homework.

My own experience with group work, both for computer assignments and for other assignments, has been overwhelmingly positive. Students regularly write good things about the experience. Some have written that they were skeptical at first, but in the end viewed it as one of the best parts of the class. As years pass, I get fewer and fewer students who are even skeptical at the beginning, probably because the practice has become increasingly common. The advantages of using group work are not limited to the assignments themselves. Having regular group assignments also encourages cooperation by students during other phases of the course. I find that it even improves the dynamics of my lectures in that students are more comfortable with each other and with discussing mathematics.

An oft-cited disadvantage of giving assignments as group work is that not everyone will fully participate. This is perhaps unavoidable, but in my experience students do not in general let each other get away with this very much. Even when one person dominates the work, the others almost always make sure they understand what is going on and the "leading" student is happy to explain. This is precisely the dynamic that we want to encourage. It has also been my experience that excessively hard and long assignments tempt students to split up the work. When assignments are simple and the overall time-investment is reasonable, students generally behave well on group assignments.

Despite my enthusiasm for group work in general, I have after several years, stopped giving simple computer homework as group work, because doing that means that not every student actually enters the commands. I am now convinced that the act of typing the commands oneself is critical to the goal I have of familiarizing the students with the commands and syntax of the software. I do not abandon the group concept altogether with regard to computer homework. Instead, I employ an intermediate alternative, requiring everyone to turn in their own write-up, while allowing and, in fact, encouraging people to collaborate. Thus, I encourage students to consult with one another on the computer homework, but ask them to physically go to the computer and input the commands themselves. Although, this option loses the advantage for the instructor of fewer papers to grade, it may still reduce the grading time somewhat, as collaboration may tend to eliminate some errors. It does have the advantage that it requires all the students to write, which is pedagogically desirable (see below).

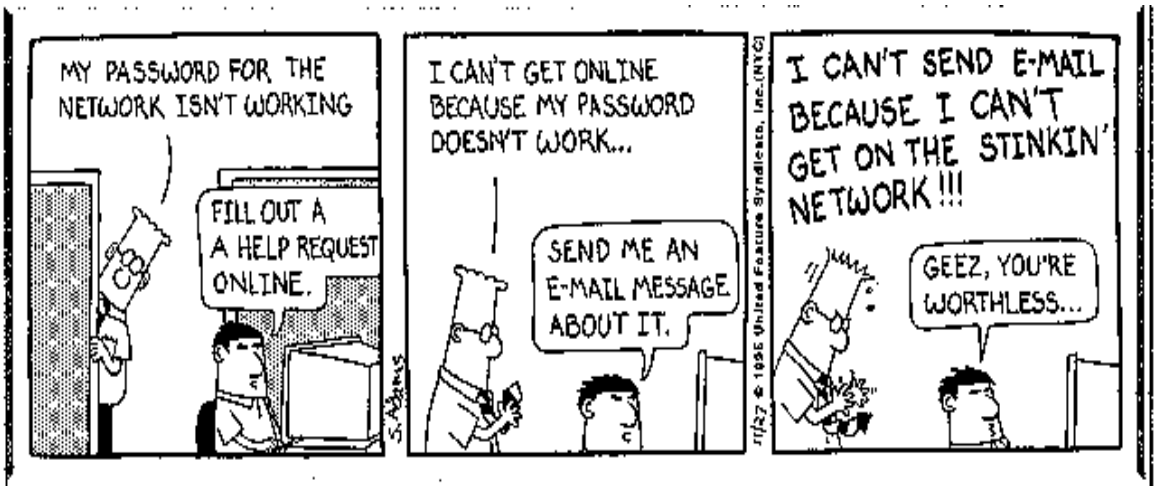
The fact that I am so overwhelmingly positive about group work, yet do not employ it for computer homework, requires some explanation. Let me state an obvious, but sometimes overlooked fact: *We should not feel obliged to employ ever positive educational principle in every context.* The initial MATHEMATICA projects at Georgia Tech contained many good educational principles, yet the end result was something too complex to be useful. Computer homework in mathematics especially requires simplicity. I continue to use group work extensively in other contexts, but do not choose to use it for computer homework. This theme will be continued in the next section on student writing.

2. Student writing

It seems that experts in college mathematics education increasingly believe that writing is an important activity for students. These conclusions are not surprising. Indeed, it only makes sense that as students write about mathematics they understand it better. It is also argued that writing well is a central part of higher education and that in order to develop this skill, students should be expected to write within many contexts.

Unfortunately, incorporating writing into a mathematics course can be difficult, and I have yet to find a completely satisfactory way to do so in computer homework assignments or otherwise. The biggest obstacle is of course that grading of student writing is difficult and time consuming. Closely related is the fact that most students are not very good at writing and they do not, in general, like doing it. I have heard particular objections to being required to write well in mathematics courses. I have actually heard students say things such as: “If I wanted to write, I wouldn’t be an engineering major.” These objections are a good segue to a mini-lecture about how communication is important in even the most technical job, but the fact remains that many students in math classes do not have an affinity to writing.

In order to make some modest effort in the right direction, I include small essay type questions on computer homework. Here are some examples:



"Explain what happened. What is the relationship between solving and factoring?" See page 97.

"Try changing the initial conditions as in 1. How does this effect the solution? How does this differ from the linear case?" See page 110.

Questions such as these promote some writing on the part of the student. I have encountered mixed results with this approach. Some students do fairly well, some students do not seem to take the writing very seriously. A colleague of mine has suggested that I make it clear from the beginning that the writing is important and design the grading system to reflect the emphasis. I have begun to implement this advice.

All this said, let me add a word of caution. Instructors often try kill two birds with one stone by doing computer work with a large writing component. In my experience, this is often associated with the "project format," the disadvantages of which I have discussed. Trying to make a single assignment incorporate extensive writing, good mathematics and interesting features of the software may make the assignments unnecessarily complicated. Further, since students don't like to write, relegating the writing to computer assignments can give the computer assignments an unnecessary bad association. If we are to take writing seriously, and I am becoming convinced that we should, then we should emphasize writing throughout a course, not just in the computer assignments.

3. Cheating

During the early stages of the MATHEMATICA projects at Georgia Tech, there were reports of students copying projects from other students. Further, since the projects were published in a book that was used every quarter, it was suspected that students were even getting completed projects from their friends who had taken the courses previously. Such problems are difficult to cure once they occur, therefore prevention is needed. Unfortunately, this means that an instructor using the project format, is probably well advised to write new projects each term. That takes too much time for my taste.

A nice effect of simplicity is that it reduces the temptation of cheating. If the homework requires an hour or less to complete, only the most dedicated cheaters will bother to get the assignments from others. If you require each person to provide a printout, then even if the cheating student gets another person's assignment, the student still has to go to a machine and input the commands and paraphrase the answers. In my view, doing these things is about half of the point of the assignments anyway.

4. Extra credit

If you are only one of a few instructors in your department requiring computer homework, or if your department is just beginning to use computer homework, then you are likely to encounter some resistance from students, even if you use a very simple approach. A technique I have used often is to give the computer assignments as extra credit. Although this is essentially a "trick", it seems to hold some psychological advantage, the students feel that you are giving them something, and that squelches most complaints about the assignments.

A problem of assigning the computer homework as extra credit is that not all of the students will do it. For some reason, it is often the students who could most use the extra credit who do not bother to do the assignments. I try, somewhat successfully, to alleviate this problem by strongly encouraging the students to take advantage of the opportunity. Also, as assignments are turned in, discussed and the mathematical knowledge associated with them becomes incorporated into the course, most student get the message that the computer homework is important.

5. Posting the homework on the web

A danger of posting assignments on the web is that students may simply cut and paste commands, thus bypassing the process of typing the commands themselves. When one of the goals is to familiarize students with the basics of the software, the act of typing the command oneself requires the engagement of parts of the brain which would not be used by simply cutting and pasting, and educational experts tell us that the more senses and areas of the brain are engaged, the more effective the learning process.

On the other hand, posting homework on the web may have some advantages, such as drawing a captive audience to your webpage. I will discuss this more in Chapter 9. If you do post homework assignments, you can make it more difficult to cut and paste by using PDF or Postscript formats. Posting assignments directly as "notebooks" or "worksheets" makes it particularly easy to cut and paste.

6. Choosing software

Remarks in this section are based on my limited experience and are intended only to give a very brief summary of some of the available software from the perspective of a technophobe looking for simplicity. I do not wish to promote any particular package. In fact, let me say up front that for the purposes of simple homework, almost any of the available packages with symbolic manipulation

are adequate. Practical considerations having nothing to do with the use of the software for classes should have a large influence on the choice of software. For example, some professors may already be using particular packages, feel loyalty to the package and may even be willing to help in the effort of introducing the software. Existing use of software packages by other departments should also be taken into account. Another important consideration should be the fact that once a software package is chosen for calculus, it becomes ingrained in the department's culture, and thus becomes the package that will be most accessible for research use.

I have found that many people who use computational software regularly tend to develop a strong opinion that the package they use is the best one. I attribute this partly to the fact that using a particular package seriously requires a good amount of startup time. Once we have made such an investment, we like to believe that it was wise. Also, with experience, we become used to the idiosyncrasies of the package. Brand loyalty could also be due in part to the fact that mathematicians just like to hold and express strong opinions about lots of things.

As I said in Chapter 4, in an ideal world, a mathematics department, in consultation with other interested departments, would decide on a standard software package that will be used throughout the calculus sequence and that would be available to students at computers everywhere on campus. The same software would then be used widely in subsequent math or math-using courses. I realize this may be asking a lot politically, but to do any less is to introduce inefficiency in the learning process and frustration for both teachers and students.

Symbolic computation in general is very new; the first symbolic manipulator became available in the late 1980's. Today several symbolic packages are available, including: AXIOM, DERIVE, MACSYMA, MAPLE, and MATHEMATICA. I am most familiar with MAPLE, MATLAB and MATHEMATICA and those two appear to have the largest share of the market. In addition to these symbolic packages one should seriously consider technologies that are mainly numerical, but also have symbolic capabilities. Among these are MATLAB and the newest calculators, the TI-89, TI-92 Plus and HP-49G. All the symbolic manipulators I have seen, except for the calculators, have excellent graphics.

Let me begin with the first package I used, MATHEMATICA. MATHEMATICA's basic document is called a notebook. The user may enter both executable mathematics commands and text into a notebook. A notebook may be saved, edited and transfered over the Internet, to be read and used by anyone with MATHEMATICA.

MATHEMATICA's help utilities are virtually useless. Much like UNIX's "man pages" the only way to get help in MATHEMATICA is to know a specific command name. This is fine once you are an expert, otherwise you are in trouble.

The thing I like best about MATHEMATICA is that it has a good balance of symbolic and numerical calculations. Almost every symbolic command in MATHEMATICA has a numerical version which is invoked by the same command name but with a capitol N at the beginning. For instance, " `NIntegrate` " is the command for numerical integration. Since one can move effortlessly between

symbolic and numerical calculations, it is a simple matter to make homework assignments for calculus that highlight the symbolic/numerical paradigms. Further, MATHEMATICA's numerical routines are fast. MATHEMATICA seems to be the most expensive package available, but in my opinion, it may well be the best.

One should probably discuss MAPLE and MATLAB together. MAPLE is primarily a symbolic program that can also do numerics and MATLAB is primarily a numerical program that can also do symbolics (if one purchases the "symbolic toolbox"). A feature of MATLAB and MAPLE is that they can interface each other (if both are available on the machine). For instance, while doing work in MAPLE one can send numbers and instructions to MATLAB for processing. In fact the MATLAB symbolic toolbox is actually a scaled back form of MAPLE, but with a MATLAB syntax user interface.

MAPLE uses a basic document type called a worksheet, which is very similar to a MATHEMATICA notebook in function and form. MAPLE is probably the most complete and sophisticated package for symbolic calculations. The syntax on MAPLE follows that of C, a positive feature since C/C++ is the programming language that our students are most likely to learn. It has been my experience that MAPLE is slow on numerics and often runs out of memory during numerical calculations. I am also bothered by the inconsistency of notation that MAPLE employs for numerical calculations. Sometimes numerical commands end in "f" for floating point (`evalf()`). Sometimes they begin with "f" (`fsolve()`). Sometimes they don't have an "f", but are capitalized (`Eigenvalues()`). In addition, the "function" vs. "expression" paradigm in MAPLE is a little bizarre to me. For instance, in MAPLE when one differentiates a "function," an "expression" is returned. One must then use the command `unapply` to change from an expression back to a function. My students are always puzzled when they first encounter this peculiarity. I am still puzzled by it.

In addition to the basic program, MAPLE includes, at no extra cost, packages for specialized subjects; for instance: combinatorics, finance, geometry, abstract algebra, networks and graphs, number theory, the simplex algorithm, statistics, and a student tutorial. The "student" package is aimed at calculus learning, but my own opinion is that it is only slightly helpful. Its contents are basically just different versions of the basic calculus commands from the main program, plus the addition of a few "learning" commands such as `middlesum` which returns an expression for the Riemann sum of a function using a specified number of equal intervals and using the mid-points of the partition. In addition, a huge library of applications comes with MAPLE, and others are freely available on the web.

MATLAB stands for "Matrix Laboratory" and it specializes in numeric computation and is especially well suited for Linear Algebra. MATLAB is a programming language and programs can be stored in files with the suffix `.m` . However, MATLAB also has a command window, which is much more appropriate for use in simple calculus assignments. To use MATLAB as the sole software package for calculus, it is essential to get the symbolic toolbox (and make sure it is available in the campus-wide distribution). Without it, no symbolic calculations are possible. The student distribution of MATLAB

includes the symbolic toolbox. MATLAB has been around for many years and is used heavily in Engineering schools around the country. This can make it a good choice politically for a math department wishing to foster good relations with other departments in their university.

One might be inclined to use both MAPLE and MATLAB for the calculus sequence. While this is appealing for various reasons, it is my opinion at this time that it is better to pick only one package. From my experience, either package is more than adequate for the purposes of simple homework. Further, the interfaces and syntax used in the two programs are entirely different. To use both will unnecessarily complicate things for the students and for the faculty.

Another option that may become more and more practical is to make use of the symbolic capabilities of calculators. It is not necessary to use these instruments directly in class. Even if one does use them in class, it is quite reasonable to give the students some homework employing the calculators, and the principles presented in this book are just as applicable in that context as for computational software residing on desktop computers.

Calculators with symbolic capabilities include: TI-89, TI-92 Plus and HP 49G. The TI-92 Plus includes keyboard input, which some people view as a disadvantage because it is makes it too easy for students to store information (definitions, theorems) for tests. I do not have personal experience with any of these calculators, but my colleagues who have used the TI-89 have spoken very favorably about it. The speed of these machines is now sufficient to handle simple homework. The graphics, however, are far from the quality of a desktop computer. Given the size and power limitations this deficiency is unlikely to change significantly. I will say more about calculators in Chapter 9.

Textbooks are available that include homework problems employing any of the major software packages. Once a department has decided on a software package, it is a good idea to choose a text that employs the particular package. Even if the department decides to use the software in a way that is totally disconnected from the text, having the textbook consistent with the software choice allows students to make use of the homework on their own, or, more likely, at the behest of rebellious or zealous faculty members.

7. Making the software available

The best way to achieve widespread availability is through a campus-wide site-license. In comparison with the costs of hardware and other computer related expenses, the cost of a university site license for many of the computational software packages is absolutely trivial. If your institution does not have a campus-wide license for any computational software, I would encourage you to lobby your administration to provide one. You might need to enlist the support of other departments to get a large license. At my university the administrators did not respond to our requests until the Electrical Engineering School voiced interest.

Once a license is obtained, the software should be made available on all campus computers, not just those in math-using departments. If the dorms are equipped with computers, the software should

be made readily available there. This can take some work on someone's part, but many universities have a centralized means for distributing software. At my own institution, there is an office which handles software, but it took some work on our part to convince them to distribute MATLAB. They were afraid that our license for 500 concurrent users would not be adequate for the need and did not want to be put into the position of answering angry phone calls about MATLAB not being available. They said that students sometimes leave a popular program running all the time, so that it would be available. We argued that a math program would never be all that popular, but in the end we had to settle for a gradual, limited distribution of MATLAB and are still waiting for the student to have access to it in their rooms.

If students have their own computers, efforts should be made to make the software available to them. For instance, now at Georgia Tech students can buy MAPLE and MATLAB as part of a standard student bundle. At Northwestern, students in classes using MAPLE may download the program over the web. Often students are offered discounted student versions of software at institutions that purchase a site license.

CHAPTER 8

Other uses of Computational Software

"It is surely time to examine the computer's role in mathematical teaching and learning. The initial flush of enthusiasm over 'new technologies' is beginning to pass, and we should now begin to think carefully about the ways in which computers may help students. A dispassionate, well-informed examination of costs, benefits, and difficulties is needed" – Ed Dubinsky and Richard Noss [**DN96**]

1. Simple homework as a baseline standard

The fact that we all have different teaching philosophies and styles is an asset, not a liability. I do not propose that all mathematics professors adopt simple homework using computational software. Creative and energetic teachers will undoubtedly develop their own styles of using computational software, many of which may be more effective than the approach suggested here, at least in the hands of those particular instructors. Admittedly, I have emphasized that there is a lot to be gained by universality. If a department can agree to use one particular software package throughout the curriculum, this is ideal. However, coming to an agreement on a software package does not imply agreeing on how it should be used. Professors will demand freedom, and this is should be accommodated.

On the other hand, I would propose that every math department should adopt some baseline standards for using technology, particularly in the calculus sequence, because it is so important to so many students. As I have already discussed, it is my opinion that the inseparable interaction between mathematics and technology makes adopting some use of computational software the responsible thing to do. The fact that most mathematics professors must spend much of their time on research, service and other teaching duties dictates that the minimum standards for introducing technology not be unduly taxing. The simple homework approach I propose makes a very workable baseline standard - it is not time consuming to implement, it can be effectively used by instructors with a variety of teaching styles and levels of technical prowess, and it meaningfully introduces students to the mathematical significance of computational software.

At Ohio University we have adopted such a policy. Simple homework using MATLAB is strongly recommended for all sections of our main calculus sequence. Alternative uses of MATLAB are also encouraged as long as they are roughly at least equivalent to the simple homework. The undergraduate committee also suggests that MATLAB constitute 5%, and no more than 10% of the students grade.

A good solution to the technology question for a department would be to decide on a software package, make a simple homework approach readily available and require that everyone teaching calculus use the software in some way that is at least equivalent to the simple homework approach. (I would define equivalency as roughly covering the same commands.) This solution seizes the benefits of adoption of a single software package and of a department-wide commitment to introduce that software, without stifling instructors' different ideas about how the software can best be used. In the following sections I will discuss some other ways in which computational software is already being employed.

2. Using computational software to do sophisticated problems

It is unfortunately my impression that computational software is most often used in the context of difficult calculations. Ed Dubinsky [**Dub99**, p.208] states that one of the common uses of software is to do computations that are too difficult to do by hand. I agree with him when he says:

> Using the power of MAPLE or MATHEMATICA, students can learn to perform highly sophisticated and powerful algorithms. The idea is that making use of mathematics in this way is going to help students learn elementary versions of the mathematics involved. Perhaps, but I am not yet convinced that using a computer algebra system to do applications that involve Padé approximations or Tchebycheff polynomials is going to help the first year calculus student understand how to compute the McLaurin series expansion of $\sin x$ and the issues that arise when you reflect on what relationship that series has to the $\sin$ function.

In my experience, most students quickly lose interest in things that they feel are "over their heads." I have also observed that even the most brilliant mathematicians like to understand new math concepts by considering them in easy examples, starting with the trivial ones. I don't think we should expect our students to respond well to the reverse process.

For instance, returning to the topic of Taylor polynomials, I believe students benefit a lot from the simple exercise of using software to graph successive Taylor polynomials of $\sin x$ against the $\sin$ function itself. By doing so they can observe things like the fact that convergence is extremely slow at any significant distance from 0 (See section 10 of Appendix C). Such an observation can make Taylor's theorem much more meaningful to them.

3. Computer assignments in textbooks and supplements

Most calculus texts today have at least a supplement using computational software. If such a text is used, professors can give assignments of their choosing from the supplement. The choice of the text should be made in conjunction with the choice of software, even if the department does not decide to require the text's assignments. If a simple approach is adopted as a baseline standard, then having textbook assignments available is a great way for individual professors to either to opt out of, or supplement, the simple homework approach. If the software is being universally used in the department, an instructor of a higher level course can get away with a textbook problem with instructions like "use MATHEMATICA to solve this", even though I would question the value of doing so.

These textbooks and supplements vary significantly with regard to the degree of integration of the software into the material. In some textbooks, computer exercises are obvious add-ons, made in response to the suggestions or perhaps insistence of the publisher. Sometimes such exercises do not even specify the software, instead are just application-oriented projects with a vague instruction to solve using computational software. This type of exercise is generally best avoided. The typical combination of complex applications with lack of details about how the software is too be used leads to headaches for student and instructor alike. In contrast, some texts are written with the express purpose of making the software an integral part of the course [**DPU94**]. While such texts do provide consistent and useful treatment, they require a complete overhaul of the course. Such texts may eventually become the standard, maybe not, but they are certainly not for the "technophobe" or other casual user. As a supplement to a baseline simple approach, I would urge the adoption of a text or supplement that contains a variety of problems, from simple to complex, in order to give instructors more freedom in choosing a level of use. When the instructor does wish to assign more complicated problems from the book, simple assignments such as those suggested here can be used as "warm up" exercises to familiarize the students with the software and can introduce the important mathematical concepts to understanding what the software can and cannot do, and how it is doing it.

I would like to add a word of caution to instructors against employing computer homework out of a book, unless the software is widely available and widely used. For example if most of the calculus sections are not using the same software package, then you may be causing your students more trouble than you realize. Perhaps this will become apparent from their complaints about the software. Out of empathy for the students, I urge you to evaluate the technical investment on the part of your students

and make adjustments so that they are not wasting valuable study time on system or syntax problems that are unrelated to the goals of the course.

4. Getting assignments off the web

Many Universities, including Northwestern (`http://www.math.nwu.edu/maple/`) and Georgia Tech (`http://www.math.gatech.edu/~bourbaki`), make libraries of assignments for computational software freely available over the web. Other libraries are available at the web sites of software vendors. Some professors, or entire departments, may choose to simply adopt some of these existing assignments. The effectiveness of this approach will obviously depend on the quality of the assignments chosen. In particular, I would urge those involved to consider looking for and assigning the more simple assignments and ones that contain good mathematics. Regardless of the assignments chosen, a certain degree of caution is in order. It will be generally necessary to work through assignments to ensure that they perform as expected rather than failing due to changes in the software. Further, if homework is assigned simply by giving the students the web address, students might cut and paste their way through the assignments without meaningfully processing what they are doing. Some care is necessary to avoid this problem.

5. Simple homework vs. Notebooks/Worksheets

As I discussed in Chapter 3, my experience with prewritten worksheets at Northwestern was so painless that I was forced to consider whether this approach is preferable to simple homework. It is my conclusion that there are some features of the simple homework approach that make it more effective than assigning prewritten worksheets or notebooks. An important advantage of the simple homework format that I have mentioned in the context of group work is that students must type every command. The act of typing gives a tactile aspect which is missing from the worksheet format. In worksheet examples, the students need only point the cursor to the right place and press Enter. I fear that this is usually accomplished with almost no thought. For the actual exercises, it is often possible for students to cut and paste the commands from the example. While the necessity of typing each command may not seem all that important, it only stands to reason that the more senses are added to an exercise, the more likely that the knowledge is retained. A student who types out a command is surely more likely to remember the command than if she had merely pointed and entered. Moreover, in taking the time to type the command, it is more likely that the student will think about what she is doing.

Another disadvantage of giving prewritten worksheets or notebooks as assignments to students is that to do so requires a little more technical savvy on the part of the persons writing and distributing the worksheets or notebooks.

Finally, the worksheet and notebook examples I have seen are, though well written, too wordy for my taste and typically have a heavy focus on applications. These aspects of the worksheets distract from important mathematical/numerical issues that can be effectively explored using the technology

and from the more basic goal of giving the student some practice using the programs. This could, of course, be easily avoided by a worksheet author who follows some of the principles in this book. Indeed, while I believe the simple homework approach is the most effective, many if not most of the principles can be incorporated in a worksheet format, and I do not consider this a bad alternative. I do not ultimately think it is the best alternative and I would urge instructors to consider adopting the approach proposed here rather than a worksheet format. It is my impression that many in the mathematics community have never had occasion to consider using computational software outside the context of prewritten notebooks/worksheets (See [**Kra99**, p.22]) and this possibility should at least be considered.

6. Computational software in class

One idea is to hold an entire course in a computer lab or "technology classroom." This has been done at my University in a few classes, mostly limited to courses for mathematics education majors. Results have been mixed. Unless your institution has very many resources or very few students, this technique is not practical on a large scale. The technology classroom seems to be a popular idea among administrators, who I doubt have seriously considered either the true educational implications, or the demand on resources, both in terms of finances and personnel, that this approach entails.

If one does choose this approach it should involve a complete overhaul of the course design. Otherwise, one would encounter the comical situation described by Krantz [**Kra99**, p. 26]:

> . . . what better sense does it make to have a mathematics classroom with a computer before each student and the instructor delivering commands to the students. Many students are not conversant with the computer environment. Perhaps their hand-eye coordination is not fully up to snuff. Many will not be able to keep up. Hit the wrong key and you get a screen full of chaotic gibberish, or you lose your work, or you accidentally reboot, or you erase your boot sector.

Krantz concludes that instead, one should have computer problems in a lab environment just as Chemistry departments have always done with lab work. Uhl responds that the problem with Kranz's illustration is that it presents a teacher-centered approach that should be abandoned in favor of an approach where the technology is fully integrated and all vestiges of the lecture method are abandoned [**Uhl99**, p. 257].

Even if one accepts this argument, the task of fully integrating the software into the course as Uhl suggests remains a daunting one for all but the most technically proficient instructors. The simple approach, in contrast, can be carried out on a large scale and with much less trouble.

7. Computational software in labs

Using software in labs as an addition to lectures is in my experience more common than in-class use of computers in fully integrated courses. Uhl, however, warns that "weekly labs tacked on to an otherwise traditional class are of dubious value" [**Uhl99**, p. 258]. Krantz [**Kra99**, p. 24] and Bressoud [**Bre99**, p.180] agree that computer lab work should be tightly integrated with the content of the class. I am not so sure that they are correct in this assertion. For example, chemistry labs are not always directly related to the lectures. They are quite often oriented toward laboratory methods rather than basic chemistry. Similarly, much of what I try to teach the students through computer homework is not just a reinforcement of what they are learning in class, but new material. Often it involves to how to compute what they are learning in class. More importantly, I think, the addition of computer homework changes the rest of the course by infusing discussions of computational considerations throughout the course.

Although I disagree with the argument that computer labs are necessarily deficient if the material is not fully integrated with the rest of the course, I am skeptical about mathematics computer labs. Like integrated courses, but to a lesser degree, this approach might not be sustainable on a large scale at many institutions because of lack of lab space and equipment. Further, this option also suffers from the disadvantages of professors being with the students while they are dealing with technical problems. Most importantly, labs seem to me an unnecessary inconvenience for students and instructors since students can learn just as much from well-planned simple homework.

8. Spreadsheets

Some instructors employ spreadsheets in teaching. If you think about it, a spreadsheet is a type of computational software. A particular advantage of spreadsheets is that they are available on virtually every computer and students are already familiar with them. For instance, I have used Microsoft Excel for assignments in an introductory Statistics course. This was extremely simple to implement and the students did not have any difficulties. On the down-side, I got feedback that doing these assignments was not particularly enlightening, because I simply had them carry out some descriptive statistics on the computer. I have heard of others making better use of spreadsheets, for instance, to implement difference methods or Newton's method. With some thought, perhaps more interesting things can be done with spreadsheets and as the capabilities of spreadsheets improve, the possibility for good mathematics with them will naturally grow as well.

CHAPTER 9

Other forms of technology

> "All of the fuss about computers serves to divert attention away from the central human needs of the school system – better conditions for teachers and better teacher training." – Neal Koblitz [**Kob96**]

1. Graphing/Programmable calculators in class

Many faculty in mathematics departments around the country use graphing calculators in class. I view this as mostly good, but largely independent of the use of computational software in simple homework. From what people have explained to me, I recognize that graphing calculators can be a fine classroom teaching tool in the hands of the right person. Some claim that the calculators are an invaluable aid for visualization [**SG94**], or for Riemann sums and for demonstrations of techniques like Newton's method [**Can95**, p. 176]. Calculators are a convenient way to do things in class and they are nice because they can be carried and used anywhere. Thus, the calculator can be a fully integrated part of the student's mathematical life.

For the purposes of homework, the graphing calculators are a much more limited technology than computational software, unless the calculators are equipped with a symbolic manipulator (see section 2 of this chapter). In addition, the screens on calculators are necessarily much smaller and still of much lower quality than a computer monitor. Personally, I have poor eyesight and do not like to look at my calculator's LCD display for any purpose for more than a few minutes. I could not imagine using a calculator as a student in a class for a whole hour. Professors should try to be sensitive to this problem and not overuse the calculator.

Further, I find that using a calculator for programming is very undesirable. First of all, inputting commands from the calculator keyboard is very tedious. Indeed, one calculator enthusiast, despite his praise for the use of calculators, acknowledges that he himself did not undertake the drudgery of entering the programs, but instead left this to graduate students [**Can95**, p. 176]. The tediousness of inputing commands from the calculator keyboard is compounded by the fact that only a small portion of the program can be viewed on the calculator at one time. This also makes the logical structure of the program much harder to grasp and this seems likely to make the exercise less valuable to the student. I experienced these things very acutely in my freshman honors calculus course. After having programmed on a desktop, programming on a hand-held was frustrating. In the end the strategy I adopted was to first write programs on paper, where I could see the entirety. Then I would carefully input it

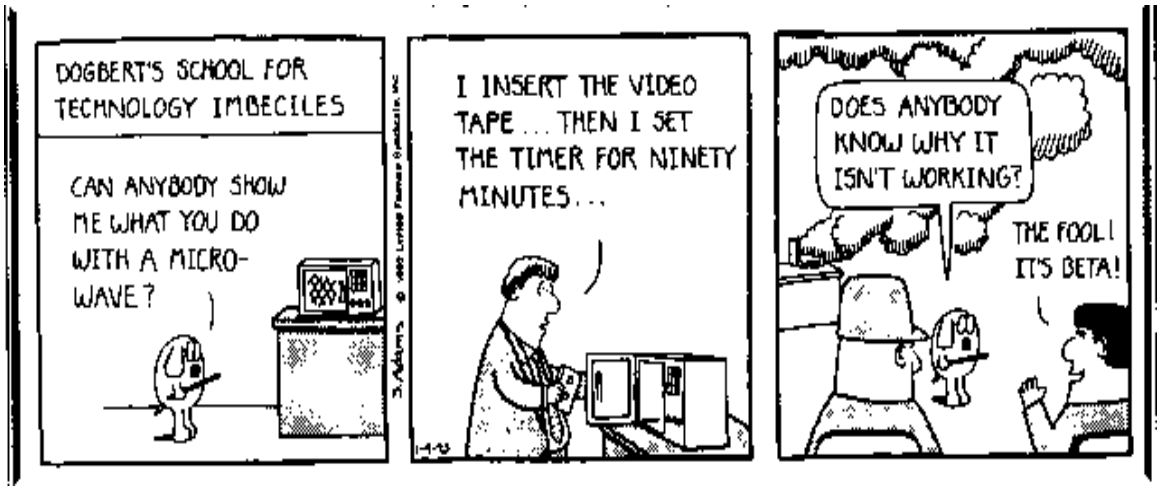
into the calculator, paying attention, not to the logically structure, but to the accurate reproduction of the symbol sequence.

Like other forms of in-class technology, in-class calculator use is best left to non-bumbling faculty - a group from which I am clearly excluded. The first time I ever picked up such a calculator I somehow managed to lock it up and the student who owned it had to have the operating system reloaded. Further, even faculty who *can* make effective use of calculators in class must consider whether the calculators' value is worth the effort needed for preparation, as well as the amount of class time and material that is displaced.

That said, for people who do want to do technology demonstrations in class, the calculator has some of advantages and is probably the best option. The primary advantage being that the students can replicate the computations themselves in real time, provided the instructions are sufficiently clear and simple. I think that there is some value derived from the students pushing the buttons and seeing the results themselves in real time. Further, using computational software directly in the classroom is typically more complicated than using calculators and may be impractical on a large scale.

Dubinsky makes the criticism that "calculators focus students on the least interesting examples" [**Dub99**, p. 207]. I am not sure I completely agree this as an argument against calculators, since the same could be said for any technology in the hands of a teacher who has not put sufficient thought into its use. In fact, the limited capabilities of the calculator, particularly its deficiency in graphing, can easily be turned into a demonstration of the need to think, rather than just push buttons.

Whether or not calculators are integrated into the course, I don't think it is wise to forbid students from using them, even on tests. In my experience, such a posture makes an instructor seem out of touch to students. To say that the calculators keep the students from learning to do calculations by hand is far too simplistic. The calculators are readily available, and they are capable of some fairly remarkable things. Some of the calculators are allowed on the SAT exams. Accordingly, in my view, it is better to teach students the mathematics needed to use the calculators properly rather than worrying about the calculators becoming a crutch. Most notably, the calculators' graphing capabilities allow exploration of mathematical ideas such as scaling, sampling, convergence and approximation. In fact, to use the graphing capabilities properly, the students must understand some of these ideas. The people I know who use the calculators in class take advantage of the opportunity to teach these important concepts. Concerns that students will be able to use calculator to succeed on examinations although they do not understand the concepts being tested should be addressed by more sophisticated testing, not by banning calculators. McCullum has an excellent discussion, including examples, of how to write tests that require more thought, less mechanical manipulation and less possibility for reliance on calculators [**McC99**].



2. Calculators with computer algebra systems

In my opinion, this technology is almost as good as computational software on a personal computer, but not quite. For homework and in the workplace, the visual advantage of a large screen, the ease of command entry and speed of a personal computer will always be preferred over calculators. It seems likely that more and more students will have such calculators in the future and that many professors will like to use them in class. All of the advantages of simple homework discussed in earlier chapters are equally applicable for such calculators. From my perspective it is not a disadvantage that the calculators cannot perform all the advanced calculations that are available in desktop computational software. What they can do is more than sufficient for simple assignments in a calculus course. Just as simple homework assignments can be used to teach students the mathematics necessary to use computational software effectively on personal computers, simple homework can be used to teach students to use the calculators intelligently, while exposing them to important mathematical ideas. It stands to reason that such use of simple homework on calculators can only enhance their usefulness in class.

3. Programming languages

It has been suggested that the most effective use of technology in teaching mathematics is to use programming to construct mathematical concepts (See [**Dub95**] [**DN96**] and [**Dub99**, p. 208]). Although this is a nice idea, I don't believe there is yet evidence to support this claim, and I am skeptical as to whether this method can be used in an easy, time-effective manner. My main area of concern about this approach is that students learning to program have to deal with a great number of syntactical and system problems. The main proponents of programming as a learning tool recognize these difficulties, but claim that problems have decreased as programming languages have evolved. They are referring to their own mathematical programming language, ISETL, however, not to commonly used languages such as C or C++.

Backing up, I did say that I think that programming is a nice idea, but not for the same reasons as some proponents. Their ideas are based on a constructivist philosophy and they believe that the computer is a good tool for allowing students to construct mathematical ideas [**Dub99**]. In contrast, I simply think that learning to program is itself valuable. In my view, the rigors of programming are a good intellectual exercise, although there is some debate about the relationship between programming and the development of other mathematical ideas [**Kob96**, p. 12] [**DN96**, p. 19]. Further, there is no doubt that programming is a marketable job skill. From my own experience, I believe requiring students to write their own programs can reinforce mathematical concepts. My first introduction to integration was in the programming course I took as a high school student and that experience made Riemann sums seem very natural to me when I later encountered them.

For these reasons, I think that every math, science and engineering undergraduate student should know how to program at least at a basic level. It is regrettable that math students are probably behind science and engineering students in this regard. This situation could be remedied if every math major were required to complete at least one course in Mathematics with a serious programming component. An obvious choice would be a numerical analysis course, but I think many of the courses we teach to undergraduates could include some programming. Number Theory, for instance, is a traditional course that is well-suited to the inclusion of programming experience. An emphasis on computing is already available in some Number Theory texts (see [**Ros93**]). In fact, a programming oriented Number Theory course can give math majors the flavor of many important emerging applications of discrete mathematics and how mathematics is carried out in real applications (for instance at world's largest employer of mathematicians, the NSA).

If one does choose to teach a mathematics course involving a programming language, I think it is best to be very conservative in what is expected of the students. Unless a specific language at a specific level is a prerequisite for the course, you should start at a ridiculously easy level. I have started by giving students a very simple running program and letting them modify it. Similarly, in [**ABD+96**, p. 16] proponents of programming say that they "begin with activities in which students try to repeat on terminal screens what is written in the text, or to predict what will be the result of running code that is given to them, or to modify code they have been given." I think that getting students to repeat, predict and modify are reasonable and worthwhile goals.

I have taught programming as part of a course in Industrial Mathematics at Ohio University using "Industrial Mathematics. A course in solving real-world problems", by Friedman and Littman [**FL94**]. The students in the course ranged from fairly weak undergraduate math education majors to advanced Ph.D. students in mathematics. Only a few of the students had any previous programming experience. The first programming assignment involved an Euler method ODE solver. I gave students a working program for a different application and all they had to do was change the equations and a few other things. As we progressed through the quarter, the students took the skeleton of this program and eventually built a finite difference solver for parabolic PDEs. Students were very positive about the

programming experience in this course. Even though most of them were not expecting to ever have to program again, and in fact probably could not write a program from scratch, they thought it helpful to know what a program is and how it works.

Although this experience was good, I would not attempt to repeat it in the calculus sequence. Despite the success in the end, the students did experience many difficulties along the way. More than once problems arose that I did not know how to resolve; we were saved only by the computer expertise of one of the graduate students in the course. Faculty considering introducing programming must remember that learning to program is a difficult, time-consuming process, fraught with many obstacles that are irrelevant to mathematics. While such difficulties may be tolerable in an upper-division course with a somewhat flexible syllabus, they pose serious problems for lower-division courses in which programming-engendered delay may detract from learning core mathematical concepts. At least in courses serving a large number of lower division non-majors, I think most of us will be much better off if we leave the teaching of programming to the Computer Science departments.

One conservative option for introducing some programming into the calculus sequence is to take advantage of the programming capabilities of computational software packages such as MAPLE and MATLAB. One can begin with simple homework, but gradually introduce some programming using the software. This can be easily achieved for instance by giving students a program and letting them vary the inputs, such as in "Monte Carlo Integration" page 104. Such an approach would be most practical in a sophomore level course at an institution where the software is used throughout the calculus sequence, and in such an environment, it is reasonable that even more advanced programming exercises be given near the end of the sequence.

4. Interactive tutoring systems

Some professors and departments may consider introducing technology in the form of interactive tutoring systems. This use of technology is somewhat different from the others I have discussed - it is not necessarily intended to illustrate computation-intensive mathematical concepts, and students are almost certainly not meant to ponder the operations of the software itself. Instead such programs are generally used simply as teaching tools replacing personal interaction between students and professors and among students themselves with interactions between student and computer. This seems to be a poor trade for students (and instructors), and I am not alone in this view. Koblitz complains, in my view accurately, that most interactive tutoring software is based on immediate gratification and does not encourage sustained mental effort [**Kob96**]. This shortcoming is only to be expected given that most software of this type is produced for profit. The main fault with these systems is that a computer cannot duplicate a humans ability to ascertain the precise nature of any lack of knowledge. As pointed out by Krantz "In a programmed learning environment . . . the student cannot ask questions. The give and take of questions and answers is a critical aspect of the human teaching process. . . . There is more to this than meets the eye [**Kra99**, p. 15]." Indeed, even if a computer can be programmed to answer some

common questions, a computer can neither make sense of a garbled question nor read the expression on the student's face. Nor can it achieve Socratic-style targeting of questions a skilled instructor can use to lead a student toward understanding.

My own department has recently begun experimenting with the use of interactive tutoring to replace lectures in remedial classes. In the past, these classes were primarily taught by graduate teaching assistants, many of whom admittedly were ill prepared for the endeavor. Introduction of the interactive technology was met with opposition by several of the faculty, but most objections have been pacified by a plan to make "help labs" with individual attention from teaching assistants a big part of the class. This experiment is still in the early stages, so I cannot report on the results, except that the professors in charge of the project have encountered many frustrating and time-consuming technical problems. In theory, supplementing interactive technology with one-to-one assistance could be a sound compromise provided that the students take advantage of the help.

5. The Web

For many years I avoided the web altogether because I saw so many people wasting so much time on it. For instance, I once shared an office with a fellow postdoc who had not yet published a single paper from his thesis, but his web page was extremely elaborate. Needless to say, despite being an excellent mathematician, popular teacher and an extremely nice person, he had a hard time finding a job in the mid '90s job market. I am completely satisfied with my decision to stay away from the game of making web pages for many years and would recommend the strategy to any young person in mathematics.

At the same time, I have become aware of some of the good things about the world wide web. For instance, I think MathSciNet is the best thing since Cauchy's Integral Formula. I am also convinced that the web page is an extremely important public relations tool. For instance, since I have had a web page, I have been contacted by several potential graduate students who were intrigued by my web page's list of my research interests. It is a safe bet that nearly all your department's prospective graduate students, as well as most potential math majors and faculty hires, will visit your department's web pages. It is extremely critical that the information presented there promotes an excellent image.

Although I am convinced of the utility of the web in some respects, I am still ambivalent about it as a teaching tool. Web-based instruction, like interactive tutoring, decreases student-instructor and student-student interaction. I have considered posting homework assignments on the web, but ultimately refrained — I would prefer that the students come to class to receive the assignments, or failing that, contact one of their classmates to get the assignments.

Even if one believes that web-based instruction provides some advantages, it may still not be worth the time investment. Learning to write interactive worksheets or to design web-based assignments etc., does not sound to me like the best use of a mathematician's time. Certainly not every professor should take the time to become a web page designer (contrary to the notions of some of our administrators).

Even typing class notes is a very time consuming process unless you are intending to write a book on the subject. Furthermore, providing your notes to students via the web is probably not as helpful to the students as you might think. They have trouble reading the book, and even though you have many brilliant ideas, in all likelihood you are not a vastly better writer than the textbook authors. We should be careful that any web-based initiatives are simple and time-efficient.

My hesitation about incorporating web usage would likely be met with great skepticism by those administrators, politicians and others who seem to think that simply using the web, for whatever purpose, is a distinct educational advantage. This indiscriminating enthusiasm for use of the web appears to be based on a misguided notion that mere familiarity with the web is in itself an important skill. I often encountered this mystical notion about needing to know computers among nonacademic acquaintances as well. But, while it is true that our students are likely to use the web a lot in their lifetimes, they can easily pick it up without artificial incorporation into the curriculum of web usage lacking any clear educational purpose. For instance, my son, now in a public school, routinely has to get at least one source from the web for his reports.

While I do advocate skepticism and selectiveness in choosing whether or how to include web usage in mathematics courses, I recognize that in some situations, using the web may offer potential advantages. For example, it might promote more "time on task," because the students are more fascinated by their web browser than by opening the textbook and doing homework problems. This effect may decrease over time, however, as students are forced to spend more and more time on the web. A recent development that deserves attention is the advent of web-based homework systems. My department is considering the CAPA system which was written and is maintained by a group at Michigan State University. Our physics department has been using this system for a few years and they report that it is stable, easy to use, and, effective. The system allows quite a bit of flexibility in the types of questions and answers that are possible and, with some thought, homework assignments can be designed that are challenging and instructive. The homework is centrally managed so that individual professors do not have to do anything to assign it. A similar commercial system "Egrade" is available from the textbook publisher John Wiley. We do not have graders in our department, despite repeated pleas to our administration, and so practically none of our calculus instructors grade homework. Adding CAPA homework thus might be a helpful strategy. In a more perfect world I would rather have graders, both for the sake of the student in the class and for the advanced undergraduates who serve as graders. However, with any care at all, web-based homework should be better than no homework.

Another argument in favor of some web based components in service courses is that it provides a math department a captive audience for advertising purposes. It is a nationwide problem that the number of mathematics majors is dropping. This should be a serious concern to anyone who cares about the future of the mathematical community (See [**Dou95**]). In my department, we have realized that the web is a very important recruiting tool for potential math majors. For instance, we have a page that promotes the idea that a major or second major in mathematics is a great way to

improve one's resumé. We suggest to the students that employers, professional schools and graduate schools are impressed by mathematics and we give some statistics justifying these claims (visit `www.math.ohiou.edu/~gotmath?`). To get students to your web pages for the purpose of advertising does not require a great deal of sophistication, only a bit of coordination. For instance, it doesn't require using an interactive homework system, but could be accomplished simply by posting class information on a web page.

To summarize, the web is an important technology and it is particularly impressive to students and administrators. The more we can use the web appropriately, the better. The mathematics community should continue to search for time efficient ways to use the web without causing unnecessary reduction in person-to-person interaction.

6. Distance learning

Web based instruction's cousin "distance learning" is becoming increasingly common. Indeed, as of 1999 there were already 762 "cyberschools" and half of all U.S. universities offered "off-site" classes [**Kle99**]. I find this development troubling. The most convincing argument I have heard in favor of distance learning is that in today's world, all education is "continuing education" and that providing distance-learning simply gives people already involved in their careers the chance to conveniently update their skills. While there is some truth in this, it is also true that institutions of higher education are so numerous and well-dispersed in this country that the vast majority of people are within easy reach of a college education. According to a commission that studied higher education in the state of Georgia, "If our goal is to make it so that everyone in the state has access to a college education without having to drive through more than two traffic lights, then we have succeeded." From my experience, commissions in most states could reach similar conclusions, provided they were bold enough to do so.

Even assuming that distance learning can be useful for continuing education, it is in my view much less appropriate for the traditional 18 to 24 year old student. There is more to education than job training. I fear that, thanks to distance learning, more and more people will obtain college degrees without ever sitting in a class or going to office hours, much less living in a dorm or taking part in college life. Such a degree is at best a very malnourished substitute for a real college experience. As Krantz notes [**Kra99**, p. 122-123], there is value in the ritual of attending class. Krantz elaborates that one of the goals of an education is to teach students the "Art of Discourse." I would add that this is an essential skill for a democratic people. I certainly plan to encourage my own children to attend an old fashioned university with a campus where they can live in the dorm, study with their classmates, experience classroom interaction, have philosophical discussions over dinner and attend football games on the weekends. Further, I think that the mathematics community should actively (but perhaps not too publicly) to discourage moves toward distance learning.

7. Simple computer homework in conjunction with other technologies

Certainly the various technologies available are not mutually exclusive. If you do choose to employ more than one technology, care should be taken to avoid overburdening the students with too many varied formats. In particular, using graphing calculators in class and computational software on computers for homework is very reasonable for those who wish to have an in-class component, but also want to expose students to the more powerful software. But, if the programming capabilities of the calculators are used as well, it may push the limits of what students can be expected to handle. Web-based homework and interactive tutoring systems in general do not require much technical knowledge on the part of students, so adding simple homework assignments using computational software will probably not be too burdensome. On the other hand, any kind of programming does require significant investment on the part of students, so adding even simple homework should be done with care.

CHAPTER 10

Conclusion: The future of the Researcher/Teacher model

> "The message to mathematicians is that, because your subject is essential to so much, you cannot afford to be anything but powerfully engaged in thinking through the future of your own institutions." – William Green, dean of the college of Arts and Sciences at the University of Rochester, quoted from [**Jac97**]

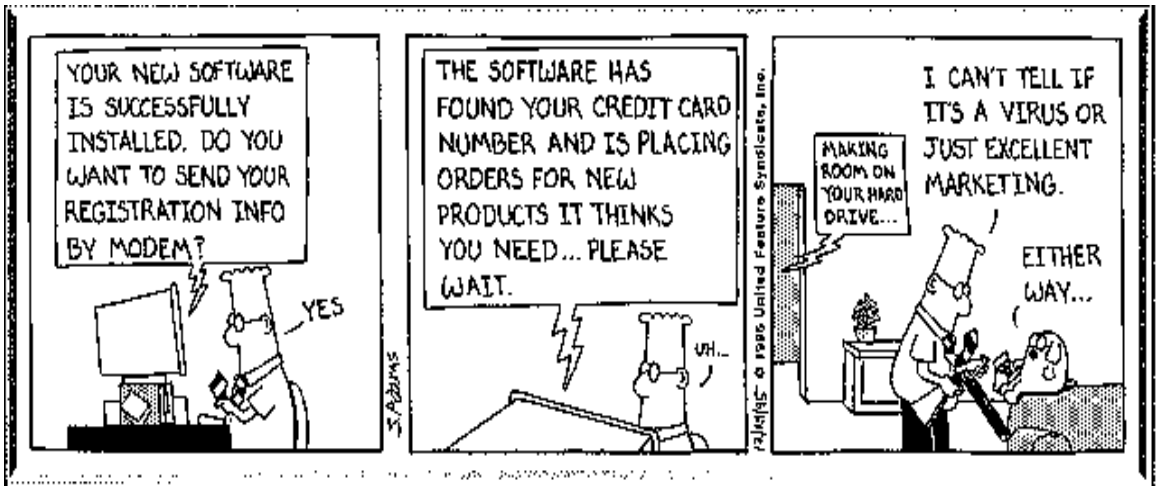
We should all pay attention to the lessons from the Rochester crisis[1]. In all likelihood, such an extreme threat to a mathematics department will not happen very often. But there is evidence that math departments face threats from various sources [**Jac96**]. For instance, at a recent trade show, the CEO of a major educational software company boasted that they would completely replace college faculty in teaching lower division mathematics courses (from [**Kra00**]). Further, the slow downsizing of departments, replacement of researchers with adjunct teaching personnel and takeover of material by other departments seem to be threats facing nearly all math departments in the United States.

In order to preempt these threats, one thing we need to do is to teach well and make it apparent to the administration that we are doing so. "Teaching is what pays the bills, and what maintains our credibility with the administration. Teaching calculus to forestry majors may not be your favorite activity, but it is important for the welfare of your department. You should do it, and you should do it well [**Kra99**, p. 109]."

The split between research faculty and teaching faculty is dangerous. Doing research should enhance one's teaching and vice-versa. That is why our system developed as it did. When the faculty becomes segregated between researchers and teachers, researchers, except for the very top level, i.e. those who bring in money, are the ones who are going to be cut when budgets are tightened. We must continue to encourage researchers to be good teachers and teachers to be good researchers.

We should also be concerned about fostering good relations with our colleagues from other departments, not only because it is the right thing to do, but also out of self interest. Cooperation with other departments has been cited as a key factor that affects the ability of math departments to maintain their

[1] In November 1995 the administration of the University of Rochester, facing financial troubles, announced the "Rochester Renaissance" which included the elimination of the graduate program in Mathematics and reduction of the Mathematics faculty by half. The goals of the plan were to improve the university's finances and to make the institution more attractive to undergraduates. It is believed that reasons the mathematics department was targeted included complaints about teaching quality and poor relations with other departments. In the end the graduate program was reinstated and the department took a smaller cut in the number of faculty. (See [**Jac97**])



size [**Dou95**]. Teaching well in service courses and paying attention to the curriculum needs of other departments are surely two ways that we can avoid making enemies among our colleagues.

Computers are at once our enemy and our ally in the fight for status in our institutions. Using technology in teaching makes us seem relevant in the eyes of administrators and colleagues from other departments. Why not take advantage of this? This may seem insincere to many, but if the heart of what you are doing when you use the technology is teaching good mathematical content, positive public relations are a welcome side effect.

Admittedly, much work remains to be done on the role of computers in mathematics education. At the some time, however, probably very few of us have the capacity to be active researchers in mathematics and at the same time active researchers in mathematics education and/or experts with computers. Experts in math education and computer enthusiasts should take care not to lose sight of this. Conversely, researchers in mathematics must recognize that because teaching is so important to our profession, we need researchers in mathematics education, and we need to listen to their results. As conclusions emerge from studies on the role of technology, they should be carefully considered by the math community.

Mathematics research is important. Teaching is important. In the end, introduction of technology in teaching must be simple enough for mathematicians to do it without decreasing their research productivity. I hope this book will persuade the mathematics community that such simplicity is worthwhile and possible.

APPENDIX A

Seven Principles for Good Practice in Undergraduate Education

The following is a summary of the Seven Principles for Good Practice in Undergraduate Education as complied in a study supported by the American Association of Higher Education, the Education Commission of the States, and The Johnson Foundation. The study appeared in the AAHE Bulletin [**CG87**].

(1) Good practice encourages student - faculty contact
(2) Good practice encourages cooperation among students
(3) Good practice encourages active learning
(4) Good practice gives prompt feedback
(5) Good practice emphasizes time on task
(6) Good practice communicates high expectations
(7) Good practice respects diverse talents and ways of learning

APPENDIX B

Reference Materials

The materials in this appendix have been used in various contexts. They serve only as example and should be adapted to the individual institution.

The key idea of these materials is to be complete but concise. Cover the needed commands and syntax without going into detailed explanation.

The first document is a page that we give to all math instructors at Ohio University.

The document "MatLab Assignments at Ohio University" is handed out to students at the beginning of each quarter.

The next two pages "A Very Brief Intro to Maple" is a the technical information sheet that I gave to students at the beginning of a course. It serves as a concise users' guide to MAPLE for the entire course. It covers the basic issues in using MAPLE, the commands that might be used in the first year of calculus, plus the most basic commands used in multi-vector calculus, differential equations and linear algebra.

Note that the page begins with very explicit instructions about how to start the MAPLE program from a standard platform (a Windows machine in the Mathematics Department at Ohio University). You will of course have to modify this for your own institution. Even if the software has been widely available for a long time in your institution, explicit instructions like these are essential for the sake of freshmen and faculty.

1. How to give MATLAB homework

Overview:

- MATLAB is now available in computer labs campus-wide.
- Simple homework assignments have been written for use in 263ABCD, 340 and 410.
- The undergraduate committee has passed a resolution strongly encouraging use of the assignments in all 263 sections. Use is to follow the guidelines in this document.
- All MATLAB homework and reference materials are available at the departments web site: `www.math.ohiou.edu/~ho`

Using the homework

- Make the homework 5% of the grade, or extra credit for the same amount.
- Hand out "MATLAB homework at Ohio University" and instruct them to download: "A Brief Intro to MATLAB" and "Defining, Evaluating, and Plotting functions" and the "Sample Solution." Instruct them to keep these as references.
- Assign about 5 assignments per quarter. Assignments can be given as hand-outs, or, you may assign them on the web page.
- Grade the assignments.
- Hand back the assignments and discuss the mathematical content.

Remarks

- Do not attempt to teach the students to use MATLAB. They pick up the basics of MATLAB directly from the assignments. As a corollary, you do not need to know MATLAB. At most, you need to be able to follow the explicit instructions of the assignments. Most student problems will result from not typing the commands exactly as directed.
- The theme of most assignments is to promote the mathematical ideas which are necessary for computing intelligently. Often this involves planting a seed of skepticism.
- Hints are placed at the bottom of each assignment. You should point this out to students and read the hint yourself before grading. The hints are meant to serve as a guide to the main mathematical ideas of each assignment.
- Since use of MATLAB is new at O.U., it is best to grade very generously. I usually give full credit as long as the student has touched on all the points in the hint, however badly.
- LaTeX versions of the assignments are available on the web page so that you can modify them.
- To learn more you are invited to read "MATLAB in Calculus and Beyond – An 1804 Fund Project" or the book *Technology in Calculus - A Simple Approach*. Both of these documents are available at: `www.math.ohiou.edu/~young` .

Steve Chapin, `chapin@math.ohiou.edu`, 3-1275.

Larry Snyder, `snyder@math.ohiou.edu`, 3-1266.

Todd Young, `young@math.ohiou.edu`, 3-1285.

2. MATLAB Assignments at Ohio University

Where to find assignments:

The O.U. MATLAB web page is at: `www.math.ohiou.edu/~matlab`. It contains assignments, reference materials and a sample solution to an actual assignment.

Where to find MATLAB at Ohio University:

Computer Services Center (ground floor) and **Alden** (2nd floor):
M - R 8am - midnight; F 8am - 10pm; Sat 10am - 10pm; Sun 12pm - 12am.
Boyd Hall 015, **Brough House 006**, **Brown Hall 000**, and **Jefferson Hall 130**:
M - R 3pm - 11pm; F 3pm - 5pm; Sat. closed; Sun. 12pm - 11pm.
Morton 314: Check lab door for available hours.
Morton 325b: M - F 8am - 5pm.
In Morton use the user name `student` and leave the password blank. In order to use the printers in Morton one must get an account from the Math Office for $5.
Stocker 264, 267: M - R 8am - 3am; F 8am - 11pm; Sat. - Sun. 12pm - 12am.
Stocker 127, 305, 308, 414: M - F 8am - 11pm; Sat. - Sun. 12pm - 12am.
To log on to Stoker NT machines, you must have a username and password. For non-EECS majors, username `student` and leave password blank.
MATLAB is available on all dorm computers through Novell.
A student version of MATLAB is available at the local bookstores or directly from the Mathworks webstore at: `www.mathworks.com`. A time-limited trial is also available on the Mathworks web site.

Getting Started:

- Download "A brief introduction to MATLAB", the sample solution, and your assignment from the O.U. MATLAB web page.
- Go to one of the labs listed above.
- Open MATLAB. All MATLAB programs can be accessed by clicking on Start → MatLab → MatLab 6.0. There should also be a MATLAB icon on the desktop in Morton.
- Work through "A Very Brief Intro to MATLAB".
- Read the "Sample Solution" to see what is expected.
- Follow the instructions on your assignment *exactly*.
- Think about what happens and take notes.
- Write a brief report, answering all the questions on the assignment. Pay attention to the comments at the bottom of the assignment.

Please return any comments to: young@math.ohiou.edu

3. A Very Brief Intro to MAPLE

How to access MAPLE from a Windows machine in the Math Department (if there is not already a MAPLE icon on the desktop, or MAPLE as a selection in the programs menu of the startup menu) :
double click: Network Neighborhood,
double click: wicke,
double click: maple v5.1,
type "`********`" as the password, ... Only necessary the first time you use MAPLE.
double click: Bin.win, and
double click: wmaple.

A window titled "`MAPLE V Release 5`" should appear. Inside the window should be a "worksheet" titled "`Untitled(1)`", which contains a prompt "`[>`." Commands are entered after the prompt.

A few general principles

- All commands must end with a semicolon "`;`" (or "`:`" if you wish to suppress the output).
- After a command is typed, it is entered by pressing the Enter key.
- To be safe, type "`restart:`" at the beginning of each new set of calculations.
- `%` indicates the output of the preceding command. (Release 4 uses `"` instead.)
- MAPLE does both symbolic and numerical calculations. Numerical commands often contain an "`f`" for floating-point, e.g. "`fsolve( )`" is a numerical version of "`solve( )`."
- An object may be an "expression" or a "function." See example below.
- When you make a mistake, you do not have to retype the whole command, simply correct the errors and re-press Enter. (Sometimes you also need to `restart;`.)
- If an operation is taking too long, you may stop it by clicking the stop icon.
- The `Help` utility is very helpful and easy to use.
- Reference manuals may be found in Morton 314 and 325b.
- Text may be added by selecting `Insert`, then `Text input`.
- Save, print and exit using the `File` menu.
- Some commands are contained in "packages", which must be included by using the command "`with( )`." A complete list of packages may be found in the `Help` utility.
- MAPLE may be used as a programming language. For a simple but instructive example see `www.math.ohiou.edu/~si`

[1] also found on page 104

Some basic commands (try them)

- `f := x -> sin(x);` .....Defines the function $f(x)$.
- `f(2);` ..... Gives a symbolic value of $f(2)$.
  `f(2.0);` .....Gives a numerical approximation.
- `diff(f(x),x);` ..... Gives the derivative of $f(x)$.
  `f1 := unapply(%,x);` .....Converts the output "expression" into a "function."
  `f1(Pi);`
- `int(f(x),x);` ..... Finds the indefinite integral.
- `int(f(x),x=0..1);` ..... Finds the definite integral.
  `evalf(%);` .....Gives a numerical approximation to the integral.
- `expr := cos(x)^5 + sin(x)^4 + 2*cos(x)^2 - 2*sin(x)^2 - cos(2*x);`
  `simplify(expr);`
- `polyn := x^5 - x^4 - 7*x^3 + x^2 + 6*x;`
  `factor(polyn):`
- `eqns := {x + 2*y = 3, y + 1/x = 1};`
  `solve(eqns,{x,y});` .....Solves the system of equations.
- `fsolve({cos(x)-x=0},x);` .....Solves numerically.
- `plot(f(x),x=-Pi..Pi);`
- `plot3d(sin(x*y),x=-2..2,y=-2..2);` Click the right mouse button, select `axes`, select `normal`. Point the cursor at the box, hold down the left button and rotate the box at will. Double click to redraw the graph.
- `with(plots);` .....Adds the "plots" package.
  `c:= sphereplot(1,theta=0..2*Pi,phi=0..Pi):`
  `d:= cylinderplot(0.5,theta=0..2*Pi,z=-2..2):`
  `display([c,d]);`
- `ode := diff(y(t),t,t) + y(t) = sin(t);`
  `dsolve({ode,y(0)=5,D(y)(0)=0},y(t));`
  `plot(rhs(%),t=0..75);`
- `with(linalg);`
  `A := matrix([[1,2],[3,4]]); B:= matrix([5,6],[7,8]);`
  `multiply(A,B);`
  `c := vector([1,2]);`
  `linsolve(A,c);` .....Solves $A\bar{x} = \bar{c}$.

4. A Very Brief Intro to MATLAB

A few general principles

- Type commands at the prompt and press Enter.
- Unless declared otherwise, variables are row vectors ($1 \times n$ arrays).
- MATLAB is case sensitive, i.e. $X \not\equiv x$.
- The command `clear` will clear all variables. Always `clear` before starting a new computation. The command `clear` will not clear the screen.
- Ending a command with a semicolon "`;`" suppresses the output.
- Enter all commands exactly as given in the assignments.
- `ans` indicates the output from the preceding command. Two useful commands are `pretty(ans)` and `simple(ans)`.
- MATLAB does both symbolic and numerical calculations. The command `syms x` declares `x` to be a symbolic variable.
- When you make a mistake, you do not have to retype the whole command. Use $\uparrow$ and $\downarrow$ to return to a line, correct the errors and re-press Enter. (Sometimes you also need to `clear`.)
- Access `Help` by clicking Help $\rightarrow$ Help Window, or by typing `helpdesk` or `helpwin`.
- Reference manuals may be found in Morton 325b.
- Text may be added to your work after the symbol %.
- Save, print and exit by clicking the File icon.
- Certain advanced procedures may be accomplished using Toolboxes. See the reference manual for details.
- MATLAB may be used as a programming language.
- For more details on MATLAB and using it we suggest the reference book:*A Guide to Matlab for Beginners and Experienced Users*, by B. Hunt, R. Lipman and J. Rosenberg, Cambridge University Press, New York, 2001.

Some basic commands using a symbolic variable (try them)**.**

- `syms x` — This makes a symbolic variable.
- `f=x*sin(x)` — This makes `f` a symbolic function.
- `f1=diff(f)` — `f1` is the derivative of `f`.
- `f2=diff(f,2)` — `f2` is the second derivative of `f`.
- `F=int(f)` — `F` is the antiderivative of `f`.
- `int(f,0,pi)` — This is a definite integral.
- `limit(log(cos(x))/x^2,0)` — MATLAB uses L'Hopital's rule to find limits.
- `limit((log(x))^2/x,inf)` — Also for ∞/∞.
- `ezplot(f)` — Plot a graph using the default interval.
- `ezplot(f,0,4*pi)` — Plot a graph for specified interval.
- `polyn = x^5 - x^4 - 7*x^3 + x^2 + 6*x`
  `factor(expr)`
  `solve(expr)` ..... This solves the equation "`expr = 0`"
- `expr = cos(x)^5 + sin(x)^4 + 2*cos(x)^2 - 2*sin(x)^2 - cos(2*x)`
  `simple(expr)`
- `ode = 'Dx = -a*x'`
  `dsolve(ode,x(0)=3)`

Some basic commands using arrays.

- `t=0:.01:8*pi;` — Makes `t` a vector with entries from 0 to 8π in .01 increments.
  `y=t.*sin(t);` — This evaluates $t\sin(t)$ for each entry of t.
  `plot(t,y)` — This plots the pairs of points $(t(k), y(k))$ for $k = 1, 2, \ldots$.
- `x = -2:.05:2; y = x;`
  `Z = sin(x'*y); mesh(Z)` — `'` means transpose.
  A figure window will appear with a graph. Click on Tools and select Rotate 3D. Point the cursor at the graph and "click and drag" to rotate the graph.
- `A = [1 2 3; 4 5 6; 7 8 10], C = [1 2; 3 4; 5 6]`
  `A*C` — multiplies the matrices.
  `b = [1 2 3]', A\b` — solves $Ax = b$ by Gaussian elimination.

5. MatLab Commands for Linear Algebra

Making vectors: Unless otherwise specified, variables are row vectors ($1 \times n$ arrays). Here are examples of ways to form vectors. Try them:

- `b = [1 2 3 4]`
- `b = b'`
- `xx = 0:.1:2`
- `yy = linspace(0,3,13)`

Making matrices:

- `A = [1 2 3; 4 5 6]`
- `C = eye(3)`
- `D = ones(4)`
- `E = zeros(5,3)`
- `F = rand(2,3)`
- `G = randn(5)`
- `H = hilb(5)`
- `P = pascal(4)`
- Commands for other specialty matrices include: gallery, hadamard, hankel, invhilb, magic, rosser, teoplitz , vnader, wilkinson.

Basic operations:

- `B = A'`
- `A*C`
- `C*A` ..... Will not work, `C` is 3 by 3 and `A` is 2 by 3.
- `x = P/b` ..... Solves $Px = b$.
- `P*x` ..... Checks the previous command.

Some specialty commands

- `[m n] = size(A)`
- `p = diag(P)`
- `diag(p)`
- `flipud(A)`
- `fliplr(A)`
- `v = randn(10,1), a = abs(v)`
- `s = sort(v), m = max(v)`
- `norm(v)`
- `norm(eye(4))`
- `N = Null(D), D*N`
- `rank(D)`
- `det(D)`
- `trace(D)`
- `inv(G), N*G, G*N`
- `cond(H)`

Some matrix decompositions:

- `[L U P] = lu(G)`
- `[V m] = eig(G)`
- `[U T] = shur(G)`
- `[Q R] = qr(G)`
- `[U S V] = svd(G)`

APPENDIX C

Some simple homework assignments using MAPLE

On the following pages are examples of simple homework assignments written for MAPLE.

At the bottom of each assignment is a brief editorial note explaining the mathematical content of the assignment.

These assignments and others are available at `www.math.ohiou.edu/~simple`. They are available directly in pdf and postscript formats. The LaTeXsource files are also available at the website for those who are interested in making improvements. All assignments at this site contain comments for students and instructors.

1. Defining, Evaluating and Plotting Functions

(1) At the prompt type: `f := x -> sin(x);` then press Enter.

(2) Type (at the prompt and then press Enter): `f(2);`
Type: `f(2.0);`
What is the difference?

(3) Type: `plot(f(x),x);`

(4) Following the example above, define and plot the function g(x) = exp(x).
Then adjust the domain in the plot by typing: `plot(g(x),x=-2..2);`

(5) Plot the functions `log(abs(x))` , `sin(x^5)` , `x^7 - x` , and `sqrt(x^2 - .00001)` , using the commands in 1. and 3..
What are the problems in these plots?

(6) How does a computer plot a function? That is, how does it decide where to put the curves? What are the potential drawbacks of this?

(7) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

In this exercise students learn basic commands for defining and plotting functions. Mathematical concepts the students must consider are the difference between numerical and symbolic evaluation of a function, the processes by which the software makes plots and issues of scale and the effects of rapid oscillation on plotting.

2. Factoring Expressions and Solving Equations

(1) At the prompt, type the following commands and press enter:
   (a) `expr1 := (x-1)*(x-2)*(x-3)*(x-4)*(x-5);`
   (b) `expr2 := expand(expr1);`
   (c) `factor(expr2);`
   (d) `solve(expr2=0,x);`
   (e) Explain what happened. What is the relationship between solving and factoring?

(2) Type and enter:
   (a) `expr3 := 4*x^4 + 3*x^3 - 2*x^2 + x + 3;`
   (b) `factor(expr3);`
   (c) `solve(expr3=3,x);`
   (d) `allvalues(%);`
   (e) `evalf(%);`
   (f) Explain what happened. Explain why an exact, symbolic solution may not be as useful as an approximation.

(3) (a) Make `expr4` be equal to `expr1 + 1` .
   (b) Try to factor `expr4` and to solve `expr4 = 0` .
   (c) Why do you think MAPLE produces a numerical solution, rather than symbolic? Hint: Is it possible in this case to give a symbolic solution? Why?

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Student learn basic algebraic manipulation commands and are led to consider the difference between numerical and symbolic solving techniques. They must confront the fact that a symbolic solution is not always possible.

3. Limits

(1) Try the following commands (at the prompt and press Enter):
   (a) `f := x -> x^2;` ..... This is MAPLE's way of defining a function.
   (b) `limit(f(x),x=2);`
   (c) `limit(f(x),x=infinity);`
   (d) `limit(1/x,x=infinity);`
   (e) `limit(log(abs(x)),x=0);`
   (f) `limit(1/x,x=0);`
   (g) Explain what happened in each example, that is, why did it give the answer it did.

(2) Use MAPLE to find the limits of the following functions at the given points:
   (a) `sqrt(x)` at `x = 0`
   (b) `sqrt(x^2 - .00001)` at `x = 0`
   (c) `sqrt(x)` at `x = -1`
   (d) `sin(x)` at `x = infinity`
   (e) `sin(1/x)` at `x = 0`
   (f) Explain what happened in each example.

(3) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Students encounter usual limits, limits at infinity and infinite limits, complex limits and oscillatory functions. They can consider why a limit would not exist.

4. Limits and Derivatives

(1) Try the following:
   (a) `f := x -> exp(sin(x));`
   (b) `m := (f(x+h)-f(x))/h;`
   (c) `limit(m,h=0);`
   (d) `f1 := unapply(%,x);` ..... (`unapply` turns an "expression" into a "function.")
   (e) `f1(Pi);`
   (f) `plot([f(x),f1(x)],x]);`
   (g) Explain in detail what happened.

(2) Repeat the procedure of 1. with: $g(x) = x^2\sqrt{x^2-1}$. Is the function defined for all real numbers? What about the derivative? Explain.

(3) Repeat the procedure of 1. with: $q(x) = (x^2+1)/(x^3-x+1)$. Are the function and it's derivative defined for all real numbers? Explain.

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

This assignment is intended to reinforce the student's understanding of the definition of the derivative. They must think about the domains of the function and its derivative.

5. Derivatives

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart;`
   (b) `f:= x -> x^2;`
   (c) `diff(f(x),x);`
   (d) `f1 := unapply(%,x);` ..... `unapply` turns an expression into a function
   (e) `plot([f(x),f1(x)],x=-3..3);`
   (f) Explain exactly what happened.

(2) Repeat the above procedure for the function

$$g(x) = \frac{x^5 + x^3 + 2}{8x + 1}.$$

Also try:
   (a) `g(1);`
   (b) `evalf(%);`
   (c) Explain exactly what happened.

(3) Adjust the domain of the plot in 2 until you can approximately guess a root of $g'(x) = 0$. Then try:
   (a) `solve(g1(x)=0,x);`
   (b) `evalf(%);`
   (c) `fsolve(g1(x)=0,x);`
   (d) Explain exactly what happened.

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Student must consider the derivative as a function, and they must consider issues of scale in plotting functions with asymptotes.

6. Indefinite Integrals

(1) Enter the following sequence of commands:
   (a) `restart;`
   (b) `int(x^2,x);`
   (c) `diff(%,x);`

   Explain exactly what happened.

(2) Repeat the above sequence for the function:

$$\frac{x}{(x-1)(x+2)(x^2-1)(x+1)}$$

   Then enter the command: `simplify(%);`

(3) Repeat the above sequence for the following functions:
   (a) $1/(1+3x+x^5)$
   (b) $\sin(x^3)$
   (c) $(1+x^6)^{3/4}$

   Why do you think MAPLE was not able to find antiderivatives of these functions? Why was it able to find the answer for the complicated rational function in 2., but could not find answers for the relatively simple ones in 3.? If you don't have any idea, ask some people.

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

In this assignment, students learn the command for indefinite integrals. They encounter the fact that MAPLE is not able to find an integral for some functions. The instructor should use this to lead into a discussion of the fundamental fact that not all functions have an integral in terms of elementary functions. The difference between this concept and the concept of integrability should also be discussed.

7. Definite Integrals and Numerical Approximations

(1) Enter the following of sequence commands:
   (a) `restart;`
   (b) `int(x^2,x=0..Pi);`
   (c) `evalf(%);`

   Explain exactly what happened.

(2) Repeat the above sequence for the function:

$$\frac{x}{(x-1)(x+2)(x^2-1)(x+1)}$$

(3) Repeat the above sequence for the following functions:
   (a) $1/(1+3x+x^5)$
   (b) $\sin(x^3)$
   (c) $(1+x^6)^{3/4}$

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

This is a simple homework assignment for integral calculus using MAPLE. The main mathematical content is that some integrals cannot be found in terms of elementary functions. Discussion then proceeds to how the computer gets the numerical answer. When students hear that the computer does a form of Riemann sum, they initially groan, but then are impressed.

8. Numerical Integration

(1) Enter the following of sequence commands:
   (a) `restart;`
   (b) `with(student);`
   (c) `f := x -> x^2;`
   (d) `leftsum(f(x), x=0..2,10);`
   (e) `evalf(%);`

   Explain exactly what happened. How close is the left sum to the actual value of the integral?

(2) Next, try:
   (a) `N := 10;`
   (b) `rightsum(f(x), x=0..2,N);` followed by `evalf(%);`
   (c) `middlesum` etc.
   (d) `trapezoid` etc.
   (e) `simpson` etc.

   What are the errors in each of the above? Why do the results from `leftsum` and `rightsum` differ as they do? Explain why it gets better as we go down the list.

(3) Repeat the above sequence but use `N := 1000`.

(4) Use `trapezoid` and `simpson` with N = 1000 on each of the following functions:
   (a) $\sin(\sin(x))$
   (b) $(1+x^6)^{3/4}$
   (c) $x^5\sin(x^6)$ with `x=0..5*Pi^(1/6)`

(5) Use the command `int` followed by `evalf(%);` on each of the functions in 4. How close were the approximations in 4 to the approximations obtained here? Was N = 1000 big enough?

(6) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Students compare different numerical schemes and think about why some methods are better than others. They consider the effect of grid size and also encounter the integral an oscillatory function.

9. Monte Carlo Integration

(1) Enter the following of sequence commands:
   (a) `restart;`
   (b) `Seed := readlib(randomize)();`
   (c) `U := evalf(rand(1000000000)/1000000000);`
   (d) `N := 1;`
   (e) `total := 0;`
   (f) `for i from 1 to N do;`
   (g) `total := total + (U())^3;`
   (h) `od;` ..... Closes the `do` loop.
   (i) `avg := total/N;`
   (j) Explain exactly what happened. The result is an approximation of $\int_0^1 x^3 dx$.

(2) Return the cursor to the top of the "worksheet" and reenter all commands. Do this 10 times, recording the result each time. Calculate the average absolute error of the ten results. Hint: Add a line to the end of your "program" which will calculate the absolute error.

(3) Change the value of N in 1(d) from 1 to 10. Then repeat 2.

(4) Continue to increase N by factors of 10 and repeat 2 until the calculations take too long (more then 1 minute each).

(5) Make a chart showing the relationship between the sample size N and the average absolute error. Try to guess a function that expresses this relationship.

(6) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Student have a lot of fun with this one. They are amazed when told that Monte Carlo Integration is actually used sometimes in practice because it is efficient in higher dimensions. This exercise also gives students a gentle introduction to programming in MAPLE.

10. Taylor Series

(1) Enter the following sequence of commands:
   (a) `restart;`
   (b) `taylor(sin(x),x=0,4);`
   (c) `poly1 := convert(%,polynom);`
   (d) `plot([sin(x),poly1],x=-Pi..Pi);`
   (e) Explain exactly what happened.
   (f) For what domain does the Taylor polynomial appear to be a good approximation of the function?

(2) (a) Change the `4` in the above commands to a `5`, and reenter all the commands. What changed?
   (b) Continue to increase the order of the approximation until it appears to be accurate on the whole interval $[-\pi, \pi]$.
   (c) For the order above, use Taylor's theorem to give an upper bound on the error of the approximation.

(3) Repeat the above process for e^x, on the interval $[-3, 3]$.

(4) Repeat the above process for $\sin(e^x)$ on the interval $[0, 3]$. What problems do you encounter. What do you think causes this?

(5) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

Students gain some appreciation for the loss of accuracy of the Taylor approximation as one leaves a neighborhood of the approximation point. Discussion in class can include the fact that computers can only do polynomial operations. So for instance, the $\sin$ function on calculators may use a Taylor polynomial and knowing the accuracy is important.

11. Defining and Plotting a Function of Two Variables

(1) Enter the following sequence of commands:
   (a) `restart;`
   (b) `f := (x,y) -> sin(x)*cos(y);`
   (c) `plot3d(f(x,y),x=0..10,y=0..10);`

(2) (a) Point at the graph, press the left mouse button and move the pointer around slowly. The graph should rotate.
   (b) Click the right mouse button, a menu should appear listing several options that change the appearance of the graph. Try them all.
   (c) Using rotations and options from the menu, make the graph as clear to you as possible.

(3) Repeat the above process for the function, $f(x, y) = x^2 - y^2$. (You will also have to adjust the domain in the plot command.)

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

The goal of this project is to familiarize the students with the higher dimensional plotting capabilities of the program and to introduce them to the notion that views and domains must be adjusted to obtain a useful picture.

12. Linear First-order Differential Equations

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart;`
   (b) `ode1 := diff(y(t),t) = -0.1*y(t);`
   (c) `dsolve({ode1,y(0)=1},y(t));`
   (d) `plot(rhs(%),t);`
   (e) Explain exactly what happened.

(2) Repeat the above procedure to solve and plot the solutions for the following differential equations. Use the same initial condition as above.
   (a) $y'(t) = \sin t$
   (b) $y'(t) = -0.1y + \sin t$
   (c) Explain exactly what happened in each example.

(3) Compare the differential equations in the three examples. Then compare the graphs of the solutions in the three examples. What do you observe from these comparisons?

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace , and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in your grade.

Students observe that for a linear differential equation qualitative features of solutions tend to ”add” as terms are added to the righthand side.

13. Linear Second-order Differential Equations

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart;`
   (b) `ode1 := diff(y(t),t,t) + y(t) = 0;`
   (c) `dsolve({ode1,y(0)=1,D(y)(0)=1},y(t));`
   (d) `plot(rhs(%),t);`
   (e) Explain exactly what happened.

(2) Repeat the above procedure to solve and plot the solutions for the following differential equations. Use the same initial condition as above.
   (a) $y''(t) + y(t) = \sin t$
   (b) $y''(t) + 0.1y' + y(t) = 0$
   (c) $y''(t) + 0.1y' + y(t) = \sin t$

(3) Compare the differential equations in the four examples. Then compare the graphs of the solutions in the examples. Based on things you have learned in class, explain the differences between the examples.

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace , and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in your grade.

Students explore the interaction of damping, restoring, and forcing effects on the solution.

14. A Spring-Mass System

(1) Type the following commands (at the prompt and then press Enter):

(a) `restart;`

(b) `M:=2; k:=5; p:=1; A:=1; a:=1;`

(c) `ode := M*diff(y(t),t,t)+p*diff(y(t),t)+k*y(t) = A*sin(a*t);`

(d) `dsolve({ode,y(0)=1,D(y)(0)=1},y(t),type = numeric};`

(e) `plot(rhs(%),x=0..50,numpoints=1000);`

(f) Explain exactly what happened.

(2) Repeat the above, modifying the values of the parameters. By looking at the solution can you find the value for `a` that gives "resonance"? Try this value to test your hypothesis.

(3) As directed by your instructor:

(a) Delete any mistakes using the mouse and Backspace, and get a printout,

(b) Prepare a brief written report answering all the questions. Writing quality will play a part in your grade.

Students consider effects of parameters on solutions and explore the concept of resonance.

15. Linear versus Nonlinear Differential Equations

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart;`
   (b) `ode1 := diff(y(t),t,t) + y(t) = 0;`
   (c) `dsolve({ode1,y(0)=1,D(y)(0)=1},y(t));`
   (d) `plot(rhs(%),t);`
   (e) Change the initial condition to $y(0) = 1$, $y'(0) = 0$. How does this affect the solution?
   (f) Explain exactly what happened.

(2) Repeat the above procedure to solve the the following differential equation (call this equation `ode2`). Use the initial conditions: $y(0) = 1$, $y'(0) = 1$.

$$y''(t) - y(t) + y^3(t) = 0$$

Why is MAPLE unable to solve this equation? Now try the following:
   (a) `with(plots);`
   (b) `sol2 := dsolve({ode2,y(0)=1,D(y)(0)=1},y(t),type=numeric);`
   (c) `odeplot(sol2,[t,y(t)],0..30,numpoints=1000);`

Try changing the initial conditions as in 1. How does this effect the solution? How does this differ from the linear case?

(3) Use the commands you learned in 2 to numerically solve and plot:

$$y''(t) - y(t) + y^3(t) = \sin t, \qquad y(0) = 1, \quad y'(0) = 1.$$

Think about how bad nonlinear equations can be compared to linear equations.

(4) As directed by your instructor either:
   (a) Delete any mistakes using the mouse and Backspace , and get a printout, or
   (b) Prepare a written report answering the questions. Writing quality will pay a part in your grade.

This assignment demonstrates that the solutions of linear equations are very “tame” compared with solutions of nonlinear equations. Almost all the time in a usual Sophomore level ODE class is spent on linear equations, and students should have some awareness of why.

16. Matrix Operations

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart: with(linalg);`
   (b) `M := matrix([[1,3,-1,6],[2,4,0,-1],[0,-2,3,-1],[-1,2,-5,1]]);`
   (c) `det(M);`
   (d) `inverse(M);`
   (e) Explain exactly what happened.

(2) Repeat the above procedure for the matrix:

$$N := \begin{bmatrix} -1 & -3 & 3 \\ 2 & -1 & 6 \\ 1 & 4 & -1 \\ 2 & -1 & 2 \end{bmatrix}.$$

Explain what happened.

(3) Multiply M and N using `multiply(M,N);` . Can the order of multiplication be switched? Why or why not?

(4) Find the determinant and inverse of the following matrix:

$$A := \begin{bmatrix} 1.2969 & .8648 \\ .2161 & .1441 \end{bmatrix}.$$

(5) Let B be the matrix obtained from A by rounding off to three decimal places. Find the determinant and inverse of B. How do A^{-1} and B^{-1} differ? Explain how this happened.

(6) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in your grade.

This exercise requires that student pay attention to the dimensions of matrices, something that students often neglect at first. It also introduces some notions of the instability of calculations. In part 4, the determinant of the matrix is near zero.

17. Solving Linear Systems

(1) Try the following commands (at the prompt and then press Enter):
   (a) `restart: with(linalg):`
   (b) `A := matrix([[1.2969,0.8648],[0.2161,0.1441]]);`
   (c) `b1 := vector([1.2969,0.2161]);`
   (d) `linsolve(A,b1);`
   (e) Repeat the process but with a vector $b2$ obtained from $b1$ by rounding off to three decimal places.
   (f) Explain exactly what happened. Why was the first answer so simple? Why do the two answers differ by so much?

(2) Try the following commands:
   (a) `f := (i,j) -> sin(i*j);`
   (b) `B := matrix(2,2,f);`
   (c) `c := vector([1,2]);`

   and use linsolve to solve $Bx = c$. Then change the 2's to 3's in the second line, change c to $[1, 2, 3]$ and try to solve again. Use `evalf(%)` to obtain a floating point value of the solution. Try the command `Bn := map(evalf,B);` and then `linsovle(Bn,c);` . When a matrix contains floating point numbers, MAPLE does floating point computations. When would an exact symbolic solution be more useful and when would a floating point solution be more useful? For big matrices, which type of computation would be faster?

(3) Input the matrix: $C := \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$ and solve $Cx = d$ with $d1 = [4, 8]$ and $d2 = [1, 1]$. Explain the results.

(4) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace , and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

The matrix in 1 is nearly singular, causing the linear system to be very sensitive to perturbations. Students are exposed to both symbolic and numerical solutions. The ideas of no solutions or infinitely many solutions are reinforced.

18. Eigenvalues and Eigenvectors

(1) Try the following commands
   (a) `restart: with(linalg):`
   (b) `A := matrix([[1,1],[0,1]]);`
   (c) `eigenvals(A);`
   (d) `eigenvects(A);`

   Find the eigenvalues and eigenvectors for this matrix by hand and interpret the output.

(2) Input the matrix

$$B := \begin{bmatrix} 3 & -1 & -1 \\ -1 & 0 & 2 \\ 1 & 1 & -3 \end{bmatrix}$$

   and try the commands:
   (a) `eigenvals(B);` followed by `evalf(%);`
   (b) `Eigenvals(B,Bvecs);` followed by `evalf(%);`
   (c) `print(Bvecs);`

(3) Create a matrix using the commands:
   (a) `f := (i,j) -> (i^2 + j^2)/(i+j);`
   (b) `C := matrix(5,5,f);`

   and repeat the process in the previous part.

(4) What are your observations about symbolic vs. numerical computations from the last two parts?

(5) As directed by your instructor:
   (a) Delete any mistakes using the mouse and Backspace, and get a printout,
   (b) Prepare a brief written report answering all the questions. Writing quality will play a part in the grade.

In the first example students must consider multiplicities. The last part should lead to a discussion of the fact that polynomials of degree 5 or higher cannot in general be solved symbolically and so exact symbolic eigenvalues cannot be found for 5 by 5 matrices. They should also notice that symbolic solutions are sometimes too complicated to be useful.

Bibliography

[ABD+96] M. Asiala, A. Brown, D. Devries, E. Dubinsky, D. Mathews, and K. Thomas. A framework fro research and curriculum development in undergraduate mathematics education. In J. Kaput, A. Schoenfeld, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 6 of *Issues in Mathematics Education*, pages 1–32, Providence, RI, 1996. A.M.S.

[And99] G. Andrews. The irrelevence of calculus reform: Ruminations of a sage-on-the-stage. *UME Trends*, 6:17–23, 1999.

[BK95] T. Berger and H. Keynes. Everybody counts / everybody else. In N. Fisher, H. Keynes, and P. Wagreich, editors, *Changing the culture: Mathematics education in the research community*, volume 5 of *Issues in Mathematics Education*, pages 89–110, Providence, RI, 1995. A.M.S.

[Bre99] D. Bressoud. Personal thoughts on mature thinking. In S. Krantz, editor, *How to Teach Mathematics*, pages 173–181. A.M.S., Providence, RI, second edition, 1999.

[Can95] R. Cannon. The road to reform. In N. Fisher, H. Keynes, and P. Wagreich, editors, *Changing the culture: Mathematics education in the research community*, volume 5 of *Issues in Mathematics Education*, pages 89–110, Providence, RI, 1995. A.M.S.

[CG87] A.W. Chickering and Z.F. Gamson. Seven principles for good practice in undergraduate education. *AAHE Bulletin*, pages 3–7, 1987.

[DGH99] G. Donovan, J. Geronimo, and D. Hardin. Orthogonal polynomials and the construction of piecewise polynomial smooth wavelets. *SIAM J. Math. Anal.*, 30:1029–1056, 1999.

[DN96] E. Dubinsky and R. Noss. Some kinds of computers for some kinds of math learning: A reply to Koblitz. *Math. Intelligencer*, 18:17–20, 1996.

[Dou95] R. Douglas. The size of a mathematics department. In N. Fisher, H. Keynes, and P. Wagreich, editors, *Changing the culture: Mathematics education in the research community*, volume 5 of *Issues in Mathematics Education*, pages 67–69, Providence, RI, 1995. A.M.S.

[DPU94] W. Davis, H. Porta, and J.J. Uhl. *Calculus&Mathematica*. Addison Wesley, Reading, 1994.

[Dub95] E. Dubinsky. Isetl: A programming language for learning mathematics. *Commun. Pure Appl. Math.*, 48:1–25, 1995.

[Dub99] E. Dubinsky. Reflections on Krantz's "How to Teach Mathematics" a different view. In S. Krantz, editor, *How to Teach Mathematics*, pages 197–213. A.M.S., Providence, RI, second edition, 1999.

[Edw98] L. Edwards. Mathematics (re)viewed through technology. *Educ. Studies Math.*, 36:291–298, 1998. Book review of *Windows on Mathematical Meanings: Learning Cultures and Computers* by R. Noss and C. Hoyles.

[FL94] Avner Friedman and Walter Littman. *Industrial Mathematics. A Course in Solving Real-World Problems*. SIAM, Philedelphia, PA, 1994.

[GH98] P. Galbraith and C. Haines. Disentangling the nexus: Attitudes to mathematics and technology in a computer learning environment. *Educ. Studies Math.*, 36:275–290, 98.

[HDP96] P. Hunting, G. Davis, and C. Pearn. Engaging whole-number knowledge for rational-number learning using computer-based tools. *J. Res. Math. Ed.*, 27:354–379, 1996.

[Hea98] J. Healy. *Failure to Connect: How Computers Affect our Children's Minds, For Better and Worse*. Simon and Schuster, 1998.

[Hei88] M. Heid. Resequenceing skills and concepts in applied calculus using the computer as a tool. *J. Res. Math. Ed.*, 19:3–25, 1988.

[HHR98] K.M. Heal, M.L. Hanson, and K.M. Richard. *Maple V Learning Guide*. Springer-verlag, New York, 1998.

[Jac96] A. Jackson. Beyond Rochester: Mathematics departments come to grips with university downsizing. *Notices A.M.S.*, 43:572–575, 1996.

[Jac97] A. Jackson. Whatever happened to Rochester? Two years later, mathematics is getting accolades. *Notices A.M.S.*, 44:1463–146, 1997.

[Jus98] W. Just. How to teach with technology and still have time to eat, sleep and enjoy the other pleasures of life. Talk given in the Ohio University Department of Mathematics Teaching Seminar, 1998.

[Kle99] D. Klein. Big business, race, and gender in mathematics reform. In S. Krantz, *How to Teach Mathematics*, pages 221–232. A.M.S., Providence, RI, second edition, 1999.

[Kob96] N. Koblitz. The case against computers in K-13 math education (kindergarten through calculus). *Math. Intelligencer*, 18:9–16, 1996.

[Kra99] S. Krantz. *How to Teach Mathematics – a Personal Perspective*. American Mathematical Society, Providence, RI, second edition, 1999.

[Kra00] S. Krantz. Imminent danger - from a distance. *Notices of the A.M.S.*, 47:533, 2000.

[KT94] J. Kaput and P. Thompson. Technology in mathematics education research: The first 25 years in the jree. *J. Res. Math. Ed.*, 25:676–684, 1994.

[Lei96] A.R. Leitze. To major or not major in mathematics? affective factors in the choice-of-major decision. In J. Kaput, A. Schoenfeld, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 6 of *Issues in Mathematics Education*, pages 83–100, Providence, RI, 1996. A.M.S.

[Lew95] W. Lewis. Educational change in a research university. In N. Fisher, H. Keynes, and P. Wagreich, editors, *Changing the culture: Mathematics education in the research community*, volume 5 of *Issues in Mathematics Education*, pages 89–110, Providence, RI, 1995. A.M.S.

[LK96] M. Linn and C. Kessel. Success in mathematics: Increasing talent and gender diversity among college majors. In J. Kaput, A. Schoenfeld, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 6 of *Issues in Mathematics Education*, pages 101–143, Providence, RI, 1996. A.M.S.

[McC99] W. McCallum. Will this be on the exam? In S. Krantz, editor, *How to Teach Mathematics*, pages 233–239. A.M.S., Providence, RI, second edition, 1999.

[Mee98] D. Meel. Honors students' calculus understandings: Comparing Calculus&Mathematica and traditional calculus students. In A. Schoenfeld, J. Kaput, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 7 of *Issues in Mathematics Education*, pages 163–215, Providence, RI, 1998. A.M.S.

[NH96] R. Noss and C. Hoyles. *Windows on Mathematical Meanings: Learning Cultures and Computers*. Derdrecht: Kluver Academic, 1996.

[O'C98] B.R. O'Callaghan. Computer intensive algebra and students' conceptual knowledge of functions. *J. Res. Math. Ed.*, 29:21–40, 1998.

[PP96] J. Pelech and J. Parker. The graphing calculator and division of fractions. *Math. Teacher*, 89:304–305, 1996.

[PT96] K. Park and K. Travers. A comparitive study of a computer-based and a standard college first-year calculus course. In J. Kaput, A. Schoenfeld, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 6 of *Issues in Mathematics Education*, pages 155–176, Providence, RI, 1996. A.M.S.

[Ros93] Kenneth Rosen. *Elementary Number Theory and its Applications*. Addison-Wesley, Reading, MA, third edition, 1993.

[SG94] Mary Margaret Shoaf-Grubbs. The effects of the graphing calculator on female students' spatial visualization skills and level-of understanding in elementary graphing and algebra concepts. In E. Dubinsky, A. Schoenfeld, and J. Kaput, editors, *Research in Collegiate Mathematics Education I*, volume 4 of *Issues in Mathematics Education*, pages 169–194, A.M.S., Providence, RI, 1994.

[Shu99] J. Shultz. Technology: Love it or hate it, you can't ignore it. Talk given in the Mathematics Department at Ohio University, 1999.

[Sma00] S. Smale. Mathemmatical problems for the next century, in *Mathematics: frontiers and prespectives*, pages 271–294, A.M.S., Providence, RI, 2000.

[TOC96] 1995 The Oberwolfach Conference, 27 November 1 December. Questions on new trends in the teaching and learning of mathematics. In J. Kaput, A. Schoenfeld, and E. Dubinsky, editors, *Research in Collegiate Mathematics Education II*, volume 6 of *Issues in Mathematics Education*, pages 215–217, Providence, RI, 1996. A.M.S.

[Uhl99] J. Uhl. Why (and how) i teach without long lectures. In S. Krantz, editor, *How to Teach Mathematics*, pages 253–259. A.M.S., Providence, RI, second edition, 1999.

[Wol96] S. Wolfram. *Mathematica Book*. Addison-Wesley, third edition, 1996.

[Wu97] H. Wu. The mathematics education reform: Why you should be worried and what you can do. *American Math. Monthly*, 104:946–954, 1997.

[Zbi98] R.M. Zbiek. Prospective teachers' use of computing tools to develop and validate functions as mathematical models. *J. Res. Math. Ed.*, 29:184–201, 1998.

[Zuc99] S. Zucker. Teaching freshmen to learn mathematics. In S. Krantz, editor, *How to Teach Mathematics*, pages 273–284. A.M.S., Providence, RI, second edition, 1999.