A Simple Approach to Technology in Calculus

Steve Chapin and Todd Young
Department of Mathematics
Ohio University
Athens, OH 45701
chapin@math.ohiou.edu
young@math.ohiou.edu

We present an approach to using computational software in calculus that is simple, both for students and faculty. The approach consists of brief assignments given as homework. These assignments emphasize mathematical principals needed to use software intelligently and contain very explicit technical instructions. Our approach, discussed in detail in [2], is designed to be a supplement to the standard calculus courses that are taught at most colleges and universities in the United States. The computer exercises, that we refer to as "simple homework," have been implemented at Ohio University using MatLab. A sample assignment is included at the end of this article.

Why Simple?
It was the experience of the authors that many uses of technology in Calculus were too burdensome, both for students and faculty. Our material was developed as an antidote to this problem, while at the same time providing students with mathematically significant experience using software.

Existing uses of technology can be divided into three categories: 1. Integrated Use, 2. Project Format, and 3. Textbook Add-ons. The first type has been promoted extensively in various forms by the Technology/Education community. While it seems that integrated use can be very successful, it is doubtful that it will ever be accepted on a large scale because of the demands made on the instructors. The project format, also promoted by parts of the Technology/Education community, is characterized by application-oriented projects that involve difficult problems. These projects are reminiscent of exercises in engineering design courses. Some math textbooks include such projects. Difficulty of the application and sparseness of actual computing time are drawbacks of this format. It is also apparent that most math instructors are not interested in the time commitment required for these projects. The third type of technology use has become quite common in textbooks. Typically these appear along with the other exercises and usually consist of the posing of a problem with instructions such as "use a computer algebra system to solve the resulting equations" (see for instance [1]). While this type of exercise may seem quite convenient to a novice instructor, it is a nightmare for the students and this difficulty invariably returns to haunt the instructor in the form of pleas for technical assistance. Finally, the main goal of the existing approaches seems to be to illustrate the material. We think this is inadequate because much more can be accomplished.

In putting together simple homework assignments, we have sought to adhere to the following principles.
– **Use of the software can and should be simple for students.**
– **Using the software can and should be easy for instructors.**
An instructor of a calculus course should be careful to not unduly burden students with technology, as has often happened in introductions of technology. It is an undeniable fact that calculus courses are very difficult for many students. Further, these courses are major career events for students interested in the sciences and engineering. To overburden students in these courses with additional difficulties, borders on irresponsibility.

Many attempts to incorporate software in math classes have required extensive time commitments on the part of instructors. This use of time is not only impractical for professors with other time demands, but it is unnecessary. A clear, simple approach to assignments makes minimal demands on professors. Also important is that professors can make good use of the software without direct in-class use. Experience has shown that in-class use of software is not a good model for many professors.

How to be simple
Steps we have taken in order to achieve the goal of simplicity include:

(1) Use simple, basic calculations,
(2) Give very clear instructions on the assignments, and
(3) Give very clear, concise technical information.

We believe that the compelling reasons in favor of incorporating computational software in calculus are mathematical rather than pedagogical. The advent of computational software represents a significant milestone in the development of mathematics. Just as proficiency with the slide rule was once an integral part of the education of any mathematician, scientist or engineer, today's students need exposure to the ideas that are necessary for the intelligent use of computational software. Specifically, our goals for the assignments have been:

- **Assignments should familiarize the student with the commands and syntax of the program.** In many previous "project-oriented" approaches to technology in calculus, the software was used only to solve a computational difficult at the end of the project. In this way students gain only minimal practice using the program. We think it is better for an assignment to contain several basic calculations.
- **Assignments should focus on the principals needed to use the software well.** Unlike the calculator, today's computational software packages perform operations for which there is no fail-safe algorithm. Students should be led to be skeptical of the computer's output and to understand the basics of the processes used in the computations. Many assignments are aimed at

training the students to understand the differences between numerical and symbolic computations and the underlying mathematics needed to use each effectively.

Uniformity

If at all possible an institution of higher learning and its Math Department should pursue a coordinated approach to computational software. They should choose a software package, make it available campus wide and use it in many classes. Widespread availability and use of the software greatly increase the efficiency of the operation, and we believe they also increase the effectiveness of student learning. Widespread use of a software package, ideally throughout the calculus sequence, convinces the students that the software is important and they pay more attention to it. They then carry some of their experience on to other classes. If a package is widely available and widely used, technical difficulties on the part of students are minimized since students may routinely find help on technical problems from classmates, neighbors and friends. The software becomes part of the culture of the institution. To implement the simple homework approach efficiently, an institution should have a campus-wide site license for the software, it should be installed on as many computers as possible and the Math Department should commit to using it.

Suggestions on use

Once the software is in place, we suggest using the homework assignments in the following way:

(1) Make the homework 5% of the grade, or extra credit for the same amount.
(2) Make sure that the students know where the software is available. At Ohio University, we give them a list of labs where the program is installed.
(3) Assign about 4 - 8 assignments per quarter, or 6 - 12 per semester.
(4) Grade the assignments.
(5) Hand back the assignments and discuss the mathematical content.

Other remarks

- Even if you wish to use computational software more extensively in your class, we think that simple assignments serve as a good starting place. For instance Young taught a numerical analysis course for Civil Engineering majors that met in a computer classroom, but used simple homework as warm up exercises. Chapin has also used the assignments in a similar way in a numerical analysis course.
- Do not attempt to teach the students to use the software. They pick up the basics of the software directly from the assignments. As a corollary, instructors do not need to know very much about the software. At most, one needs to be able to follow the explicit instructions of the assignments. If

assignments are clear enough, almost all student difficulties will be a matter of 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 in the student's thinking.
- 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.
- In order to make introduction of the software go smoothly, it is best to grade very generously. We usually give full credit as long as the student has touched on all the points in the hint.
- To learn more about the simple approach represented in these assignments you are invited to read "MATLAB in Calculus and Beyond – An 1804 Fund Project" or the book *Technology in Undergraduate Math - A Simple Approach.* Both of these documents are available at: `www.math.ohiou.edu/~young`.
- The Ohio University MATLAB homepage is: `www.math.ohiou.edu/~matlab`.

Many math professors at Universities are busily engaged in research in addition to their teaching duties and so are not interested in technological enterprises which require a lot of time. They are also not interested in changing their basic teaching style. However they do recognize computational software as an important innovation and will appreciate our emphasis on the mathematics of computing intelligently.

In short, we 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 instructor. Our simple approach 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.

References

[1] E. Johnston and J. Mathews, *Calculus.* Addison - Wesley, 2001.
[2] T. Young, *Technology in College Math – A Simple Approach.* Preprint, `www.math.ohiou.edu/~young/book/`

A Sample MatLab Assignment:

Defining, Evaluating and Plotting Functions

(1) At the prompt type: `syms x` and then press $\boxed{\text{Enter}}$.
Now type `f = sin(x)` and then press $\boxed{\text{Enter}}$.

(2) Type (at the prompt and then press $\boxed{\text{Enter}}$):

```
subs(f,2)
subs(f,'2')
double(ans)
```

Which of the above answers are numerical and which are symbolic? (You may want to type: `help subs` and `help double` for explanations)

(3) Enter: `ezplot(f)`

(4) Following the example above, define and plot the function $g(x) = \exp(x)$ by typing:

```
syms x
g = exp(x)
ezplot(g)
```

Then adjust the domain in the plot by typing: `ezplot(g,-2,2)`

(5) Enter: `ezplot(x^2)`

(6) Plot the function $\sqrt{x^2 - .00001}$ by typing: `ezplot(sqrt(x^2 - .00001))`.
Plot the function $x^7 - x$ by typing: `ezplot(x^7 - x)`.
Because of the domain chosen by the computer, important features of the graphs are missing. What are they? Try adjusting the domains until these features are shown.

(7) Plot the function $\sin(x^5)$ by typing: `ezplot(sin(x^5))`. A computer plots a function by locating a finite number of points and "connecting the dots". How does this go wrong for $\sin(x^5)$?

(8) Prepare a brief (< 2 page) written report describing what happened, answering all the questions and sketching the plots. Use complete sentences and standard mathematical notation. Do **not** get a printout.

This exercise introduces basic commands for defining and plotting functions. We consider the difference between numerical and symbolic evaluation of a function and the processes by which the software makes plots. We address issues of scale and the effects of rapid oscillation on plotting.