Computing for 3^{rd} year Geoscience Undergraduates

Dylan Mikesell

October 3, 2016

My experience teaching computation is limited (I just started my second year as an Assistant Professor), but I use computing in all of my research. I strongly believe that any student studying STEM these days should know the basics of computing and how computers work. This semester I developed an introductory computing course for geoscience students at Boise State University. My goal for this class is to introduce undergraduates to computing and basic applications in the geosciences. My hope is that this course sparks their interest in computing and they pursue other courses offered at BSU that utilize computing. I think that all modern geoscience (historically geology) departments should be producing computationally literate graduates. When I say computationally literate, I do not mean that the students need to be proficient programmers with the ability to implement finite-element solutions to PDEs or know the different computer architectures. I mean that they should know some basic characteristics of computers and software development. They should also realize the limitations of computing when it comes to solving applied physics problems.

Most, if not all, students have computers in their pockets, but after teaching this course this semester I have come to realize they have no idea how this device works. Within our undergraduate program we do not require any programming; this fact has proved to be a barrier to the level of computation I can teach. In this class I use MATLAB to introduce the students to computer programming. We are also using GIT via the Github Desktop Application for almost everything in this class, and we cover a little Markdown. Part of computer literacy, in my opinion, is the notion of software development and how collaboration works. The knowledge that software is developed iteratively and in teams is important as these students enter the workforce. We do lots of soft skills development in other classes, but I don't notice it so much in computing course. These seem very individual; therefore, I have been trying to use group programming as much as possible in order, not only to develop soft skills, but also help each other learn how to code. Everybody does it different and I think exposure to the different approaches to numerical problem solving is good for the students.

I have also been noticing that the lecture part of class is not as good as it could be. I think it has to do with their confidence. No one wants to offer an incorrect solution. We do activities individually and in groups during lecture. I find that many students often do not participate though, and I have not figured out how to teach in a computer lab and keep them from surfing the Internet.

In this course I spend a significant amount a time teaching the basics of computing and MATLAB. Each homework is designed to introduce a new geoscience application of the computing tools we cover in lecture. The homeworks are designed to introduce students to the limitations, approximations and assumptions that go into specific modeling and computation tasks. We then review these ideas in detail once the homeworks are completed. I have really been trying to teach them not to assume that their computer models are correct. For example, before they interpret their results, they have to consider the accuracy and potential sources of error in their results. They also do this in partners because I think a healthy skepticism of other peoples computational tools is good when they first start comparing solutions.

I designed this course with other professors who teach higher level courses in our department. The goal is to use this course to introduce students to MATLAB and programming so that they can spend less time on the basics in these upper division courses. I have been using existing SERC resources in order to develop the homeworks that I am giving. I find that the homeworks on the SERC MATLAB for Geoscientists page are not necessarily complete. For example, one activity uses finite differences to model hillslope evolution; however, neither the term finite difference nor the notion of computing a derivative is ever introduced. I really like this activity so I have modified it to use for the section of class where we talk about numerical derivatives.

The things I would like to take away from this workshop are strategies for lectures and activities. I feel like the homeworks are going really well and that the students are (most of the time) connecting the physics to the computing, but my lectures suck. I create these MATLAB practical files that accompany each lecture. I then go through these to demonstrate the new ideas after I first go over something on the board; for example, talk about the gradient and vectors and them implement the gradient operation in MATLAB in the practical. My plan was that the students would then use the practicals as a study tool, but I get the sense that this in not happening. So, I would be interested to hear what kind of tools people use, aside from homework, to get the students practicing with code examples that I give them.