

Teaching Statement

As a graduate student, I know many people who see teaching in competition with research for time and energy, but I find that these two academic activities have more in common than not, at their core. In fact, I see them as two sides of the same coin. An existing community of academics conducts research to discover new knowledge; newer members of a community attend universities to learn existing knowledge. This duality leads to my characterization of teaching (like research) as a simultaneously intellectual and social process. Keeping these twin aspects of teaching in mind has guided my growth as a teacher, especially insofar as I have sought to understand the interactions between these intellectual and social natures.

I suspect that most of us are aware of teaching as a social experience and find it rewarding to share our knowledge with students. We all feel the responsibility of a society to provide quality education for its citizens. I believe it's important to remember the social nature of teaching not just to motivate why we teach but also how we teach. This simple precept underlies much of the teaching advice I've received and given, such as being enthusiastic and engaging students. The bottom line always seems to be that teachers must be effective communicators and leaders. My clearest illustration of this principle came while taking an undergraduate course on abstract algebra from an ancient professor who was fluent in half a dozen languages. Ironically, this polyglot rarely understood the questions we asked, so unable was he to remember not intuitively grasping the subject matter. Since then, I have made it a top priority to understand where my students are coming from.

In practice, this insight leads me to seek connections between what I'm teaching and what motivates my students or is familiar to them. While leading discussion sections for a course on artificial intelligence, I could appeal to a natural fascination with the human mind or demonstrate how a constraint-satisfaction algorithm applies to familiar newspaper puzzles. In my most recent teaching experience, three semesters designing and giving a course on the Perl programming language, I sought to bring to life some rather dry material about syntax and data structures. I shared my favorite language features, gave advice about idioms students might encounter in their next real-world job, and emphasized the use of real data in programming assignments. Students in my course plotted climate changes from decades of local meteorological data, found the most frequently occurring words and names in classic novels, and computed the "Bacon number" of famous actors and actresses from raw data downloaded from the Internet Movie Database. Yet I have found that students sometime respond most to simple touches, such as taking the time to leave personalized feedback and advice in addition to the corrections and grade on a homework assignment. In short, for most students, learning is as much about connecting with a human teacher as it is about absorbing pure ideas.

To my surprise, teaching has also given me some of the most intellectually challenging problems I've encountered. I often don't realize how incomplete is my knowledge of a topic, until I wonder how to communicate that knowledge in class. The question often becomes how to distill a body of knowledge into a couple of key ideas, or alternatively into those seeds that will best flourish after taking root in students' minds. I particularly struggled with this question for my course on Perl. The language's own designers admit they intended it to be easy to use (for experts), not easy to learn. In class, I focused on organizing principles and examples of use, but I dedicated proportionally more of my time and energy to developing the hands-on component of the course. Compared to my predecessor's use of a few large projects, I assigned weekly programming exercises to give students as many experiences with the language in different situations as possible, with more frequent feedback. The key intellectual task became designing assignments that naturally focused on central concepts while de-emphasizing irrelevant details. In this way, teaching requires a deeper understanding of what is important than I might otherwise develop.

Of course, designing appropriate challenges for students is not a purely intellectual task. In fact, some of the most critical constraints I impose are social: the challenge should provide students a sense of accomplishment and minimize frustration and wasted effort. These considerations require an understanding of both the ideas to be learned and the students who will learn them. As someone who always learned most rapidly from text, I continually remind myself to employ diverse methods, to reflect the diversity in students. In class, I deliberately try to give a somewhat different

take than the textbook, and I give references for material not in the text. I discovered the hard way that this philosophy also applies to homework. I once designed a programming assignment in Java that I tried to make manageable by writing the “boring” framework myself and leaving only the “interesting” components for the students to write. I underestimated the students’ ability to implement this component just from the interface I described, without seeking to understand all the code. Some students may also have struggled with an interface very different from the one they would have used. While some students may have learned a valuable lesson in coping with somebody else’s code, this wasn’t the lesson I intended!

Interactions between social and intellectual aspects therefore govern my approach to teaching at every level, from designing a homework assignment to considering a curriculum. One of my favorite courses at the University of Texas is CS 302 Computer Fluency, which gives a basic understanding of how computers work to people with perhaps no background in computer science whatsoever. At first, I was aghast to learn that the professor wanted me to show non-majors how to program in assembly! I later realized that the students were not picking up a professional skill; they were gaining an appreciation for what goes on beneath the hood whenever they use computers in their daily lives. In that one course, I went from teaching assembly language to showing people how to create web pages in raw HTML. The challenge remained the same: figure out what’s important and find a way to communicate it effectively. Whether I’m showing non-majors how computers work, teaching a new language to programmers, showing students the breadth of a field such as artificial intelligence, or even helping my peers digest a particularly dense research paper in a reading group, I have found it intrinsically rewarding to accept this challenge.