Teaching Statement

Samuel Deng
samdeng@cs.columbia.edu
samuel-deng.github.io

Developing a cohesive teaching philosophy requires learned experience. Over the past six years, I've had the immense fortune to reflect on what it means to be a teacher through:

- Creating and teaching a brand-new course, **Math for Machine Learning**, under a school-wide Teaching Fellowship, while driving its adoption as an official undergraduate offering.
- Immersing myself in the broader teaching community by publishing and sharing Math for ML as a **poster** with other educators at SIGCSE 2025.
- Co-creating and co-teaching a brand-new introductory Python companion course, **Natural and Artificial Neural Networks Lab**, for students with diverse academic backgrounds.
- Serving as Head TA for Machine Learning, Computational Linear Algebra, and Discrete Math, leading teams of TAs in classes as small as 15 students to multiple sections with upwards of 300 total students, and designing and teaching semester-long weekly **recitations**.
- Engaging in research-backed pedagogical training through Columbia's Center for Teaching and Learning's Teaching Development Advanced Track Program.

I've found that three guiding principles are central to my personal philosophy of teaching.

A driving and cohesive narrative should propel all parts of a course. From a single lecture to an overall syllabus, I've found that developing as a *storyteller* is integral to developing as a teacher. My individual lectures tend to have this structure: present a simple example that still contains all the intricacies of the underlying idea (*introducing the world*), move to clearly delineating each component needed to understand the example (*setting up the characters*), explaining each component in turn (*plot development*), and then, most importantly, showing how each component was absolutely necessary for understanding the example and the broader idea (*climax and resolution:* any good story should abide by Chekhov's gun!). For example, my **lecture** on the linear algebraic proof of least squares in Math for ML starts with an **interactive 3D rendering** of the theorem statement, moves to carefully defining projection, rank, inverses, and columnspace, and then comes back around to fitting each ingredient into the whole. On the level of the course overall, I attempt to make sure that each lecture is in service of a key narrative. In Math for ML, **the overarching story** centers around least squares regression and gradient descent, the *what* and the *how* of machine learning. I reinforce this narrative by constantly looping back to **"big picture" slides** during each lecture: I gradually add more and more components to a 3D rendering of least squares and gradient descent that develops **week** to **week**. My style of "teaching as storytelling" has been well-received:

> *"It really felt like the instructor was prepared to teach the class - the contents were not only organized but it had story to it. It worked up its way to a bigger concept. He had amazing slides and each concepts were supported with examples that he clearly worked through in class." (Anonymous student review, Math for ML Summer 2024)*

Ideas should be presented as if the student could've discovered them themselves. I try to embody a sense of discovery in my teaching by creating as many opportunities and resources as possible for students to gradually discover the kernel of ideas themselves. For every topic in Math for ML, every lesson included hand-made **interactive 3D renderings** that allowed students to literally see an idea from as many different perspectives as needed. These are continuously woven into the lectures as the main driving examples in the slides, and I develop them bit-by-bit as students learn the necessary pieces to "uncover" another piece of the pictures. This philosophy also pervades how I design problem sets and programming assignments. In Computational Linear Algebra, a particularly well-received **programming assignment** I designed had students gradually scaffold very basic functions implementing change of bases until, at the end, they discovered how to piece them together to a nontrivial application: flattening digital images. In Math for ML, my **problem sets** aren't just sequences of exercises: they model the process of mathematical discovery by giving a nontrivial result or theorem as a problem, but then guiding students through the process of: (i) testing it on simple examples (ii) proving key lemmas (iii) piecing the lemmas together to complete the theorem. This is exactly how I discover and prove the unknown as a researcher, and I've found that this instills students with the confidence that they understand exactly how a theorem could have been proven by *them*. I'm thrilled that students notice this teaching principle as well:

> *"Samuel is really good at teaching. Not only does he have the knowledge base, but he also has a very good energy about him while he's teaching that draws you into the material. Also, he can dumb things down 'simple stupid' which make it easier to broadly grasp a concept before building upon its intricacies that make it complex." (Anonymous student review, Computational Linear Algebra Fall 2022)*

An instructor should never forget how they first struggled when learning the same ideas. As an undergraduate, I was a first-generation, low-income student, and, as such, I've always felt a little behind, whether on prerequisites, knowledge, or even how to carry myself in college. This **motivated** my development of Math for ML. I sincerely believe that, if I could make the jump from naïve undergraduate to machine learning researcher, anyone can, and this conviction is at the deepest core of every part of my teaching. On the behind-the-scenes side, the first question I ask to myself before generating any kind of course material is: "What did I find difficult about this when I first learned it?" This informs everything from my lecture slides to my homework problems to even the diction I use to present a concept. On the student-facing side, I strongly believe that no question is too trivial to delve into and no confusion is "unworthy" of clearing up. On day one of Math for ML, I made sure to emphasize my **rationale** for creating the course to let students know that I was "once in their shoes," providing students with an anonymous feedback form from the first day to encourage a two-way conversation in my teaching. During lectures, I've developed a real-time "red light, yellow light, green light" system open on my PC and in sight at all times, in which students could anonymously and synchronously vote on the pace of the lecture at anytime. This system comes from the understanding that, oftentimes, students may be insecure or shy about expressing confusion. Outside of class, I make sure to prioritize office hours, doubling up on denser, conceptually difficult weeks. Thus, the bedrock to my teaching philosophy is the understanding that a successful student just needs preparation—not innate talent—and the humility that, once, I did not have the same preparation I have today.