TEACHING STATEMENT

MICHAEL BERNSTEIN

How do we ground students not just in the practice of human-computer interaction (HCI), but also in the theories that animate the field? HCI teaches an interdisciplinary design praxis, but rarely grants students the tools to adapt that praxis, refashion it, or question why it works the way that it does. When there's a vast library of theories, visions, architectures, and critiques, not every design idea has to start from the drawing board. So, I developed a series of HCI courses that weave major HCI theories into modern topics, ranging from an advanced HCI Foundations & Frontiers course to a broadly accessible Social Computing course. Complementing this effort, I work to expand student access to research opportunities: I developed an undergraduate course that now engages over 75 Computer Science students per year in mentored research. I was named a Bass University Fellow at Stanford University, an honor that "recognizes faculty for extraordinary contributions to undergraduate education." I aim to live up to that honor.

HCI FOUNDATIONS & FRONTIERS: CS 347. In leading a redesign of Stanford's HCI curriculum, I focused in on a gap: our graduates lacked strong HCI theory foundations, often viewing HCI as nothing more than a user-centered process to follow. The result of this redesign was a new required course, HCI Foundations & Frontiers, aimed to teach major HCI theories and sharper conceptual thinking amongst our students. The basic concept is, for each lecture, to teach a major theory in HCI (a "foundation") and use that theory to explore, explain, and question the modern problems facing computing and society or HCI today (the "frontiers"). For example, my first lecture introduces Mark Weiser's concept of ubiquitous computing—a theory proposing that computing technology ought to become as invisible as infrastructure—and anchors it in demonstrations of engaging recent results in activity recognition, reactive environments, and programmable objects. The year that this redesign took place, student response was strongly positive.

SOCIAL COMPUTING: CS 278 (SOCIOLOGY 174/274). Today we interact with our friends and enemies, our team partners and romantic partners, and our organizations and societies, all through computational systems. How do we design these social computing systems effectively and responsibly? I designed this course as a deep dive into design patterns for social media and related platforms. On the surface, CS 278 is the kind of project-based design course that is familiar to Computer Science students. However, under the hood, it is a vehicle for teaching major social theories: Goffman's presentation of self in everyday life; Granovetter's strong and weak ties; norms both descriptive and injunctive (e.g, Cialdini); path dependence in cultural spread (e.g., Salganik, Dodds, and Watts); social influence (e.g., Asch's conformity experiments, social proof); and peer production (e.g., Benkler). In each lecture, I pose a design question, teach the relevant social theory, then weave the theory back into design principles that answer the original question. The course has grown popular, perhaps as a side effect of the visibility that it gets each year when I challenge students in their first assignment to "go viral" so they can learn just how difficult it is to design social behaviors.

COMPUTER SCIENCE RESEARCH: CS 197. When I took over the Computer Science department's undergraduate research program (CURIS), I surveyed students and was struck that many felt they simply did not have an entry point for getting involved in research. For some, competition was too fierce: the Computer Science major has grown far faster than our faculty, and many groups were maxed out in mentorship capacity. For other students, they were not equipped with the skills to engage in advanced research until late in their academic careers. So, I created a new course as a clear entry point accessible to more junior students: to do research, just enroll. Not every Computer Science research project is appropriate for more junior students, of course, but the course serves as an existence proof that a subset of projects can be designed to be a good fit. I recruit PhD students from the department to CA the course, then work with them to identify projects that are achievable for groups of younger students within ten weeks. I now teach the course each Fall quarter to train the graduate student CAs, then the graduate students continue to teach it each Winter and Spring, including a parallel version (CS 197C) designed for incoming summer CURIS interns. Together, these offerings expose over 75 students to mentored research experiences each year, and several teams each year go on to publish their work.

EDUCATIONAL LEADERSHIP. I will serve as Interim Director of the Symbolic Systems Program for 2024-2025. I also currently serve on the Vice Provost for Undergraduate Education's Undergraduate Advisory Council, and as the Akiko Yamazaki and Jerry Yang University Fellow in Undergraduate Education. I continue to lead initiatives in our human-computer interaction curriculum, ranging from requirements, to course design, to our Ph.D. qualifying exam, to our HCI Seminar. Finally, along with Professor Ramesh Johari, I initiated the Carta course exploration platform at Stanford, which lives on as an ASSU-sponsored student club. Over 90% of Stanford students use Carta for course planning.