Teaching Statement

Sam Lau (lau@ucsd.edu)
October 2022

I love teaching because of the way empathy and creativity in the classroom can directly impact students' lives. Even as class sizes at universities grow to meet demand, I aim to maintain a personal touch through teaching practices that encourage students to engage with the materials and each other. Becoming a teaching faculty has been my long-term career goal—over the past nine years, as an undergraduate, masters, and PhD student, I've taught across 17 academic terms, including 14 as a teaching assistant (TA) and 3 as the instructor of record. My teaching experience spans computer science, software engineering, data science, and human-computer interaction courses ranging from 12 to 500 students. My dedication to teaching is recognized by my students (teaching evals $\mu = 4.5 / 5$) and my colleagues (two department teaching awards). This statement describes my teaching experiences, approach to creating inclusive learning environments, curriculum development, and novel software tools I developed for teaching.

	Course	Students
UC Berkeley	CS 61AS	90
	CS 61AS	90
	CS 169	240
	Data 8	100
	Data 8	250
	Data 8	500
	Data 100	100
	Data 100	260
	Data 8	70
	Data 100	110
UC San Diego	COGS 10	260
	COGS 108	430
	COGS 108	440
	COGS 108	440
	COGS 124	12
	COGS 18	350
	DSC 10	35

Figure 1: I've served as a TA and instructor for a variety of courses across campuses, departments, and class sizes. Each entry denotes one course. Courses I taught as a TA are unbolded; courses as the instructor of record are bolded. Course = official course number. Students = number of students enrolled in the course.

Experiences from Nine Years of Teaching

My teaching experiences are summarized in Figure 1. I started teaching as an undergraduate, when I was hired as a teaching assistant for UC Berkeley's CS61AS, an introduction to programming. After CS61AS, I also served as a TA for CS169, Berkeley's software engineering course.

My enthusiasm for teaching gave me the opportunity to join the teams that developed Data 8 and Data 100, two of Berkeley's flagship data science courses that would eventually become core classes for the newly developed Data Science major. In addition to teaching my sections, I created the first versions of several class projects, including a comparison of water usage across California districts and a sentiment analysis of Twitter posts. These courses were also the first time I encountered the problem of scale. For example, during my time in Data 8, the course doubled in size each time it was offered, from 100 students in 2015 to 500 students in 2016. The computing infrastructure couldn't keep up with the growth—at several points, students submitted so many error reports that opening the list of reports would crash my web browser! To address this, I helped design the first versions of DataHub, a scalable infrastructure for running student code that became widely adopted by programming and data science courses at UC Berkeley and UC San Diego. For my contributions to Data 8, I was selected to be the instructor of record in 2017, the first time the course was taught by a student.

In addition to the usual duties as a TA in Berkeley's Data 100, I also focused on curriculum development. One challenge of teaching is that different people have different ideas about what we should teach. Data science as an undergraduate major is relatively new and doesn't yet have an established curriculum across institutions. During my time serving Data 100, I've had many pedagogy discussions with instructors from both computer science and statistics departments. These discussions motivated me to help iterate on our lectures and assignments across the course, including lessons on sampling, algorithms, and statistical modeling. This work culminated in a SIGCSE paper that, to my knowledge, is the first paper to discuss why computer science and statistics instructors take different approaches to data science pedagogy[1].

[1] Sam Lau, Deborah Nolan, Joseph E. Gonzalez, and Philip J. Guo. How Computer Science and Statistics Instructors Approach Data Science Pedagogy Differently: Three Case Studies. In *Proceedings of the 53rd ACM Technical Symposium on Computing Science Education*, Providence, RI, USA, March 2022. Association for Computing Machinery

As a PhD student at UC San Diego (UCSD), I continued to seek teaching opportunities. I served as a TA for my department's introductory programming course (COGS 18), data science course (COGS 108), and human-computer interaction course (COGS 124). In COGS 108, I overhauled the section materials, replacing lengthy lecture-style content with short, practical demos and time for pair programming. These changes were received well by students and have become adopted by the course. In 2019, I was invited back to UC Berkeley as the instructor for Data 100. And in 2022, I was selected to be the instructor for UCSD's introductory data science course (DSC 10), where I introduced a novel tool for program visualization (Pandas Tutor) discussed later in this statement. In parallel with my PhD research work, I also authored a textbook called *Learning Data Science*, discussed later in this statement.

Making Classrooms Personal and Inclusive

At large public universities like UC San Diego and UC Berkeley, programming and data science courses regularly enroll hundreds of students per offering. Because these classes are large, students can easily feel lost in the crowd. To address this, I make every effort to create an upbeat, personal, and inclusive learning environment whenever I interact with students. For example, every time I teach, I personally reach out to struggling students midway through the course to let them know that I and my course staff want to help them succeed. These students often mention that I'm the first instructor who has ever personally reached out to them during their time in college.

One of the first things I noticed was that first-generation and minority students tend to struggle in silence—one of my students was hesitant to ask for help even though he just got evicted from his home. Because of these experiences, I address first-generation and minority students during a few lectures each term. I explain that I frequently talk with their classmates about hurdles that come up outside the classroom, and that I want to help everyone succeed in the course in the context of their academic goals. In my experience, underrepresented student groups also tend to have less background knowledge coming into the course compared with their peers. Because of this, I frequently encourage my students to avoid comparing themselves against other students in the class. Spending more time to complete course assignments isn't because of innate ability; more likely, it's because some students have extra experience beyond the course's prerequisites. Even though it may take time to remedy existing social inequalities, I want to work towards making my classrooms and curricula inclusive of all of my students.

Developing an Open-Source Textbook for Teaching Programming and Data Science

During my PhD, I helped write an open-source, online textbook called *Learning Data Science*[2]. Originally developed as course notes for Data 100 at Berkeley, the book quickly grew in popularity—as of this writing, the book is currently used by 2,000 Berkeley students per year with an additional 20,000 readers from 30 countries. The book's broad appeal caused it to be recognized by O'Reilly Media, which will publish the book in hardcopy in 2023. As part of my commitment to broadening access, the book will remain open-source and freely available to read online, like all of my past lecture slides and other teaching materials.

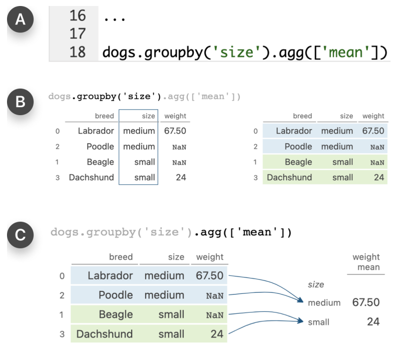
[2] Sam Lau, Deborah Nolan, and Joseph Gonzalez. *Learning Data Science*. O'Reilly Media, Inc., 2023

Learning Data Science is, to our knowledge, the first book to teach the entire data science lifecycle, from acquiring data to exploratory data analysis to statistical modeling. I and my co-authors come from both computer science and statistics, and we integrate both perspectives throughout the book. I'm especially excited about the book's case studies, which apply data science principles to calibrate air quality sensors across the US, model donkey weights for Kenyan veterinarians, estimate bus arrival times, and detect fake news.

Interactive Visualization Tools for Teaching

I'm particularly interested in designing interactive visualization tools to help instructors teach. Instructors often draw diagrams to help explain what they mean. For example, I've spent many hours drawing boxes and arrows on my lecture slides to explain how the group-by operation works on a data table. Unfortunately, this approach is time-consuming for the instructor, and students might not know how to draw correct diagrams when working independently. To address this need, I developed Pandas Tutor: a system that automatically creates explanatory visualizations of `pandas` code. `pandas` is the most widely used Python package for data table manipulation and is taught in introductory data science classes. When a user pastes their `pandas` code into Pandas Tutor, the system uses a combination of static and runtime analysis to draw a diagram for each step of a `pandas` expression, illustrated in Figure 2.

I piloted Pandas Tutor in Summer 2022 when I taught DSC 10, UC San Diego's introductory data science course. I integrated Pandas Tutor into Jupyter notebooks, which allowed me to add Pandas Tutor diagrams with a single additional line of code. During lecture, I used Pandas Tutor to draw diagrams for every data table operation I taught, including filtering, grouping, and merging. Student feedback was very positive—during lecture, I observed students tweaking code to see how the resulting diagram changed. Students also used Pandas Tutor to understand and debug their code when working on assignments. Because of this feedback, the other instructors for DSC 10 were enthusiastic about including Pandas Tutor in the course and plan to use it themselves in future offerings.



```
...

dogs.groupby('size').agg(['mean'])
```

```
dogs.groupby('size').agg(['mean'])
```

	breed	size	weight
0	Labrador	medium	67.50
2	Poodle	medium	NaN
1	Beagle	small	NaN
3	Dachshund	small	24

	breed	size	weight
0	Labrador	medium	67.50
2	Poodle	medium	NaN
1	Beagle	small	NaN
3	Dachshund	small	24

```
dogs.groupby('size').agg(['mean'])
```

	breed	size	weight
0	Labrador	medium	67.50
2	Poodle	medium	NaN
1	Beagle	small	NaN
3	Dachshund	small	24

size	weight mean
medium	67.50
small	24

Figure 2: Pandas Tutor automatically generates explanatory diagrams for `pandas` code. Users can freely input `pandas` code, including expressions that use method chaining. Pandas Tutor draws a diagram for each method call in the expression. For example: when a user inputs code that uses `groupby()` followed by `agg()` (A), Pandas Tutor will draw a diagram explaining both the `groupby()` step (B) and the `agg()` step (C).

Conclusion

I'm excited to teach because I can see potential impact not just for my classroom but also in classrooms across the world: by improving infrastructure for scale, designing teaching materials, and creating new visualization tools. As a teaching faculty, I hope to be a positive role model, colleague, and collaborator.