# Teaching Statement

John V. Monaco

Email: contact@vmonaco.com

Website: www.vmonaco.com

January 15, 2018

## 1 Philosophy

I have come to realize, and appreciate, that there is a fundamental interplay between teaching and research which can amplify the productivity of both in a kind of positive feedback loop. Successfully explaining a concept, which appears novel to the student, requires the innate ability of the instructor to decompose an idea into its primitive components, analogize to concepts the student has already mastered, and recast the problem if necessary. As these concepts provide the foundation for computer science, the process of distillation forces the instructor to keep their tools for research sharpened. Consequently, I have found that the mastery of basic computing principles taught to students, such as function growth, computer architecture, and software patterns, can greatly speed up the innovation and discovery process which is so vital to research. Conversely, research can breathe new life into even the most basic concepts that are taught. This is important in education so as to demonstrate the relevance of old ideas to new problems. For example, my work in modeling user behavior has shaped the way I present exploratory data analysis techniques, often using an example or sample data from my research for motivation.

## 2 Pedagogy

I am fortunate to have been taught by several wholly inspirational and passionate professors. As a result, my goal as an educator is to impart future students with this same inspiration to learn. This process is more of an art than a science and depends largely on the particular student. However, as I continually strive to improve through self reflection [2], I have found the following pedagogical patterns to enable student success.

*Appeal to student interests.* If at all possible, I will tailor a specific project or assignment based on the interests of the student. For many students, relevancy to personal goals becomes a strong source of motivation in attempting to master the material and transition their newly acquired skills outside the classroom. As Teaching Assistant for

*Computing Projects*, a graduate-level course that integrates software engineering and data analysis, I had the freedom to specify project requirements based on the interests and capabilities of the students. For students that expressed an interest in machine learning, I would specify a project that emphasized the scientific method and introduced them to the latest tools for data analysis that I use myself (e.g., the Python stack, numpy+pandas+scikit-learn). For students that expressed an interest in engineering and systems design, I would specify a project that involved a software deliverable, such as a web interface for biometric data collection. All students were required to write a whitepaper and present their results throughout the course, and I found that appealing to their interests within the scope of the course objectives led to an overall greater chance of success.

*Foster team building and collaboration.* Shaping a curriculum around team exercises and collaborative projects can be extremely rewarding for students, especially in the development of skills beyond computing. Besides equipping the student with soft skills which complement their core competencies, engagement with and between students is more authentic in a team environment. I find that this effect is generally amplified when some aspect of competition is introduced. For example, when teaching introductory cryptography to middle and high school students, I organized team-based cryptanalysis competitions. As the problems increased in difficulty, from simple monoalphabetic ciphers to polyalphabetic and transposition ciphers, the student teams were forced to apportion the workload amongst each other. In this way, my students often learned just as much from each other by sharing their own strategies as they did from my lecturing.

*Empower growth from student to collaborator.* I have had the pleasure of working with many talented students, ranging from STEM outreach at the middle and high school levels to graduate-level teaching. For many students, education is a kind of transition period wherein they prepare for their desired role post-graduation, whether it be industry, academia, or government. Providing the student with the opportunity to be seen as a collaborator, and not an apprentice, can help build the independence and confidence needed to successfully make this transition. At the graduate level, this may involve seeking input on some of my own work, and if a student expresses a particular interest, proposing a joint publication. This can also serve to recruit talent for my own research agenda which itself is often shaped by student interests and capabilities. As an example, I recently published a joint patent with an undergraduate student who proposed his own ideas based on interest in my research [1].

*Demonstrate my own passion.* Finally, I have discovered that one of the best ways to inspire has been to share my own personal experiences. When relevant, I will share with students my open source software projects, favorite books and authors, and introduce them to my own personal sources of admiration in computer science. For example, as Teaching Assistant to *Pattern Recognition and Machine Learning*, I would introduce techniques using one of my own research problems as motivation. One exercise I designed required the students to record the confidence scores from a commercial biometric authentication platform and calculate error rates (type I and type II) at different threshold settings. This exercise served to introduce binary classification and receiver operating characteristic (ROC) curve analysis.

This list is by no means complete, and I see these patterns as complementary to, and not to be used in place of, traditional pedagogical patterns, such as well-defined course objectives, a grading reward system, and periodic tests of aptitude. The tools to deliver the above patterns also vary widely; for example in a collaborative environment, a grading scheme can be designed to emphasize the complete deliverable of an entire team while also taking into account individual contributions. Also notably missing from the above list is an emphasis on effective writing, which I consider to be one of, if not the most, important skill at the graduate level. In many fields, it is difficult for one's work to achieve high impact without clearly communicating the process and results. This is a skill that I continue to improve myself and try to bestow upon my students, especially those performing research. As Teaching Assistant for *Computing Projects*, I have emphasized effective writing by holding periodic paper reviews throughout the course, much like an external peer review, so as to provide students with an opportunity to incrementally improve their work instead of waiting until the end for feedback. This also helps me gain a better understanding of the student's capabilities and thought process throughout the course.

With my background in computer science and experience educating a diverse body of students, I am comfortable advising graduate-level research and teaching both undergraduate and graduate-level courses on machine learning, algorithms and data structures, introductory programming, and software engineering.

## References


[1] Jordan A. Berger and John V. Monaco. Universal keyboard. US Patent No. 9,864,516. Filed on 27 July 2015. Published on 9 Jan 2018.

[2] Charles C. Tappert, Andreea Cotoranu, and John V. Monaco. A real-world-projects capstone course in computing: A 15-year experience. In *Proc. EDSIG Conference on Information Systems and Computing Education (EDSIGCON)*, 2015.