**Helping students tinker with the black box**

The increasingly quantitative and interdisciplinary nature of biological research has led us to add a BioMath Bootcamp summer course for graduate students in neuroscience, psychology and molecular biology. Students come to this mandatory but optional four-week course from all sorts of science backgrounds but typically have not had any formal math or computer science training at the college level. The idea of computational science is quite overwhelming for the students; they think it may be important but claim to be “bad at math” or “not be any good at this.” There are three overlapping goals to the course.

The first goal is a pedagogical: teach the mathematical basis which underlies the tools that they use. This is accomplished through interactive lectures each morning. During these lectures the instructors demonstrate code and give the background necessary to understand the code. We are not always as rigorous in our definitions and proofs as we could be, but the goal is to introduce the topics and mathematical vocabulary necessary to understand what is being computed behind the scenes.

The second is a practical: give the students the skills they need to run computational tools. In the first few days, we make sure that everyone has a working version of MATLAB and understands things like the path and file management. These are tools necessary to functionally use the black boxes of quantitative tools that exist. This goal is primarily met through afternoon problem sets where students work in small groups to develop code based on the topic of the day. The instructors for the course help troubleshoot computers, look over code, figure out how things are broken or just rephrase our written questions using different words. Over time, the students develop the vocabulary that they need to think and talk about computational problems, even if they can’t answer them themselves.

The third goal is provocative: show off the wow factor of where computational biology is going. We’re in an era when computational tools are enabling all sorts of new experiments that were never before possible. The instructors bring in examples from current literature or their own research to show the relevance and excitement related to the topic of the day. This third goal may make the largest impact on changing the way the students think about computation; we want them to leave the course thinking of ways that computational tools will be helpful to their own research.

There is no assessment in the course, and so students get out of it what they put in. Some students are satisfied with the cooking show approach that they get in the morning lecture. They happily watch someone else type out some code and make a plot or two but never write their own code. Every once in a while, there’s a student who loves programming and never knew it. But those aren’t our main focus. Our main focus is the students who try to do all the problem sets but it is a battle for them. They struggle through and come out at the end with the ability to be users of quantitative tools. In the end, those are the kind of scientists we want to train; scientists who are willing to work hard at a problem and stick with it until it is solved. They might not be able to troubleshoot an error, or design a new algorithm, but they’ve practiced and can ask intelligent questions to their collaborators who are designing new tools. These are the students that use all the tools necessary to answer important questions in science, even if those tools are black boxes of code that require a little tinkering to be functional.