I personally used Matlab during 28 years of physics research before beginning a teaching career, and I believe thoroughly in empowering students with computational skills and tools via Matlab.

I also presently teach at an institution where the course loads are very high, leaving woefully limited time for activities such as curriculum development and grading. And yet I am extremely particular about the content of the assessments I give students, and I’m passionate about giving students good feedback.

And so when I first began teaching, I found myself spending endless hours creating most of my own assessments (homework assignments, quizzes, and exams) by hand, and grading them tediously. One morning I woke up and realized I could automate most of the process of creating the assessments by writing some Matlab scripts, as much of the work of creating assessments in my field involved simply coming up with new input numbers each time and computing answers, and also choosing randomly between various potential questions. Within a few days I had a working prototype, and both my life and the quality of my assessments improved greatly.

The scripts evolved gradually, and I was, for example, able to incorporate symbols and formulas because Matlab supports latex based print commands. I also incorporated graphics, and ultimately many other features.

Eventually the scripts evolved into a comprehensive program for creating assessments, complete with a user-friendly GUI. At this point I had a system in which I could (and can) create and manage numerous “module files” containing the question content, and then quickly transform these into new assessments that drew on arbitrary combinations of content from the module files. It was a great relief to be able to produce fresh new assessments in seconds that contained exactly the content I wanted.

At this stage I was still printing out and distributing pdfs of my assessments, however, which is paper intensive, and still spending enormous amounts of time grading. All the more time in fact because I was now creating very comprehensive assessments with which students can really build up their knowledge continuously and thoroughly.

So one morning I woke up again and realized I could also make the distribution, performing, retrieving, and grading of assessments electronic. Within a relatively short period of time I had a working prototype, and this developed into a system I now call IKAATS, or “Interactive Knowledge Assessment and Teaching System.”

I found immediately that this system not only saved me time, but the students really liked the approach as well for a number of reasons. Primary among these is that each and every question in the electronic version has its own “Calculation Box,” in which students can create and execute a short Matlab script to solve that question.

A key aspect of these Calculation Boxes is they are fully retained: This provides a permanent record for the students of how they performed the computations. And because these are fully embedded in the assessments, there are no vexing issues with things like naming scripts, file management, etc, that would occur if I simply had students writing regular Matlab scripts (m-files). And finally, a given Calculation Box can be “held” over to be used on subsequent questions as well, for example for a set of interrelated questions.

There are many other nice features of the system as well for students: Students can enter tentative answers and see at a glance which are still tentative (so they can go back and fix them easily later). And when they get their assignments back, they can see at a glance which are incorrect, find out the correct answers, and read feedback from me.

Grading of multiple choice questions is now also fully automated, which saves me enormous amounts of time, although I can add comments to the students containing hints or further elaboration, or comments on their scripts.

I’ve now used the full IKAATS system for a few years, with great success. It has worked well, and especially for my flipped courses for reasons I’ll elaborate on briefly below.

As I’ve used the system so some deeper themes about its pedagogical aspects have emerged, which I find also relate to questions that people typically ask me about the system:

First, I quickly realized that, and have learned to emphasize to both students and other professors, the Calculation Boxes in which students do their computations in IKAATS are no substitute for doing whatever problem set-up activities are needed, such as drawing diagrams, developing the right system of equations to use, and performing any needed algebraic manipulations. So I require students to keep a separate and well organized notebook for these sorts of activities, and to perform these activities with pencil and paper.

Secondly, at the same time, I’ve also learned that the Calculation Boxes are much more than a substitute for a hand held calculator. Indeed, writing scripts causes students to focus on the theory involved, that is, the formulas, and NOT the numbers, and also on the order of the computations. So the Calculation Box activity actually reinforces the other theoretical work, instead of bogging students down with repeatedly punching numbers into a calculator.

Thirdly, requiring scripts for each and every problem that involves computation ingrains in students the essentials of computational coding, such as choosing good names for variables, assigning the input numbers to variables first and then performing whatever additional computations are needed, and choosing the most efficient choice of Matlab functions possible.

Fourth, it is nontrivial to get students to begin writing computational scripts instead of using their calculator. Although some students of mine still doggedly stick to their calculators, I’ve managed to get most not to do so, and embrace the computational coding, by doing three things: 1) The writing of scripts starts early on in the course; 2) Coding examples are included in certain early questions to help get them started; 3) I introduce them to the concept from the get-go, stress why its valuable and makes life easier for them, and then I consistently emphasize it and largely require it as things progress.

Fifth, when I first began deploying the electronic assessments I still had a long-standing bias against using multiple choice questions, because I felt that this undermined the rigorousness of the learning experience. And so I initially included lots of fill-in-the-blank type questions, which also required a great deal of my time to grade. I eventually learned, however, that the multiple choice approach actually works quite a bit better for student learning in my courses for the following reason, especially in the context of writing computational scripts: When students know that one of the possible answers presented is the correct one, AND WHEN THEY CAN QUICKLY REVISE AND RE-RUN A SCRIPT for that question, then I find that, pretty much as a rule, they will work ardently on each and every question, all semester long, to produce answers that match one of the choices. Against the prevailing wisdom in some quarters, I’ve also learned not to include very many wrong-answer choices that common mistakes tend to produce. So in almost all cases, a student will know when they’ve finally performed the calculation correctly, as it will be very unlikely that they will accidently produce a number matching one of the incorrect choices. This system provides instant feedback (negative and positive), and in my view, builds confidence as well as skill, the former being especially crucial for weaker students. And if their comprehension is lacking, they and I both find that out quickly, so that we can start addressing those gaps early on.

Over the past few years I have been flipping most of my basic physics courses, which I’ve done by producing my own teaching videos (which is a whole other story involving a special approach I developed create an adequate looking presentation of formulas and calculations and tables of contents in the videos). IKAATS has turned out to be especially helpful for flipped classes because it provides students with a highly engaging and structured environment for their in-class work. My students also utilize a laptop cart to enable students to better collaborate with one another (some do, some don’t).

There is much more to say about how IKAATS is structured and how it works technically. For example, electronic gradebooks are included, and there are many special utilities included in IKAATS for managing all aspects of the system. Those interested in it can learn more at [www.ikaats.com](http://www.ikaats.com). The system is still prototypical, and is not presently for sale, but I am happy at this time to let others experiment with it free of charge, with the only requirement that someone who does will not distribute it to others without permission, and will also provide feedback to me about their experiences. And if I ever do market it, I will not withdraw usage rights from those already using it: Such users may continue in perpetuity if they desire.

The basic technical requirements for using IKAATS as just an assessment generator is that you have access to a computer with a regular basic Matlab license. One can produce pdfs of assessments this way and distribute to students in hardcopy. To utilize the full capabilities of IKAATS as a learning management system, one needs an environment where students have adequate access to computers with Matlab and which are also networked with a “shared drive” on which a common “repository” can reside for the distribution, collection, and returning of assessments. IKAATS contains tools for easily setting up and updating such repositories.