AC 2012-3680: LEARNING MATLAB IN THE INVERTED CLASSROOM

Dr. Robert Talbert, Grand Valley State University

Robert Talbert is Associate Professor of mathematics at Grand Valley State University. Formerly, he was Associate Professor of mathematics and computing science at Franklin College, where he was also the Director of that school's 3+2 engineering program with Purdue University. His scholarly interests include cryptography, computer science, and educational technology with a special emphasis on using technology to support active learning environments in the university classroom. He holds a Ph.D. in Mathematics from Vanderbilt University.

Learning MATLAB in the Inverted Classroom

Introduction: The traditional classroom

University courses have retained the same basic structure for hundreds of years. That structure follows an easily identifiable work flow:

1. Students come to a class meeting during which a lecture is given. Students take notes and occasionally ask questions.
2. Following the lecture, learners are assigned work to be completed outside of class. This usually takes the form of homework, test preparation, or writing papers.
3. The outside-of-class work is submitted or assessed in class. The cycle then repeats.

Learning theorists would note that parts 1 and 2 correspond roughly to two stages of learning. The first stage, known as *transmission*, involves learners acquiring new information and placing it into their conscious minds. The second stage is known as *assimilation*. During assimilation, learners take what they have acquired during transmission and assimilate it into their pre-existing cognitive structures for making sense of information. We generally do not consider a student to have learned a concept until the student can demonstrate successful assimilation of that concept through tasks that demand more than basic recall of information.

In the traditional classroom structure described above, class meetings are devoted to transmission, and assimilation is delegated to outside-of-class work such as homework. Assimilation targets the most complex cognitive tasks, as measured by rubrics such as Bloom's Taxonomy[2], associated with a given concept. Therefore, this is the point where students would benefit most from the presence and coaching of an expert learner, namely the instructor. But since the tasks are done outside of class, this is precisely the moment when instructor guidance is least readily available.

Conversely, transmission -- hearing and taking notes -- is far simpler than assimilation and therefore less needful of help from an expert, but it is the time when the instructor is most fully available. Furthermore, the “live” element of a lecture makes even transmission difficult for some learners. Many would benefit from the ability to pause, rewind, and replay lectures to view the information again. It is also the case that the length of a lecture often exceeds the attention span of the audience many times over, and if a lecture must be 50 minutes long or longer, the ability to watch only parts of it at a time would be helpful.

In computing instruction, the issues described above are particularly acute. The amount of factual information in an introductory programming course is low relative to introductory courses in other disciplines, and the amount of assimilation of those basic ideas that is required is relatively high. It is difficult to learn to program by merely watching a lecture on how to program. The difficulty lies in implementing the concepts of programming, and students in the traditional classroom are largely left to figure this out on their own.

The inverted classroom

The STEM disciplines include notable exceptions to the traditional classroom model. Laboratory components to courses typically expect students to complete preparatory readings and exercises before lab, and then the lab time is spent assimilating what they have read through hands-on activities in the presence of a guide. Courses designed using project- or problem-based learning[1] extend this methodology sometimes to an entire course. STEM courses designed along these lines show evidence of being highly effective in preparing learners, particularly future engineers, for the realities of modern professional practice[8].

Both of these examples include the same two stages of learning, transmission and assimilation, that are present in the traditional classroom, but these stages take place in different contexts. Transmission is completed outside of the class meeting, and the meeting -- using time freed up by the removal of transmission -- is spent by having students work on tasks aimed at assimilation of what they saw or heard during transmission. We will refer to this classroom setup as the *inverted classroom*, since it inverts the allocation of time and space for learning tasks.

The inverted classroom is increasingly finding purchase in K-12 and higher education. For example, at Miami University of Ohio, a large-scale move toward the inverted classroom is underway in computer science, with significant qualitative and quantitative improvements in the first wave of implementation in a software engineering course[5]. Miami also uses the inverted classroom structure in economics courses with similarly positive results[7]. The University of California at Irvine changed its introductory biology course to an inverted model and saw significant gains in results of an objective test on biological concepts when compared to students in a traditional setup of the same course[9]. There is also significant interest in the inverted classroom among the K-12 community, where it is often referred to as the “flipped” classroom[11].

Modern implementations of the inverted classroom benefit from a recent burgeoning of inexpensive, accessible technology that make it easier than ever to reallocate transmission time to outside of class. In particular, the profusion of software for creating, hosting, and sharing video online makes it possible even for technological novices to put lecture materials on the Web. Students can then pause, rewind, and replay presentations and view them on devices and in doses that best suit their schedules and learning needs. The class meeting itself can then be entirely refocused on the most difficult learning tasks, with the instructor present as a guide.

MATLAB in the inverted classroom

We focus now on a particular instance of the inverted classroom in the instruction of an introductory MATLAB course for freshmen, designed and taught by the author. The institution at which this course was taught is a liberal arts college of around 1000 students with a 3+2 engineering program affiliated with a nearby university. The course, titled "Computer Tools for

Problem Solving", was designed to fit into the 3+2 curriculum specifically so that it would satisfy credit for an introductory MATLAB course taught at the partner university.

Beginning MATLAB courses like these are standard fare in most engineering programs. The Computer Tools for Problem Solving course, on the other hand, had several features that set it apart from the usual introductory course:

1. The audience for the course was broader than the usual MATLAB course. Computer Tools for Problem Solving was a required course for all students intending on taking Calculus III. This includes not only students in the 3+2 engineering program but also students with majors in Mathematics, Mathematics Education, and Elementary Education with a mathematics endorsement, as well as some students getting a minor in Mathematics. Most of these students have no experience with programming coming into the course, and many had almost no experience with using computers in any capacity in prior coursework.

2. Being taught at a liberal arts college, the objectives of the course extended to general intellectual skills and not just MATLAB programming. For example, the ability to acquire new information on one's own in response to a problem or personal need was considered a core skill (as it was in all courses at the college).

3. Due to constraints in the curriculum on staffing and budget, the course, when created, was only approved for one credit hour. This meant that the course met only once per week for 75 minutes.

The traditional classroom model, and traditional introductory MATLAB textbooks, were a poor fit for Computer Tools for Problem Solving for the reasons just listed. The breadth of the audience, particularly the wide variety of backgrounds in computing and science, was the wrong one for most introductory textbooks on MATLAB, which draw most of their examples and exercises from chemistry, physics, and engineering problems and assume a certain level of familiarity with computing not shared by many of the students. Many pre-existing materials focus only on content mastery and do note share the course's dual focus on content and process skills. Finally, the 75-minute, once-per-week schedule made a lecture-oriented approach disadvantageous for students, particularly those with no programming experience who need as much practice as they can get.

The decision was made to design the course using the inverted classroom model after considering all these features and constraints, with the additional reasoning that the inverted approach, which places the responsibility for acquiring information on the student, suited the liberal arts philosophy of the institution better than the traditional model.

Instructional design of the inverted MATLAB course: General principles

In designing the Computer Tools for Problem Solving course, three main instructional objectives were central:

1. Students should be able to write functional blocks of code in a scientific programming environment to accomplish specific tasks related to a problem at hand.
2. Students should be fluent in finding information they need about the language to accomplish tasks. This includes looking up new functions or features and reviewing functionality they have forgotten.
3. Students should apply sound, creative problem solving techniques, such as Polya's Four Stages[10], to problems in a variety of domains.

No prior computing experience was expected, and the only prerequisite was a grade of C- or above in first-semester Calculus. The course was made a prerequisite for Calculus III and for Linear Algebra. The latter courses were redesigned to involve MATLAB programming and visualization tasks as core learning objectives.

The student demographic of the course consisted of a wide variety of computing backgrounds, with most students having no programming experience at all, and a background in the sciences that was less deep than the usual science or engineering student. To make the course more accessible to this audience, the course was designed around an early focus on visualization. Programming was deferred until the fifth week of the course, following a four-week intensive treatment of plotting and data visualization. The idea was to get students to an acceptable level of comfort with MATLAB, its basic internal tools (such as the M-file), and computing in general before diving into the complexities of programming.

The content domains used for examples, exercises, and other class work were largely selected from outside the natural sciences and engineering. For example, rather than use data from a structural engineering experiment, we used data from the Statistical Abstract of the United States on tuition costs in the US, or historical data on the cost of a gallon of gas since 1975. The broader content focus in the class was intended to make the course more meaningful for all students, not just those with interests or background in the STEM disciplines.

We identified six main areas in which students will eventually demonstrate expertise in the course:

- Computation: Using MATLAB to perform complicated calculations quickly and with accuracy.
- Visualization: Taking data and functions and turning them into meaningful graphs or other visual forms.
- Data Analysis: Taking real-world data from experiments, surveys, repositories, etc. and getting usable information from the data using statistical tools.
- Programming: Extending MATLAB’s built-in features by writing programs to automate different kinds of tasks.
- Symbolic Math: Using MATLAB’s Symbolic Math Toolbox at the MuPad notebook interface to work with symbolic mathematical expressions and equations.

- Publishing: Converting programs, computation, and visualizations from MATLAB into publication-quality materials.

All elements of the course were to be aligned with a combination of the three main instructional objectives, instantiated on one or more of the six areas of expertise.

Instructional design of the pre-class experience

To support the posting and downloading of key documents for the course and to facilitate communication between the instructor and students during the six-day period between class meetings, a blog was set up for the class. The most recent version of that blog can be found at http://cmp150fc.wordpress.com. The Wordpress.com platform was selected because it is free and supports both LaTeX typesetting of mathematics and syntax-highlighted posting of source code.

While several standard introductory MATLAB textbooks were considered for the course, none was a true match for the purpose and audience of the course. A lecture-based approach was also a poor fit. Therefore, series of screencasts was created to replace both a printed text and the in-class lectures. In the first offering of the course, students used videos available at the website of Mathworks, Inc. While the videos are of a high quality, they are not targeted towards users with no prior scientific computing or programming experience, and students struggled to make sense of them. Eventually these videos were supplemented with screencasts made by the instructor. In the second offering of the course, all of the screencasts were made by the instructor using professional screencasting tools and posted to a YouTube playlist for easy access. The playlist eventually housed 41 different videos, each focused on a single subject and kept to an average length of 6--8 minutes. That playlist is located at http://www.youtube.com/playlist?list=PL60D54836FB8893F0.

The intent, following the standard arrangement for the inverted classroom, was for students to view the videos outside of class in place of the standard live lecture and then to use the information they gained on assimilation-oriented tasks in the class meeting. In order to facilitate successful transfer of knowledge from the transmission phase to the in-class activities, each group of videos was given in the context of a structured assignments known as *Guided Practice*. Every week, following that week's meeting, a new post was made to the course blog with the Guided Practice for the next week's meeting. Each post consisted of five parts:

1. An overview of the upcoming class.
2. A list of video and print resources to digest before the upcoming class. This list would include links to screencasts, rarely more than 50 minutes' worth of viewing in total, along with supplementary print resources found on the web.
3. A detailed list of competencies that each student was expected to have mastered before coming to class. These were phrased in simple, atomistic terms and using action verbs, such as “State the syntax for the first line of a MATLAB function'” and “Use the FPRINTF command to display formatted output including the contents of variables”.

4. A list of three exercises to help students map the basic information from the video and print resources onto tasks given in the competency lists. While varying in style and complexity, the exercises are intended to train students to handle new information the way expert learners do: By instantiating what they view or read in the context of simple problems that mimic examples from the screencasts. Some of these exercises involved writing an English explanation for a piece of code or a passage from a video or reading assignment. Some involved writing very short snippets of code to replicate something that appeared in a video. To encourage timely submission of the exercises, students were required to complete one of the exercises (their choice) within three days of assignment and then the remaining two prior to class time.
5. A list of specifications detailing exactly what to submit, how to format it, and a deadline for submission.

The purpose of Guided Practice was to structure the students' out-of-class experience. While the transmission phase of learning is simple relative to the assimilation phase, students still need help in navigating transmission well and making sure that information gleaned from lectures or reading finds its way into their minds successfully. This is the case in the traditional classroom as well. The Guided Practice focuses students on the upcoming tasks, provides links to high-quality learning resources, delineates clearly what they need to get from those resources, and provides them with simple but meaningful exercises to help instantiate basic concepts as a first step toward the assimilation that will take place in class.

Instructional design of the in-class experience

Class time was designed around the notion that this time was to be spent on assimilation-oriented tasks. The class meetings were segmented into three sections.

1. A five-minute multiple choice quiz given at the beginning of the meeting to focus on the essential ideas from the list of competencies. The quiz was mainly intended to provide incentive for students to do the pre-class assignments, but it also provided opportunities for the instructor to catch any serious misconceptions before the day's work began. The quizzes were taken via classroom response systems (“clickers”) so that students could get immediate feedback on their performances and the instructor immediate data on student knowledge.
2. A 5-10 minute period of questions and answers over the pre-class assignments or the quiz. If a substantial number of students missed a quiz question, that question would receive attention automatically. Otherwise, students submitted questions on paper at the beginning of class, and frequently asked questions would get first priority. (Students were also able to ask questions about the pre-class assignment in office hours during the week.)
3. The majority of time in the class -- about 60 minutes each week -- was spent on a lab assignment given at the class meeting and worked in groups of 2 or 3. Each week had a specific set of instructional objectives to address, and these were spelled out in the Guided Practice. The lab problems consisted of a small number of problems that were realistic in nature, difficult, and involved successful mapping of the basic ideas from the pre-class

assignment onto a new problem. Students were encouraged to complete the lab in class but were given until 11:00 PM of the following day to submit their final product.

The lab problem sets were graded on a 50-point scale. Ten of those points, assigned on the basis of completeness and effort, were allotted to a rough draft of the problem set that was to be turned in by the end of the class session. The remaining 40 points were broken up into four 10-point criteria: *correctness* (the extent to which the students' product produced correct results), *specifications* (the extent to which the students' product addressed the correct problem, regardless of correctness), *readability* (the extent to which the students' MATLAB code was organized and easy to follow), and *efficiency* (the extent to which the students' product accomplished the appointed task without extraneous effort). The rubric for grading lab problems sets is posted at the course blog linked above.

To complement this ongoing cycle of pre-class and in-class work, students also completed a project in which they chose a topic not covered in the main course and presented it (via an M-file published as a PDF or HTML document) to the class during the last two weeks of the semester. There was also a final exam in which students chose tasks from five of the six main areas of competency listed above (all except Publishing, which was assessed through the semester project) and completed those tasks over a two-hour period. The final exam from Spring 2010 semester can also be found at the course blog.

Example of a typical inverted MATLAB class

What follows is a sample of a typical week's work in Computer Tools for Problem Solving. The first three weeks of the course focused on basic MATLAB usage and data structures, particularly the concept of an array and the M-file, and on two-dimensional function plotting. In week 4, we shifted from plotting mathematical functions to plotting data and on performing basic data analysis using MATLAB. This corresponds to the Data Analysis competency area designed for the course.

The Guided Practice for this class can be found at http://bit.ly/x4Z2Gm. The following Overview was given in the Guided Practice:

> *This next week we will continue with plotting (for one more week), but we will shift gears from plotting functions to visualizing data. MATLAB is especially adept at handling real-world data of all sorts and in all kinds of ways. We'll be seeing how to get data from a spreadsheet into MATLAB; how to create scatterplots of data; two different ways of finding lines and curves of best fit to data (similar to the trendline features in Excel); and how to calculate basic summary statistics on data.*

Following the Overview, students were given links to six videos covering using MATLAB with popular spreadsheet software, array manipulation, basic data plotting, the Basic Fitting Tool, polynomial regression, and basic statistics in MATLAB. The videos consumed less than 44

minutes of viewing time, which is fairly close to the length of a single lecture in a standard 50-minute course. However, the online presence of the videos gave students much more flexibility in how they chose to watch than a typical live lecture.

Following the list of videos, a list of Competencies was given, ideally for students to read concurrently with the video-watching so they would have a sense of the "take-away" points from the viewing. Since this meeting introduced several new concepts and tools, the list of Competencies was quite long, with 17 distinct competencies to have mastered before arriving at the class meeting. Those competencies included:

- Import a collection of data from a spreadsheet into a MATLAB variable via the cut-and-paste method.
- Extract a single row or single column from an array using the “slicing” or “colon” syntax (for example, A(:,2) returns the second column of A).
- Use the reshape command to alter the number of rows and columns in an array; in particular, use reshape to turn a 2-dimensional array into a vector.
- Create a scatterplot of a collection of paired data and format the plot using markers and (if appropriate) lines connecting the markers.
- Find the value of a polynomial at a given input or at a vector of inputs using the polyval command.

Students understood that the quiz for the class meeting was to be taken directly from the list of competencies from the Guided Practice.

Following the Competencies list, students were given three exercises. One exercise had students create the array

```
A = randi([-10,10], [10,12])
```

Students were to save this as a .MAT file and then create an M-file that would access different locations in the array and return the values there.

The second two exercises were related. Students were given the following xy-data:

x	y
1.0	2.6
2.3	2.8
3.1	3.1
4.8	4.7
5.6	5.1
6.3	5.3

In one exercise, students were to create a basic scatterplot of the data, connect the points, and label the axes, and then compute the mean, median, and mode of the data, all in a single M-file.

In the other, students were to write a separate M-file that would find the line of best fit through the data and then plot it on top of the scatterplot from the previous exercises.

In other words, the exercises are simple instantiation tasks in which students take what they have viewed in the videos and apply it to simple, closely-related tasks. The exercises constitute "low-hanging fruit" for students to provide them not only with first steps toward assimilation of the information but also early success in acquiring the concepts. These are both critical ingredients of student engagement.

Upon arriving to class, students were given a five-question, five-minute quiz over the Competency list. A sample question from the quiz is:

> *Suppose B is an array in MATLAB with 9 columns and 4 rows. To get MATLAB to return the third column of B, type:*
>
> *(A) B(3)*
> *(B) B(4,3)*
> *(C) B(1-4, 3)*
> *(D) B(:, 3)*
> *(E) B(3, :)*

It should be noted that students had access to their computers and MATLAB during the quizzes. Therefore it is theoretically possible for students to simply make a quick example on MATLAB to determine the right answer to the question. However, this consumes a lot of time, and only one minute is given to enter in one's answer using a clicker.

Following the quiz and question-answer session, students got into their lab groups and began work on the lab problem set for the week. For this week, students were to download an Excel spreadsheet from the 2011 Statistical Abstract of the United States showing the national average charges for tuition, room, and board from 1970 to 2009 for various kinds of institutions of higher education. Students were first asked to create a MATLAB variable that contains the year (1970--2009), tuition and fees for public 4-year universities in those years, board rates for public 4-year universities in those years, dorm charges for public 4-year universities in those years, and then the corresponding charges for private 4-year universities. The result is a 40 x 7 array of data.

Students were then asked to construct four plots, arranged in a 2x2 grid of subplots:

1. Tuition at public and private universities from 1970 to 2009 (i.e., two graphs on the same axes)
2. Board charges at public and private universities from 1970 to 2009
3. Dorm charges at public and private universities from 1970 to 2009
4. Total charges for tuition, dorm, and board at public and private universities from 1970 to 2009

These plots were to be constructed entirely from an M-file, pulling data from the .MAT variable created earlier. This task required students to draw upon previous knowledge (subplots, for instance) and use basic concepts from the videos, such as accessing single columns of a variable and creating a sum of multiple columns of an array for the fourth plot. Students were also asked to write a one-paragraph summary comparing the total charges for public versus private universities.

Finally, students were asked to create a data plot that addresses one of the following questions:

1. To what extent is there a relationship between the charge for tuition at US universities and the charge for dormitory space?
2. To what extent is there a relationship between the charge for tuition at US universities and the charge for board?
3. To what extent is there a relationship between the charge for dormitory space at US universities and the charge for board?

There were no explicit MATLAB directions given for this task; students were required to consider the question, think about what sort of data plot would best show the relationship (or lack thereof) being asked about, and then interpret the data plot logically and with clear communication. This task required students to map what they know onto a question for which they had no external directives on how to incorporate MATLAB and then defend their reasoning with writing, which is one way of viewing the "liberal arts" objective of the course.

Students turned in their work by submitting their .MAT variables to a shared Dropbox folder made for the course and then emailing their M-files and written materials. This allowed for simple grading of the lab problem sets by running the students' M-files using the variables they submitted, and then reading the code for readability issues.

Meanwhile, in the afternoon following the class meeting, the Guided Practice for the next week's meeting is posted to the course blog, and the cycle starts over.

Results and long-term effects of the inverted classroom

The inverted classroom model is a significant change from most students' conceptions of how a course should be run. It places the responsibility for learning on students (although it does not consume more time in students' schedules than the same course set up using the traditional classroom model), and some students resist the radically active nature of this arrangement. Also, since many students tend to conflate lecturing with teaching, the absence of a live classroom lecture leaves many students to feel they are not being taught. Unsurprisingly, many students initially expressed feelings about this model ranging from uncertainty to hostility.

Happily, though, most students came to accept and even embrace the inverted classroom by the end of the course after they saw that it empowered them to learn interesting, challenging, and useful knowledge on their own. The inverted classroom sets students up to experience what

psychologist Mihály Csíkszentmihályi describes as "flow"[4], which he defines as "optimal experience" and "the way people describe their state of mind when consciousness is harmoniously ordered." Examples of individuals experiencing flow would include musicians engaged in the creation of music, athletes performing at a high level in their sports, and chess players whose minds are completely engaged by the game at hand. When a person is in a state of flow during an activity, there is a correlation between skill level and challenge level, with a combination of high skill level and high challenge producing the most complex form of flow.

Csíkszentmihályi establishes three necessary conditions for an activity to become a flow-inducing or "autotelic" activity:

- *Clarity*: The expectations for the activity are clear and add helpful structure.
- *Balance*: There is a balance between skill level and challenge level. An activity with too little challenge for the skill level of the individual produces boredom; an activity with too much challenge produces anxiety.
- *Feedback*: The activity provides (or gives access to) clear and timely feedback to help the individual move from a state of anxiety or boredom into a more balanced state.

Many of Csíkszentmihályi's conditions for flow are intrinsic to the inverted classroom:

- The Guided Practice sets up clear goals and expectations for student work, and those goals are attainable and aligned well with the students' skill sets.
- The exercises in Guided Practice are simple enough so that direct and immediate feedback is provided by the exercises themselves; that is, it is immediately clear whether or not a student has completed the exercises correctly because students get direct results from MATLAB.
- The fact that the most difficult components of learning in the course are done in class, with the instructor present and actively guiding student work, means that direct and immediate feedback is also readily available during the most complex tasks. This is perhaps the greatest advantage of the inverted classroom over the traditional.
- The lab problem sets are created to mimic real-world problems, making the problems intrinsically rewarding.
- By doing the most difficult tasks in a limited time period (the 75-minute class meeting), students are more likely to concentrate and become absorbed in their work.

In the semester following the first offering of this course, when the some of the alumni of the course were enrolled in Calculus 3, students from the MATLAB course were given a survey as to usage or, and attitudes toward, MATLAB and other software. There were only seven students in this category, so the results of the survey are of limited statistical value, but the responses are revealing nonetheless.

When asked, "What one topic, concept, or skill stands out as being particularly important from Computer Tools for Problem Solving?", student responses included:

- *How to solve a problem by writing an algorithm and how to think about a problem logically.*
- *Through exercises in class, I improved upon my critical thinking skills.*

- *Being able to use the Help in MATLAB effectively. I never used the help section in a program before.*
- *I really liked Plot Tools when I graphed a function. I will more than likely use those graphs later in life[...].*
- *I learned that discovering things on my own is generally more beneficial than having things spelled out.*

Interestingly, only a few of the students mentioned specific components of the software as being the most important elements from the course. Most students listed general intellectual and critical thinking skills instead.

Also on a positive note, Tables 1 and 2 show (respectively) a heightened comfort level with using MATLAB and computer software in general in Calculus 3 and a high self-rated efficacy in utilizing print and video resources to acquire new skills. This is quite encouraging, since lifelong learning is both a mission statement component of the college and a key element of continued intellectual growth in the modern world.

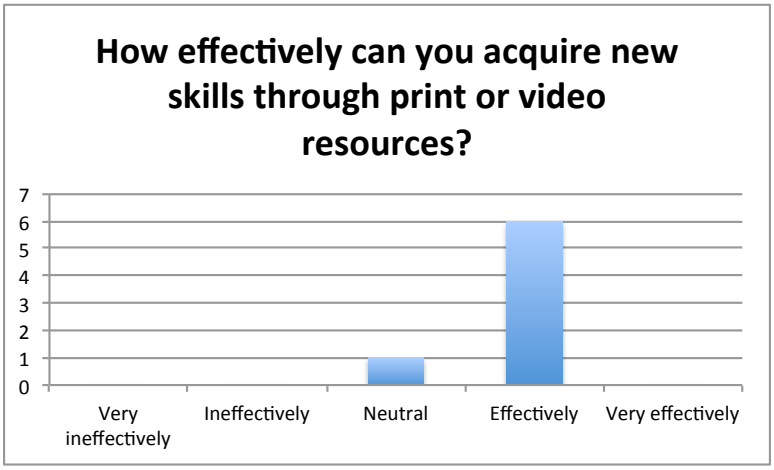

Table 1: Self-rated effectiveness in acquiring new skills among former MATLAB students currently taking Calculus 3.

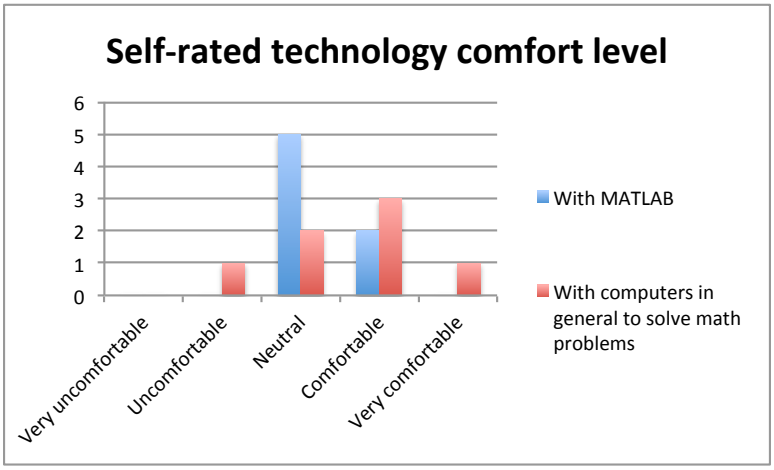

Table 2: Self-rated comfort level in using technology among former MATLAB students currently taking Calculus 3.

Although these are encouraging signs, there are also signs that more care needs to be taken to make MATLAB more useful after the MATLAB course is over. Tables 3--5 show that most students did not use MATLAB frequently, either in or outside of Calculus 3, and that students did not typically find MATLAB to be very useful in Calculus 3.

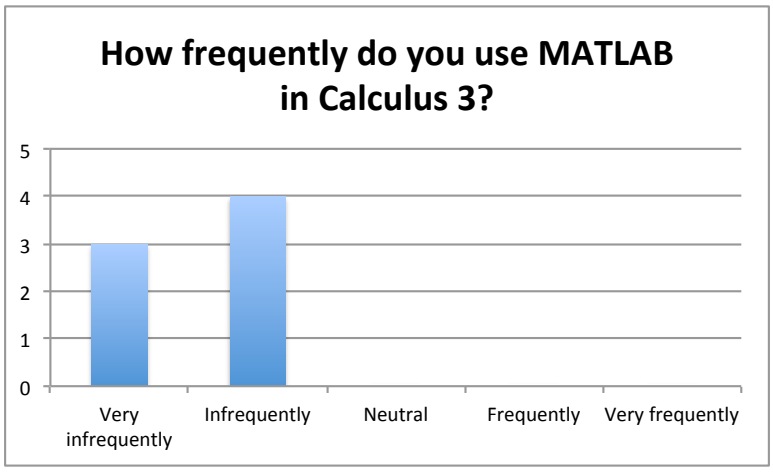

Table 3: Self-reported frequency of use of MATLAB in Calculus 3 among former MATLAB students.

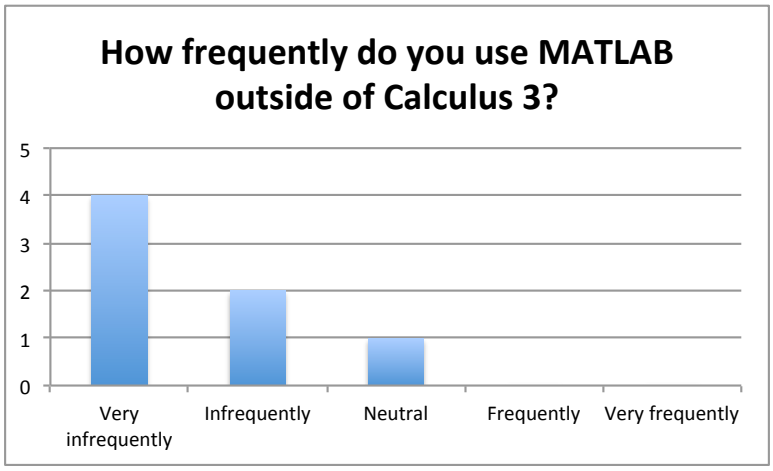

Table 4: Self-reported frequency of use of MATLAB outside of Calculus 3 among former MATLAB students.

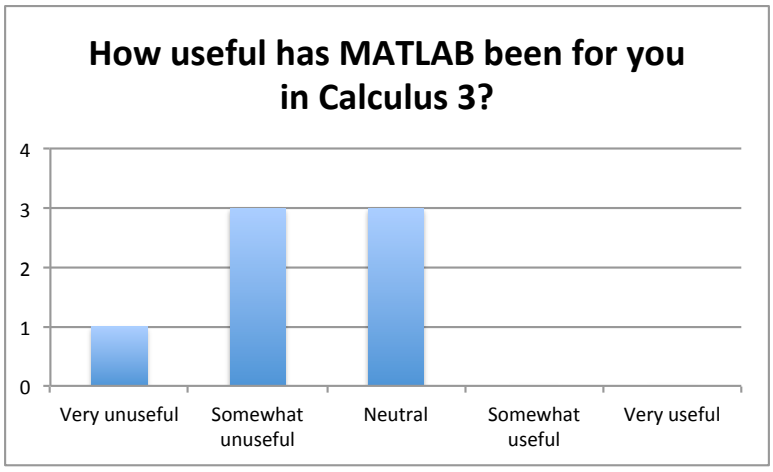

Table 5: Self-reported usefulness of MATLAB in Calculus 3 among former MATLAB students.

There are at least two possible explanations for these results. First, although MATLAB was used in targeted situations in Calculus 3 (mainly to make three-dimensional plots), the software had not yet become part of the mainstream in that course. Second, the first offering of Computer Tools for Problem Solving -- out of which these students were coming -- did not include a treatment of the Symbolic Math Toolbox and MuPad, which are much more applicable to Calculus 3 than the main MATLAB software. In fact, the decision to include two class meetings on the Symbolic Math Toolbox in subsequent offerings of the MATLAB course was primarily driven by the desire to make MATLAB more useful to students in Calculus 3.

Management and effective practices in the inverted classroom

The inverted classroom shows great promise for deepening student learning when it is used effectively. There are some challenges to the instructor in attaining effective use of this model. Below, we discuss some of these challenges and potential solutions.

Perhaps the most prevalent question in the inverted classroom is how to get students to complete the pre-class assignments prior to class. In the Computer Tools for Problem Solving course, three approaches seemed to be effective:

1. Provide structure to the pre-class assignments through Guided Practice. The simplest and least coercive way to get students to complete the pre-class assignments is to provide them with a reliable, structured means for doing so. The Guided Practice assignment described above accomplishes this by laying out clear learning objectives, providing a digestible amount of high-quality print and video learning resources, giving exercises that lead to early learning successes and direct feedback, and giving students some choice in what they would like to do and when. Having a staggered deadline for submitting their exercises -- with one of the three exercises due within a couple of days of the assignment -- helped students engage with the assignment early on, and once they had an initial engagement with the exercises, they typically could finish the remainder well in advance of the class meeting.

2. Give a quick, simple assessment of the pre-class assignment as an “entry ticket” to the in-class session. The five-minute clicker quiz at the beginning of the class meeting keeps the students honest about completing the pre-class assignments. The data from the clicker quizzes also provides instantaneous feedback for students (which is an important component of “flow”) as well as data to drive the question-and-answer session preceding the lab session.

3. Set clear ground rules about what will and will not happen in the class meetings and abide by those rules consistently. Setting appropriate boundaries and abiding by them is crucial. The syllabus for Computer Tools for Problem Solving clearly states that there is no lecture and no “re-teaching” during the class time. Students are instructed that they may ask as many questions about the material as they like during the week via office hours, email, or blog comments, but once the class starts, it will be assumed that all their questions have been addressed. If a student asked a question about MATLAB commands learned during a previous class, he was told to look it up in the Help documentation. If a student requested further tutoring or re-teaching on material from the videos, that request was re-directed to the student's lab partners. Holding firm to the ground rules and explaining why doing so is helping them learn keeps students from developing the sense that the instructor will “bail them out” during the class if they don't do the pre-class assignment.

It also helps to provide students with a framework for structuring their time spent outside of class. Students in Computer Tools for Problem Solving were given the following recommended breakdown of how to spend time outside of class:

- Spend one hour per week doing the video viewing and reading tasks, including doing exploratory work and playing with ideas from the viewing and reading, to be completed within the first two days after the Guided Practice is posted.
- Spend between one and two hours per week working through the Guided Practice exercises with the end goal being to perform with fluency all the tasks given on the Competencies list for the week. At least one exercise must be submitted within three days of the Guided Practice being posted. Finish the remainder over the following 3-4 days prior to the class meeting.
- Spend about 30 minutes per week, preferably the day or night prior to the class meeting, reviewing the Competencies lists from previous weeks and practicing tasks that have possibly been forgotten.

Without accounting for office hours visits and the like for obtaining help, this amounts to between 2.5 and 3.5 hours per week outside of class working on class material, which is appropriate for a one-credit course. It was also stressed that students spending more than this amount of time on a regular weekly basis should seek help in office hours.

Another issue involving time is the overhead required for creating the video and print resources for the out-of-class component of the inverted classroom. There are two points to make regarding this. First, the number of quality resources already available is increasing as more instructors use this model. Therefore it is conceivable that instructors wanting to use the inverted classroom need not create their own materials but just use existing ones, screening them beforehand for appropriateness.

Second, if an instructor should choose to make his or her own materials, then the tools for doing so are increasingly inexpensive and easy to use, and the instructor can take heart that for the most part the creation of video materials is a one-time startup expense. The creation of videos can be started (possibly completed) during a semester break or a sabbatical. For effective practices on how to create screencasts, instructors can access a series of articles at the author's blog[12].

It is also important to create in-class activities that are meaningful and engaging to students. Whenever possible, it is helpful to use real data and content domains that appeal to students' familiarity. For example, in Computer Tools for Problem Solving, students worked with problems from their Calculus classes, problems involving analysis of higher education data (see above), and problems involving gas prices. The idea is to create lab activities that are demonstrably useful, challenging, accessible, and intrinsically rewarding.

Finally, instructors using the inverted classroom must make a special effort to create a positive classroom environment and to forge relationships with students that engender mutual trust, so that students can feel confident and supported as they step out to learn on their own. Many students have never been asked to learn and implement a new concept in school before with this level of independence, although all of them have done so outside of school (learning language, learning a musical instrument or a sport, etc.). The challenge is high, and the support from the

instructor needs to be proportionally great. Students will rise to the challenge if they feel they are in a safe place to do so.

Conclusion

The inverted classroom shows great promise for providing students with a learning experience that can persist after the course ends. In the inverted classroom, students shift from being passive recipients of information to active evaluators and users of information, and the instructor shifts from an impersonal lecturer to an involved coach. The classroom environment shifts from a transactional model to a relational model, substituting the transfer of information with personal guidance through problems that are difficult and meaningful. Students are trained not only on course content but on how to acquire and assimilate content once their university coursework is finally over. In short, the inverted classroom prepares students to be learners.

In terms of instruction of MATLAB and other computing topics, the inverted classroom seems particularly well-suited, as MATLAB and other software are in a state of continual flux, and the specific content students learn today may be obsolete by the time they enter the engineering workforce. For future work on the inverted classroom in computing instruction, two lines of research are conceivable. First, it would be useful to see whether, in the short term, there are significant gains in student learning on computing tasks (including but not restricted to programming) for students in an inverted classroom versus a traditional one. Second, and perhaps more interesting, is whether students from an inverted classroom show a heightened ability to learn on their own and show an improved attitude toward learning in general versus those in an traditional classroom. Instruments such as the Motivated Strategies for Learning Questionnaire[6] could be of use in the latter kind of study.

References

1. Barrows, H.S. (1996). Problem-based learning in medicine and beyond: A brief overview. In Wilkerson, L & Gijselaers, W.H. (eds). New directions for teaching and learning, no.68. Bringing problem-based learning to higher education: Theory and practice, 3-13. San Francisco: Jossey-Bass.

2. Bloom, B. S. (1956). *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook 1, Cognitive Domain.* New York: David McKay.

3. Csíkszentmihályi, M. (1990), *Flow: The Psychology of Optimal Experience*, New York: Harper and Row.

4. Csíkszentmihályi, M., S. Abuhamdeh, and J. Nakamura (2005). “Flow”, in Elliot, A. Handbook of Competence and Motivation, New York: Guilford Press, 598--698.

5. Gannod, G., J. Burge, and M. Helmick (2008). Using the inverted classroom to teach software engineering. *Proceedings of the 30th international conference on Software engineering*, Leipzig, Germany, Association for Computing Machinery, 777--786.

6. Johnson, G. R. et al. (1991) Teaching Tips for Use of the Motivated Strategies for Learning Questionnaire (Technical Report No. 91-B-005). Ann Arbor, MI: National Center for Research to Improve Postsecondary Teaching and Learning Retrieved from ERIC database. (ED 338123)

7. Lage, Maureen J, Glenn J Platt, Michael Treglia, J Lage, and Glenn J Platt, 2011. Inverting the Classroom: A Gateway to Creating an Inclusive Learning Environment. *Journal of Economic Education*, 31 (1): 30-43.

8. Lei, Z. and Chengxiang, X. (2010). Problem-based learning in civil engineering education. *2010 International Conference on Education Technology and Computers (ICETC)*, 3(V3--41).

9. Moravec, Marin, Adrienne Williams, Nancy Aguilar-roca, and Diane K O Dowd. 2010. Learn before Lecture: A Strategy That Improves Learning Outcomes in a Large Introductory Biology Class. *CBE -- Life Sciences Education* 9: 473- 481. doi:10.1187/cbe.10.

10. Polya, G. *How to Solve It*. Garden City, NY: Doubleday 1957.

11. The Flipped Class Network website, http://vodcasting.ning.com/.

12. Bundle of links to blog posts on screencasting, http://bitly.com/lQfv8h.