Application of Mastery Learning in an Online MATLAB Programming Course

Dr. James Edward Toney, The Ohio State University

James Toney earned the Ph.D. in applied physics from Carnegie Mellon University in 1998 and the B.S. in electrical engineering from Rensselaer Polytechnic Institute in 1984. He is a Senior Lecturer in the Department of Engineering Education at Ohio State, where his focus is on curriculum development for teaching computer programming, primarily in MATLAB. He has previously held R&D positions at Battelle, Penn State Electro-Optics Center, and SRICO, Inc., where he worked on modeling, design, and testing of electro-optic devices and sensors. He is the author of the textbook, Lithium Niobate Photonics.

Application of Mastery Learning in an Online Intermediate MATLAB Programming Course

Introduction

The Ohio State University has long offered an intermediate course in MATLAB programming for engineering majors as one option to satisfy the computer science requirement of their degree program. During the COVID shutdown of 2020-2021, all instruction moved online temporarily; since resuming in-person instruction, we have continued to offer one online section of the course per semester, in addition to several on-campus sections. In the Fall 2022 semester, we piloted a self-paced, mastery-learning model for the online section, while the in-person sections continued to follow a traditional format.

Mastery Learning

The mastery learning approach was articulated in the 1960s by Bloom [1], who saw it as enabling nearly all students to achieve mastery of a subject, despite variations in aptitude and learning styles. The essential idea, which derives from Carroll [2], is that variations in aptitude do not imply differences in the capacity to master the material, only to differences in the time required to achieve mastery. Mastery learning is therefore closely linked to self-paced instruction.

A review of prior work on mastery learning in computer science education is given in [3]. The authors discuss mastery learning as a means to address the typical bimodal distribution in introductory programming courses - which we have routinely observed in this course as well - by requiring students to achieve competence in foundational skills before moving on to more advanced material. This is consistent with previous findings that mastery learning has a larger positive effect on lower aptitude students [4]. This was our chief motive for adopting the mastery learning approach.

Course and Student Background

The course is intended for engineering majors who have had a prior introduction to MATLAB programming in the first-year engineering sequence. Roughly equal numbers of sophomores, juniors, and seniors are enrolled, with a few freshmen in the spring semester. The majority of students are from the least computationally focused majors in Engineering, such as Civil, Chemical, Biomedical, and Aeronautical Engineering. The demographic breakdown is typically about one-third women, one-sixth underrepresented minorities, and one-sixth international students. All students have taken the standard (non-honors) Fundamentals of Engineering I course or equivalent.

Course Organization

The course is divided into six units:

1. Scalar and Array Operations, Basic Input-Output and Plotting
2. Logical Arrays and Data Analysis, User-defined Functions
3. Branching, Looping, and Program Structure
4. Testing & Debugging, File Input/Output, Cell Arrays, Struct Arrays
5. Graphical User Interface Programming with App Designer
6. Simulations

The first 3 units contain the core material that constitutes basic proficiency in MATLAB; completion of these units is therefore sufficient for a minimum passing grade. The latter 3 units deal with more specialized or advanced topics; completion of all units is necessary to earn an A grade.

Each unit has a single summative assessment (unit exam) and two types of formative assessments: programming exercises, which are short problems, typically requiring less than 10 lines of code, and application assignments, which require longer, more complex programs. The exercises are implemented in the MATLAB Grader platform from Mathworks [5], an auto-grading system that facilitates mastery learning by allowing students to submit repeatedly until they pass the exercises. Students are required to complete all exercises in a unit at the 90 % level before taking the unit exam.

Application assignments are submitted through the learning management system (LMS) and graded manually by undergraduate teaching assistants. A minimum score of 80 % on each application assignment is required to pass the unit; students earning less than 80 % must revise and re-submit the assignment.

Conduct of Classes

The course is delivered in a synchronous online format, with a “flipped” structure. Recorded lectures are posted in the LMS, while class meetings are focused on review of concepts, presentation of example problems, and one-on-one help sessions between students and the instructor/teaching assistants. The latter are facilitated by Zoom breakout rooms. Polling questions in the Top Hat platform are used to engage students in the group review session.

There is no required textbook, but students are provided with a free, interactive e-book with embedded checkpoint questions [6]. Students are encouraged to read the material and answer the questions, but their responses are not recorded or scored. A code examples file for each class day is provided in the form of a MATLAB live script; these files are the basis for most of the presentations in the lecture recordings. Powerpoint presentations are also used occasionally, and the slides are provided to the students to review at their convenience. The variety of resources provided – e-book, lecture recordings, live scripts, and Powerpoint slides – allows each student to utilize them in a way that fits their learning style. Visual learners may prefer reading the e-book and Powerpoint slides, while auditory learners may benefit more from the lecture recordings. Regularly scheduled office hours with the instructor and teaching assistants are available for students who desire tutoring. We have discussed the use of the various Mathworks resources in the classroom in a previous paper [7].

As an incentive to participate in the class sessions, students are given some credit towards their course grade if their score on the Top Hat questions is 80 % or greater. (Details of course grading are described in the next section.) For each question, ½ point is given for participation, and ½ point for correctness. In Fall 2022, slightly fewer than 3/5 of the students attended and participated regularly enough to earn this credit.

Summative Assessments and Grading

In lieu of the traditional two mid-terms plus final exam, there are six unit exams, with course grades determined primarily by the number of units completed. Students must re-take the exam if they fail to achieve the minimum passing score of 80 %. A minimum of three of the six units must be completed at the 80 % level to pass the course, while all units must be completed at the 90 % level for an A grade.

In Fall 2022, exams were administered in Microsoft's Azure Labs remote desktop environment. Required data files were loaded onto the virtual machines in advance, and exam instructions were incorporated into a MATLAB live script template that was downloaded from the LMS during the exam. While the experience with Azure Labs was largely positive, there were occasional performance issues, such as slow response, students being disconnected from their virtual machine, or errors while reading data files from disk. While these problems are minor in the context of a homework or lab assignment, they are stressful for students and instructors in an exam context. Consequently, we do not expect to use to use Azure Labs for exams again.

Students submit the completed live script through the LMS. There are multiple, roughly equivalent versions of each unit exam; students who repeat an exam take a different version each time. Exams are graded on a 10-point scale, with a minimum deduction of 0.5 point.
A representative portion of an exam template is shown in the Appendix.

Students' course grades are determined by the number of units completed and their performance in each unit, based on a point system. If the average of their scores on the unit exam and the application assignment is 90 % or greater, they receive 2 points for the unit. If it is less than 90 % but 80 % or greater, they receive 1 point. Students also earn 1 point added to their total for the course if their class participation score (primarily based on the Top Hat questions) is 80 % or greater.

The grading scale is as follows:

Table 1: Determining the Course Grade

Units Completed	Points	Grade
6	12 - 13	A
6	10 - 11	A-
6	8 - 9	B+
6	6 - 7	B
5	9 – 10	B+
5	6 - 8	B
5	5	B-
4	7 - 8	C+
4	5 - 6	C
4	4	C-
3	5 - 6	D+
3	3 – 4	D
0 – 2	N/A	E

There is an optional final exam – the same one given to in-person sections – that gives students an opportunity to raise their course grade by one level (from B+ to A-, for example). Taking the optional final cannot hurt a student's grade. Table 2 shows how the final exam score affects the final grade.

Table 2: Determination of Course Grade for Students Taking the Optional Final Exam

Grade from Table 1	Final Exam Score	Final Course Grade
A-	93 – 100 %	A
A-	< 93 %	A-
B+	90 – 100 %	A-
B+	< 90 %	B+
B	87 – 100 %	B+
B	< 87 %	B
B-	83 – 100 %	B
B-	<83 %	B-
C+	80 – 100 %	B-
C+	<80 %	C+
C	77 – 100 %	C+
C	< 77 %	C
C-	73 – 100 %	C
C-	< 73 %	C-
D+	70 – 100 %	C-
D+	< 70 %	D+
D	67 – 100 %	D+
D	<67 %	D

The option to improve the course grade by taking the final exam is not available to students who have passed fewer than three unit exams and therefore have a failing grade. There is no benefit to taking the final exam for students who have already secured an A based on Table 1.

Outcomes

A challenge in a mastery learning-based course is designing the unit exams at an appropriate level – not so difficult that most students require multiple attempts on most exams, but not so easy that most students routinely earn a perfect score on the first attempt. Another concern, with multiple versions of each unit exam, is ensuring that the versions are sufficiently different that students cannot simply memorize the solution from the first attempt, but similar enough to be equivalent in difficulty.
Figure 1 shows the number of attempts and passes on each exam in Fall 2022. Each student took the A version of each exam first; versions B and C were used for second and third attempts, respectively. With the exception of Unit 5 (GUI programming), which all students passed on the first attempt, between one-quarter and one-third of students typically required multiple attempts, with a small number requiring a third attempt. We conclude that the difficulty level of the exams was generally appropriate, although the Unit 5 exam could have been more challenging.

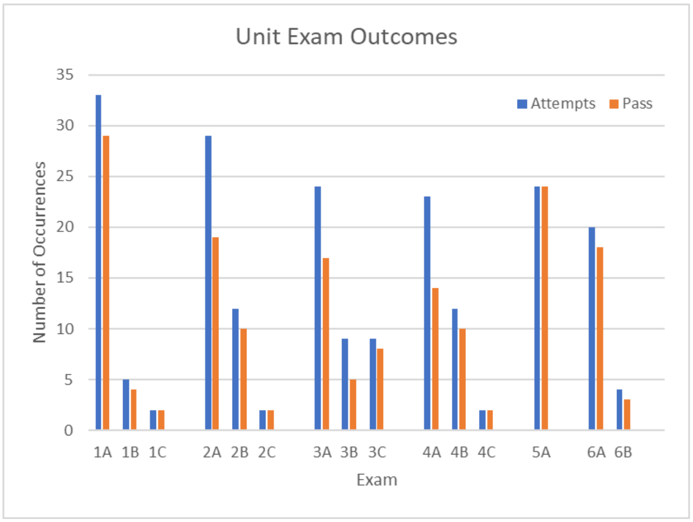

Figure 1: Attempts and passes for each version of the unit exams.

A concern with self-paced instruction is that some students, without the pressure of firm deadlines, may postpone most of the work until the last few weeks of the term. Even without strict deadlines, students may feel that they have “fallen behind”, which can be a source of anxiety and an impediment to course completion [8]. Figure 2 shows the distribution of completion dates for each unit by week. The spike in completions during the last week shows the extent of the “falling behind” issue.

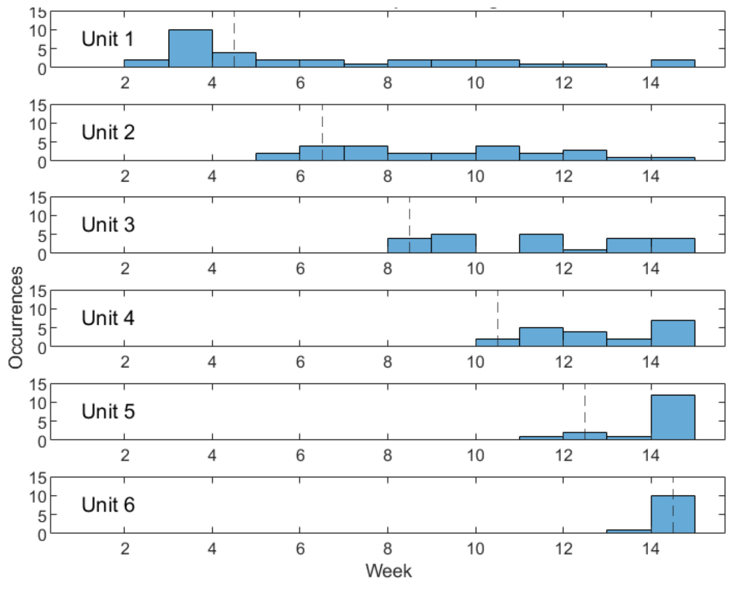

Figure 2: Unit completion by week. The vertical dashed lines are the recommended completion dates

Of the 36 students who initially enrolled in the course, 33 remained enrolled throughout the semester, 29 completed the required 3 or more units to pass the course, and 18 completed all 6 units. For comparison, in the in-person section taught by the same instructor, 31 of 36 students remained enrolled, and 29 earned a passing grade. Figure 3 shows the distribution of number of units completed.

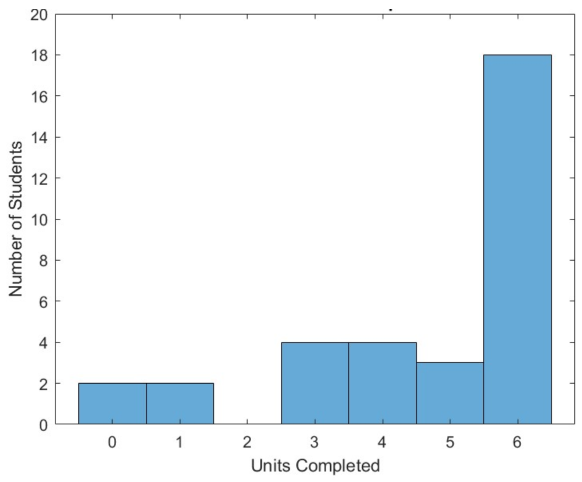

Figure 3: Number of units completed for the students who remained enrolled until the end of the semester

Student Feedback

Several optional, anonymous surveys were administered to the students to gauge their attitudes about the course structure and content. The responses to the last two surveys, in particular, show a bifurcation in views – the students are roughly evenly divided between those who liked the self-paced/mastery learning format and felt that it enhanced their learning, and those who felt that the absence of hard deadlines hindered their progress.

While students self-selected to enroll in the online section, they did not know prior to the start of the semester that it would use a self-paced / mastery learning approach. Had students made a conscious choice to enroll in a self-paced course, it is likely that there would be fewer negative responses.

Selected queries and responses are shown below. In some cases, responses were not mutually exclusive – that is, a respondent could potentially select two responses that might seem to be contradictory. Likewise, a respondent might choose neither of two complementary responses.

From Survey 4 of 5 (24 respondents): How satisfied are you with your progress in the course so far?

Response	Number
Satisfied, and I expect that I will be able to achieve my learning goals	12
A little unsatisfied, and I'm not sure if I can achieve my learning goals	4
Well behind where I would like to be, and it will be a challenge to achieve my learning goals	7
Hopelessly behind, and I see little chance of a positive outcome	1

From Final Survey (22 Respondents): How do you feel that the self-paced, mastery-learning approach of this course has affected your learning? (Select all that apply)

Response	Number
The flexibility on due dates was a great help	10
The lack of firm due dates was a hinderance and contributed to my falling behind	7
The minimum performance requirement was a good incentive to master the material	9
the minimum performance requirement made it very difficult for me to achieve a good grade	8
The flexibility in scheduling helped me to perform well on the exams	8
The lack of fixed exam dates made it hard for me to stay on track in the course	8

In addition to questions related to the self-paced / mastery learning format, some questions on the final survey sought to gauge students' views of their final programming proficiency and preparedness to use MATLAB in the future. 4 – 5 respondents of 22 gave negative answers. For comparison, in the in-person section, 8 out of 26 respondents gave negative answers. (A formal statistical comparison between the two sections would not be appropriate, since there was no randomization of the student population.)

From Final Survey (22 Respondents): How has completing this course changed your confidence in your programming ability?

Response	**Number**
I feel significantly more confident and proficient in programming now.	11
I was already pretty confident. This course increased my proficiency incrementally.	5
I was already pretty good and didn't really gain much from this course.	1
I still don't have much confidence in my programming ability.	5

From Final Survey (22 Respondents): If you are asked to use MATLAB in one of your major courses in the future, how do you think you will feel about that?

Response	**Number**
Cool! I like MATLAB!	7
That's OK, I don't mind MATLAB.	10
All right, but I'd really rather use Python or C or Java.	1
Oh no, anything but MATLAB!	4

Conclusions

In our implementation of mastery learning / self-paced instruction in an online intermediate MATLAB programming course, the rate of successful completion was comparable to the on-campus section using a traditional structure. However, we do not have sufficient data for a detailed comparison of the level of mastery achieved by students in the two sections. The usual pitfall of self-paced courses - the “falling behind” problem - was evident both in students’ progress and in their feedback about the course.

For this model of instruction to be viable, some measures must be put in place to prod students to stay on schedule. One possibility is modifying the point system so that each unit must be completed by a certain date for the full 2 points to be awarded. This approach has the benefit of imposing consequences for procrastination, while avoiding hard deadlines, which are contrary to the spirit of mastery learning. Another measure that may be helpful is advertising some sections prior to registration as self-paced, so that students must consciously commit to taking responsibility for regulating their own learning.

References

1. B.S. Bloom, “Learning for Mastery”, *UCLA - CSEIP - Evaluation Comment. Vol. 1. March 1968.*
2. J. Carroll, “A model of school learning”, *Teachers College Record*, 1963, pp. 723-733.
3. J. Garner, P. Denny, and A. Luxton-Reilly, “Mastery Learning in Computer Science Education” in *Twenty-First Australasian Computing Education Conference (ACE’19)*, January 29–31, 2019, Sydney, NSW, Australia.

4. C-L C. Kulik, J. A. Kulik and R. L. Bangert-Drowns "Effectiveness of Mastery Learning Programs: A Meta- Analysis", *Review of Educational Research* , Vol. 60, No. 2, 1990, pp. 265-299 https://www.jstor.org/stable/1170612
5. https://www.mathworks.com/products/matlab-grader.html
6. J.E. Toney and A. Jayakumar, *MATLAB Programming for Engineering Applications*, url: ohiostate.pressbooks.pub/matlab (2023).
7. J.E. Toney and A. Jayakumar, "Utilizing the Full Range of MATLAB Capabilities in the Classroom," Proceedings of the 2019 ASEE North Central Section Conference. 2019.
8. J. Campbell, A. Petersen, and J. Smith, "Self-paced Mastery Learning CS1", in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19),* February 27-March 2,2019. https://doi.org/10.1145/3287324.3287481

Appendix – Partial Live Script Exam Template

ENGR 1221 Online

Exam 2 Version B

1. Load the data file, greenhouse`.mat`, using the command:

```
load greenhouse.mat
```

This will load a matrix called greenhouse into the workspace. The matrix contains one week of data from the temperature/humidity sensor in a home greenhouse, in 1-minute increments, taken during the first week of February 2022.

The format of the data matrix is:

Column 1: Day (1 – 8)

Column 2: Hour (0 – 23)

Column 3: Minute (0 – 59)

Column 4: Temperature in Fahrenheit

Column 5: Relative Humidity (%)

For example, if a row of the matrix contains

3   8 20 61.0 42.5

that indicates that at 8:20 AM on February 3, the temperature in the greenhouse was 61.0 ºF, and the relative humidity was 42.5 %

```
%
```

```
% Load the data file as shown above
%
```

2. Using the input() function, prompt the user to enter a day (a number between 1 and 8) and hour (a number between 0 and 23). *When you run the program, enter 3 for the day and 12 for the hour*

```
%
% Prompt user for day and hour
%
```

3. Calculate the average temperature and humidity for the day and hour specified by the user. **Select the correct range of rows using logical indexing or the find() function. Do not hard-code the indices!**

```
%
% Calculate average temperature and humidity for the specified day and hour
%
```

4. Print the results to the command window to one decimal place with a suitable message. For example,

```
On Day 2 between 12:00 and 12:59, the average temperature was xx.x degrees, and
the average relative humidity was yy.y %
```

```
%
% Print results
%
```

The dew-point temperature can be estimated by the following formula:

$$T_{dewpoint} = T_{Farenheit} - \frac{9}{25}(100 - humidity)$$

where *humidity* is the relative humidity in percent.

Write an *external user-defined function* to calculate the approximate dew-point temperature for a range of data and return the MAXIMUM and MINIMUM values of the dew-point.