ASEE'S VIRTUAL CONFERENCE

At Home with Engineering Education JUNE 22 - 26, 2020 #ASEEVC

Paper ID #30557

Integration of Instructional Technology Tools including Matlab Grader to Enhance Learning in a Hybrid Vibrations Course

Dr. Natasha Smith P.E., University of Virginia

Dr. Smith is an Associate Professor at the University of Virginia

Integration of Instructional Technology Tools including Matlab Grader to Enhance Learning in a Hybrid Vibrations Course

Abstract

This paper discusses the delivery of a 4-week summer course in Vibrations to a diverse group of face-to-face and online graduate and undergraduate students, with a focus on how instructional technology tools enabled learning. The tools were selected to engage students in multiple formats: 1) those that attended class in person, 2) those who participated in the live stream class, and 3) those who would watch the recorded class later in the day. The Matlab Grader feature was instrumental in providing students practice with computational solutions to vibrations problems. With this tool, instructors create assignments that students solve with a Matlab script. When the student submits his or her own code, it is automatically graded against the specified output of the instructor solution. Though it would be useful regardless of delivery mode, the Grader was particularly valuable in getting real-time feedback to students in keeping with the fast pace of the summer course. Other homework assignments were peer-graded within the learning management system. Lectures were delivered as 'live' notes using PDFAnnotator. Note taking has been shown to improve learning, so this was preferred over using premade slides. At the same time, traditional board notes would not have been easy for online students to read. The final instructor notes were then made available to all. Students were able to join the lecture via Zoom, a video conferencing platform. Pauses in lecture allowed for questions and comments from both the live class and via the chat feature within Zoom. The versatility of Zoom was also critical for evening office hours. The ability to screen share was especially valuable for addressing errors in student programs. The paper will discuss how various tools were implemented and their pedagogical value, as well as the limitations and challenges experienced. Metrics on student use of the Matlab Grader (e.g. number of attempts, etc) and student assessment for the course will also be provided.

Introduction

A Vibrations course was offered as a technical elective to a wide audience of both graduate and undergraduate engineering students during a 4-week summer term at the University of Virginia. In total, four sections were offered simultaneously: an undergraduate 'in person' section, and undergraduate web-based section, a graduate 'in person' section, and a graduate 'web-based' sections. The cohort included undergraduate Mechanical and Aerospace engineering students; Masters of Engineering (ME) students in Chemical Engineering, Civil Engineering, and Mechanical Engineering; Master of Science (MS) and PhD students in Mechanical or Aerospace Engineering; and a non-degree program student. Many of the ME students were enrolled in online degree programs targeting working professionals. The university requires a minimum enrollment of 10 students for summer courses, so the variety in delivery mode and student program was needed to generate a sufficient combined enrollment. Eighteen students completed the course. This diversity, however, created some challenges for providing an effective learning environment for each student. In addition, the short duration of the course required a demanding pace for homework, with quick feedback. A final hurdle was that this was the author's first time teaching Vibrations. Therefore, it was necessary to prioritize lesson plan preparation over delivery logistics. This precluded time intensive efforts such as learning complicated software or prerecording lectures in an instructional lab environment.

Mastery learning (or learner-centric) techniques introduced by Bloom in the 1970s and expanded by researchers over the last half century have an established track record for enhancing student learning.[1–3] In particular, periodic formative assignments are necessary and should be designed so that students reflect on mistakes and adjust their learning efforts as needed. Within the Vibrations course, the mechanisms for formative assessments included regular homework and Matlab assignments. For this to be effective given the accelerated pace of the course, it was necessary to find options for feedback other than manual faculty grading. Peer grading has proven to be consistent and reliable when used with massive open online courses (MOOCs). Luo et. al studied nearly 2000 peer graded assignments and found that scores were not statistically different than those graded by an instructor.[4] Beyond the benefit for the student being graded, peer grading forces the student grader to engage with the solution and provide constructive feedback to their peer. For the Matlab assignments, feedback was given through an automated grading tool called Matlab Grader.[5] Though this is a fairly new product, web-based grading tools for programming assignments have been around for over 2 decades.[6,7] They enable instructors to assign a larger quantify of smaller targeted assignments (since grading time is negligible), improve grading accuracy and consistency, and most importantly give students real-time feedback.

In addition to formative assessment, student engagement with course material is critical for learning. Waters et. al demonstrate the importance of attendance, preparation, and note-taking on student performance, particularly for those performing below the mean.[8] In particular, taking notes during lecture is more than a means for collecting a record of information presented; it is also a valuable tool for processing content.[9] For this reason, the Vibration course was designed to deliver notes in a traditional way (i.e. hand-written in real time) for all students. However, there was an initial question on whether the notes produced should be made available to students. The concern was whether students would bother taking their own notes if this was the case. However, separate studies by Perry at the University of Memphis and Cunningham et. al at Rose-Hulman found no statistical difference between test groups that for which instructor notes were and were not made available.[10,11] In addition to the daily engagement with course material through lectures and homework, a final project was introduced to engage students in higher level learning, i.e. evaluating and creating in accordance with Bloom's taxonomy for mastery learning.[12]

The instructional technology tools discussed in this paper were instrumental in meeting these unique challenges of the course with a reasonable amount of effort on the part of the instructor.

Course Design & Content

The content of the course was typical for a first course in mechanical vibrations, namely the development of and solutions for lumped-mass single and multiple degree of freedom models (with a cursory introduction to continuous systems). Matlab programs were integrated in several ways: 1) to visualize and compare responses, 2) to find numerical solutions to ordinary differential equation (ODE) models, and 3) to find eigenvalues and vectors for multiple degree of freedom systems. The course textbook, Rao's Mechanical Vibrations[13] includes Matlab examples in at the end of each chapter. Thus, it was not difficult to find, modify or create Matlab assignments to enhance student understanding of Vibrations concepts. However, the diverse group of students in the class came with a wide range of programming skills, from novice to expert. One of the course goals was to build upon these skills regardless of the starting point.

A weekly rhythm was established for course activities. The official course met for 2 hours and 15 minutes each day (slightly less than that for a whole week during a regular semester). This time was used for a lecture (typically 1-1.5 hours) and guided student work. Regular homework problems were due on Tuesday and Thursday evenings, worth 20% of the course grade. These were to be peer graded within 24 hours. Matlab problems were due on Wednesday nights; these were worth an additional 20%. Take-home exams were delivered on Fridays to be due the following Monday, worth 10% eachs. This schedule persisted for the first 3 weeks. In the final week, students presented independent work from the review of a scholarly article (graduate students) or results from an experiment or demonstration (undergraduates), worth 10%. A cumulative final exam, worth 20%, was also administered.

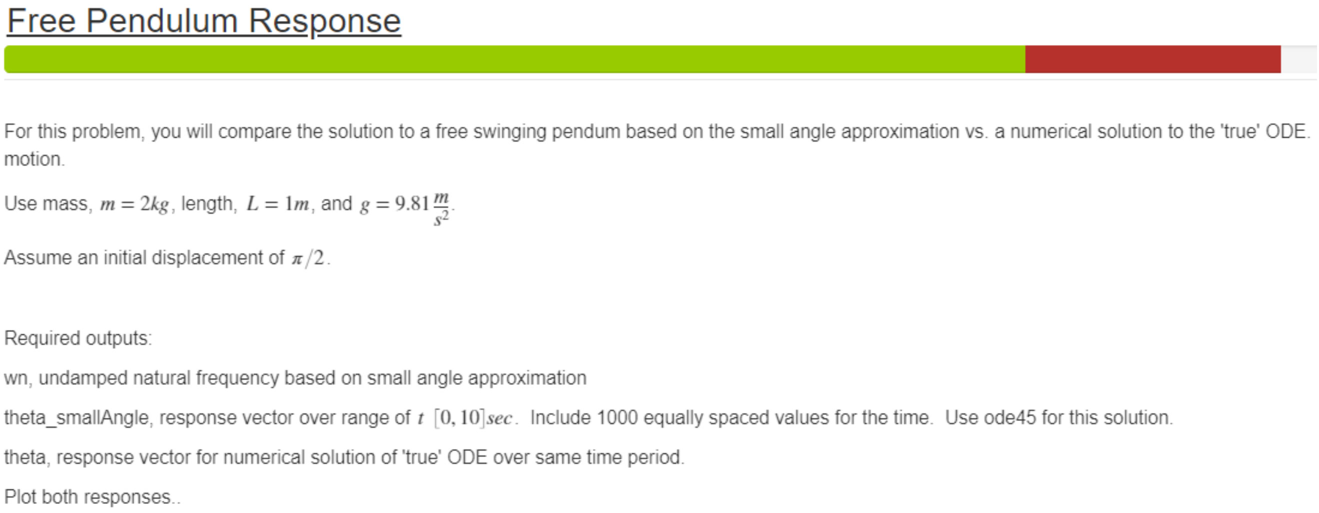
Implementation of Matlab Grader

Two characteristics of this course presented certain constraints for incorporating Matlab assignments. First, the varying levels of programming skills for students in the class meant designing assignments that were within reach of novice programmers, but which advanced students would also learn something new. Second, the short duration of the course meant students had limited time to complete the assignments, and the instructor had limited time to provide feedback. Finally, the assignments needed to, first and foremost, help students understand Vibrations concepts; it was important that programming did not overwhelm that priority. The Matlab Grader[5] was chosen to address these challenges.

The Matlab® Grader™ is a web-based environment for creating, executing, and assessing Matlab coding assignments. It has many useful features, not the least of which is that it provides real-time, automatically graded assessment for student work. In addition, the platform gives a lot of flexibility to instructors both in the problem design and in rubrics used for grading. Finally, it provides both detailed information on student attempts as well as aggregated metrics for the entire section. Though Matlab Grader has direct integration for several learning management systems (LMS), it was not available for Collab (the Sakai-based LMS used at the University of Virginia) at the time the course was offered. Nevertheless, creating the course and inviting students to participate was straightforward. Students were able to join using their university email address.

The course included 3 Matlab assignments (one per week). Each of these had 2-3 separate coding problems. For the first Matlab assignment, as students were getting introduced to the Grader platform, they were asked to solve a spring-mass-damper system, plot a Fourier series of a period function, and use an ODE solver to analyze a simple pendulum. In the second assignment, they plotted undamped forced vibrations of a system approaching resonance, used the ODE solver for a forced system with Coulomb damping, and created frequency plots for a system with base excitation. In the final assignment, they solved a 2 DOF forced system and conducted modal analysis of a 5 DOF system.

A Matlab Grader problem has four components: the problem description, a learner template, a reference solution, and assessment measures. The problem description could include formatted text and images. To facilitate the auto-grading feature, it was important to be specific with regard to variable naming standards. For example, the statement from the pendulum problem in the 1st assignment is provided in Fig. 1. The intent of this problem was to introduce students to a method for numerically solving ordinary differential equations (ODEs) used to model vibrating systems. In the assignment, they compare the numerical solution for a single mass on a pendulum, $\ddot{x}+\frac{g}{L}sin(x) = 0$, to an analytical solution applicable for small angles (i.e. by substituting $sin(x) \approx x$). In addition to the problem statement, students are provided a learner template. This is an incomplete script that can provide additional guidance for solving the problem. The instructor is able to lock individual lines which students are unable to change. For example, since the focus of the pendulum assignment was on using the Matlab ODE solver, the template code provided suggestive comment lines, input variables, partial lines with correct output variables, and lines for plotting, as seen in Fig. 2. One lesson learned from the first assignment, was that it was better to create templates without syntax errors. For Fig. 2 this would have meant either removing the incomplete output statements (e.g. as in line 14) or replacing them with a complete (but incorrect) filler line for students to change. For more complex problems, this allowed students to check their work incrementally.



Free Pendulum Response

For this problem, you will compare the solution to a free swinging pendum based on the small angle approximation vs. a numerical solution to the 'true' ODE. motion.

Use mass, $m = 2kg$, length, $L = 1m$, and $g = 9.81\frac{m}{s^2}$.

Assume an initial displacement of $\pi/2$.

Required outputs:

wn, undamped natural frequency based on small angle approximation

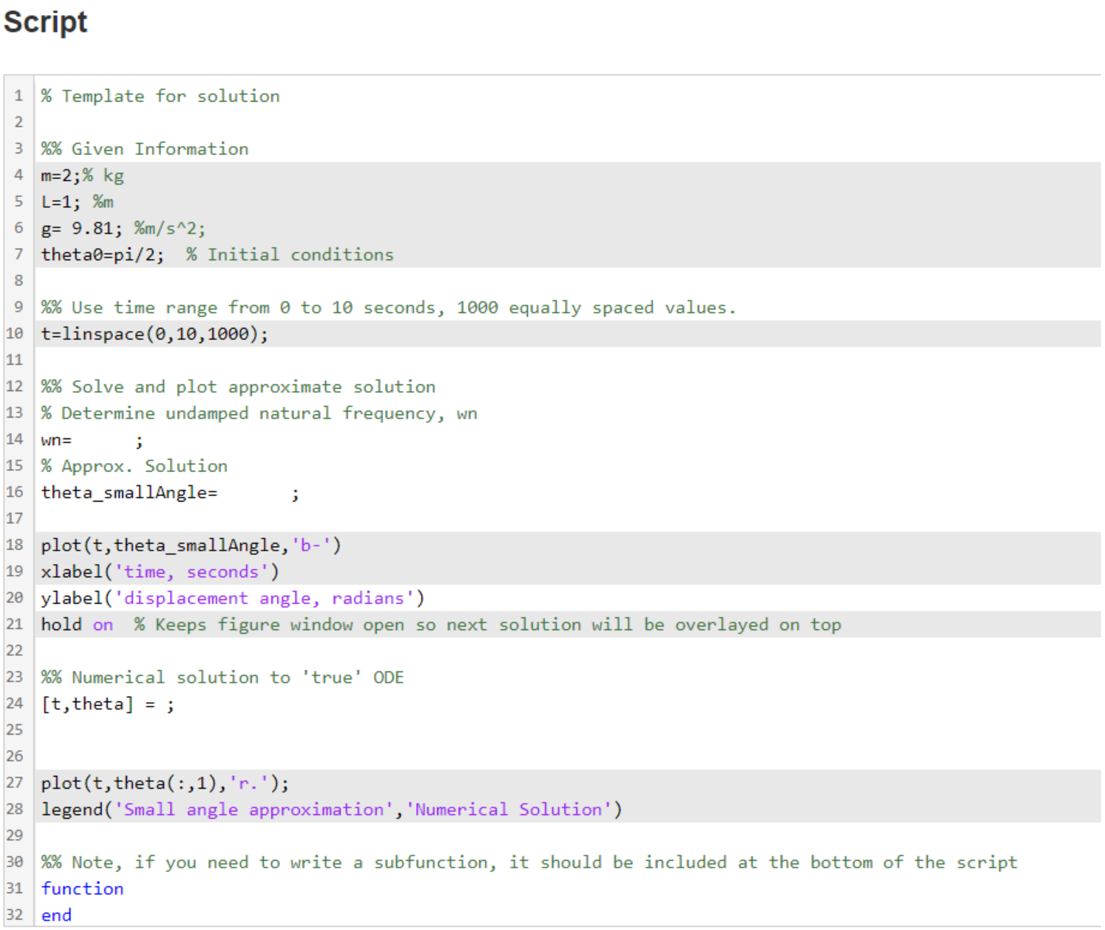
theta_smallAngle, response vector over range of t $[0, 10]sec$. Include 1000 equally spaced values for the time. Use ode45 for this solution.

theta, response vector for numerical solution of 'true' ODE over same time period.

Plot both responses..

Figure 1: Problem Statement for Pendulum Analysis.

The final steps to creating a Grader problem were to write a solution script, called the reference solution, and develop assessment metrics. The Grader has a few assessment options, the most basic



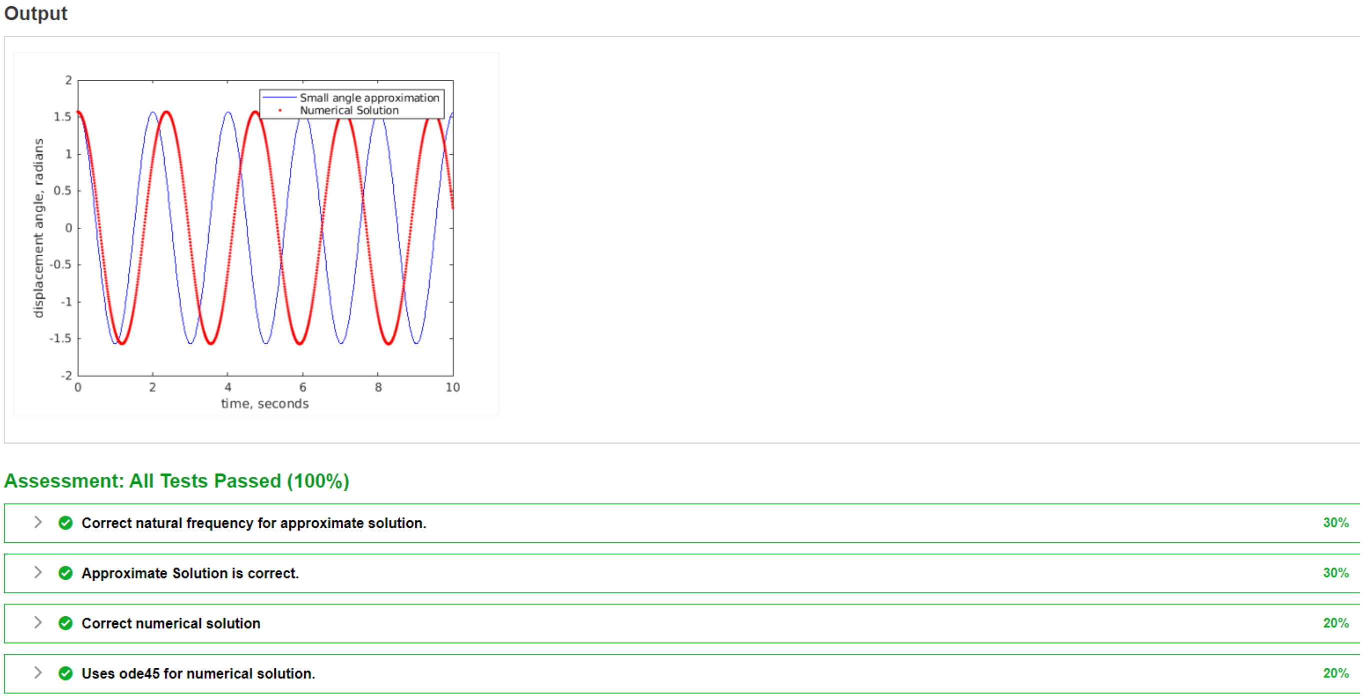
Script

```
% Template for solution

%% Given Information
m=2;% kg
L=1; %m
g= 9.81; %m/s^2;
theta0=pi/2;  % Initial conditions

%% Use time range from 0 to 10 seconds, 1000 equally spaced values.
t=linspace(0,10,1000);

%% Solve and plot approximate solution
% Determine undamped natural frequency, wn
wn=      ;
% Approx. Solution
theta_smallAngle=      ;

plot(t,theta_smallAngle,'b-')
xlabel('time, seconds')
ylabel('displacement angle, radians')
hold on  % Keeps figure window open so next solution will be overlayed on top

%% Numerical solution to 'true' ODE
[t,theta] = ;


plot(t,theta(:,1),'r.');
legend('Small angle approximation','Numerical Solution')

%% Note, if you need to write a subfunction, it should be included at the bottom of the script
function
end
```

Figure 2: Student template for Pendulum Analysis. Shaded lines are locked lines of code.

of which directly compares outputs from the student's code to that from the reference solution. In addition, one can check whether specific Matlab commands are or are not present. A screenshot of the metrics for the pendulum problem in Fig. 3 shows 3 checks of variables matching the solution, and one check to ensure the student solution used the 'ode45' command. Beyond these stock assessment types, it is also possible to write Matlab code for instructor designed assessments. Another lesson learned, was to write assessment code that would accept a reference solution vector as well as its transpose, so the check was not sensitive to a student's choice to use row or column vectors. Finally, in addition to the assessment measures themselves, the instructor can provide hints for students with incorrect solutions. The hint was often a reminder of the expected size of an output variable in comparison with the their own output.

When students receive the assignment in the Grader platform, they initially see the problem statement and the learner template. They then edit the template on the platform (i.e. from their browser). They are able to run their code and see the output at any time. The functionality of the Grader platform is essentially the same as a stand alone version of Matlab. Once submitted, the code is executed and students receive both the output from their code and the assessment. As these assignment were meant for formative assessment, students were given unlimited attempts. Grading was



Assessment: All Tests Passed (100%)

✓ Correct natural frequency for approximate solution.	30%
✓ Approximate Solution is correct.	30%
✓ Correct numerical solution	20%
✓ Uses ode45 for numerical solution.	20%

Figure 3: Grading rubric for Pendulum Analysis.

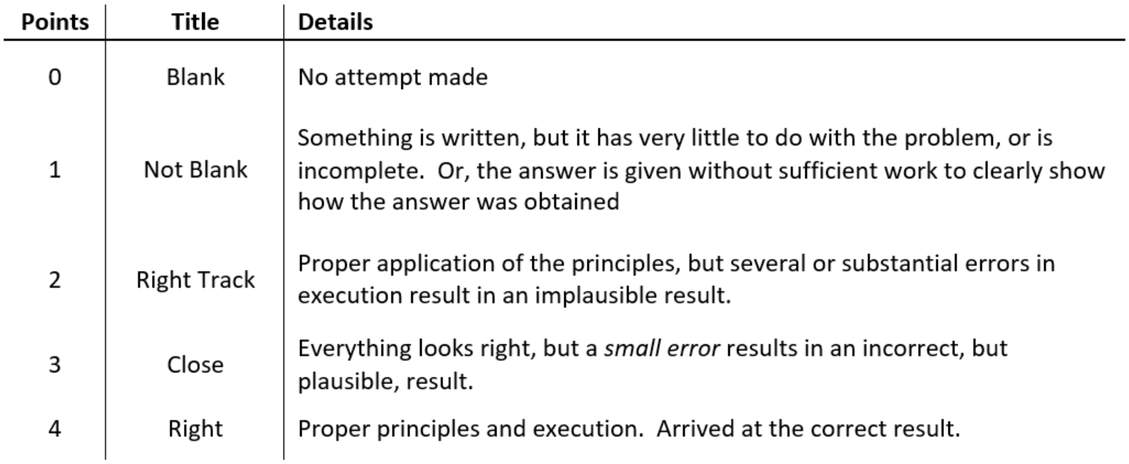
kept simple, with each check within each problem equally weighted.

In all, the Grader tool addressed the constraints of the course. Learner templates helped to create assignments that were achievable for those with novice programming skills. The student contribution was limited to parts of code directly relating to Vibrations concepts. They benefited from visualizing solutions from code output without needing to know the syntax of every line. At the same time, programmers who were able to comprehend the entire script could adapt it to a wider variety of problems. Often, these students used adapted Matlab scripts to solve or verify exam problems. In addition, the auto-graded assessment saved considerable time for the instructor, and offered immediate feedback for students.

Other Instructional Technology Tools

Several additional instructional technology tools were instrumental in creating an effective learning environment for this heterogeneous and fast-paced class. As with any online or hybrid course, clear organization and communication was vital to student success. The Learning Management System (LMS) was designed with a single weekly page containing links to all course content and assignments. Content included pdf notes and a video recording of each lecture. During the lecture, notes were delivered live using a smart screen and PDFAnnotator. By delivering notes on the screen, both face-to-face and online students could benefit by taking their own notes. Research has unequivocally shown the value of hand note-taking in conceptual learning,[9] so live notes were considered a must-have component of the course. Videos were recorded using Panapto, which included both a camera recording of the instructor station and screen capture of the pdf notes.

Assignments (except for Matlab Grader problems) and exams were delivered and submitted through the LMS. Two weekly assignments were distributed for peer grading, each worth 20 points. After the assignment deadline, students who completed the assignment gained access to the solutions. Each student was responsible for grading assignments from two classmates using a 0-4 scale for each problem using the rubric shown in Fig. 4 included in the syllabus. Grades from both student graders were averaged. The primary benefit from peer grading was the quick turn around time. Another advantage was that peer grading requires students to engage with the assignment a second time.



Points	Title	Details
0	Blank	No attempt made
1	Not Blank	Something is written, but it has very little to do with the problem, or is incomplete. Or, the answer is given without sufficient work to clearly show how the answer was obtained
2	Right Track	Proper application of the principles, but several or substantial errors in execution result in an implausible result.
3	Close	Everything looks right, but a *small error* results in an incorrect, but plausible, result.
4	Right	Proper principles and execution. Arrived at the correct result.

Figure 4: Peer Review Rubric

A final tool that was instrumental in the course was the interactive video conferencing, using Zoom. First, this allowed online students to participate in the live class. The Zoom broadcast included both voice and screen capture of the pdf notes. During the lecture, online students were muted but they could type questions using the chat feature. Zoom also facilitated online office hours which were held Tuesday through Thursday of each week (on the same days assignments were due.) During these sessions, students could connect by voice (through computer microphones or by calling in) or through the chat feature.

Student Performance and Feedback

Almost all students excelled in the class, with the lowest overall grade awarded at a B-. This did not indicate they found it too easy. Eighty-one percent of the students self-reported working over 10 hours per week for the class (10+ hours was the highest option) and over half of them mentioned either a high-workload or ‘intense’ pace to the course. Both of these are a reflection of the short duration for a 3 hour engineering course. However, a few students also suggested this format worked best for their busy summer schedules.

The Matlab assignments were given unlimited attempts, and all students were able to successfully complete them. Inspection of the average number of attempts for each problem provides a rough measure of difficulty. The first problem, a damped-mass-spring, was actually part of the Grader library, so it seems like a good baseline. Students, on average, needed just under four attempts to successfully complete the problem. Three of the custom problems had similar results: the Fourier series, the pendulum model comparison and the 5 DOF problem. The frequency plotting problem and the 2 DOF example appeared to be easier with less than 2.5 attempts per person. However, the undamped resonant simulation and the forced excitation with coulomb damping problem, both part of the week 2 assignment took almost twice as many attempts. If the course is offered again, it would be wise to revisit these. One frustration with Matlab Grader was the format for reports on student performance. For one, there was no option to create a single report at the course level, only the assignment level. Secondly, there is no collation of student attempts. One has the option to create a *.csv report with the information on either the latest attempt, the best attempt, or all attempts. If all attempts are selected, columns are organized in the order of submission. Finally, the report includes a lot of extraneous columns which obfuscate the relevant information. (As a case in point, the tests passes are in column H, separated by ‘tests passed’ in column L, by the student name and email address!) This is likely a non-issue when the Grader is integrated within the LMS.

Peer grading worked well, with consistent results among two peer graders; the average deviation was 0.70/20.00 points. However, feedback was generally limited to question scores. In the future, peers could be asked to provide more specific feedback. This would require deeper scrutiny by the grader and provide better information for the student being graded.

Other feedback from course exit surveys suggested that the instructional technology tools as a whole were effective in delivering course content and enhancing student learning. Thirteen of the fourteen respondents strongly agreed that they ‘learned a lot.’ All fourteen indicated that the professor was ‘approachable,’ which was encouraging given that six of them had no face-to-face contact. In short, the combination of instructional technology tools were more than sufficient to not only set clear expectations and deliver content, but also to establish a welcoming tone for the class.

Conclusions

This paper described the use of instructional technology tools to connect with both face-to-face and online students from varied backgrounds. The most notable of these was the Matlab Grader tool, a web-based environment for administering computer assignments. The platform is effective and easy to use as long as some attention is given in crafting the problem statement and establishing grading rubrics. Two additional tools that were instrumental for the course included the combination of video recording (Panopto) with live notes using PDFAnnotator on a smart screen. In addition, the video-conferencing app, Zoom, enabled live participation in the class from online students and online office hours. Finally, the LMS provided an adequate environment to collect course resources, communicate information, and exchange assignments.

References

[1] B. S. Bloom; J. T. Hastings: and G. Madaus. *Handbook on formative and summative evaluation of student learning*. McGraw Hill, New York, 1971.

[2] Thomas R. Guskey. Lessons of mastery learning. *Educational Leadership*, 68(2), 2010.

[3] Linda B. Nilson. *Teaching at Its Best*. John E. Wiley & Sons, San Francisco, CA, 4 edition, 2016.

[4] Henry Luo; Anthony Robinson; and Jae-Young Park. Peer grading in a mooc: Reliability, validity, and perceived effects. *Online Learning Journal*, 18(2), 2014.

[5] Matlab grader documentation. Accessed: 2020-02-03.

[6] Brenda Cheang; Andy Jurnia; Andrew Lim; and Wee-Chong Oon. On automated grading of programming assignments in an academic institution. *Computers in Education*, 41, 2003.

[7] Kevin Mathews; Thomas Janicki; Ling He; and Laurie Patterson. Implementation of an automated grading system with an adaptive learning component to affect student feedback and response time. *Journal of Information Systems Education*, 23(1), 2012.

[8] Cindy Waters; Taher Abu-Lebdeh; Messiha Saad; and Steven OneYear. Preparation, attendance, and note-taking, how to promote student buy-in. *2006 ASEE Annual Conference & Exposition*, June 2006. https://peer.asee.org/1410.

[9] Pam A. Muller and Daniel M. Oppenheimer. The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychological Science*, 25(6), 2014.

[10] Edward H. Perry. The impact of online lecture notes on learning outcomes of beginning thermodynamics students. *2007 ASEE Annual Conference & Exposition*, June 2007. https://peer.asee.org/2035.

[11] Patrick Cunningham; Shannon Sexton; and Julia Williams. Why come to class? posting notes from tablet pc lectures. *2009 ASEE Annual Conference & Exposition*, June 2009. https://peer.asee.org/5414.

[12] Benjamin Bloom. *Taxonomy of educational objectives; the classification of educational goals*. Longmans, Green, New York, 1956.

[13] Singiresu S. Rao. *Mechanical Vibrations*, volume 6. Pearson, Harlow, 6 edition, 2018.