New Orleans JAZZED about Engineering Education

Paper ID #16496

Teaching an Undergraduate Introductory MATLAB Course: Successful Implementation for Student Learning

Dr. Kyle Frederick Larsen, Eastern Washington University

Dr. Larsen currently teaches mechanical engineering at Eastern Washington University. He received his B.S. and M.S. degrees in mechanical engineering from California State University Sacramento and his Ph.D. in mechanical engineering from Brigham Young University.

Dr. N.M. A. Hossain, Eastern Washington University

Dr. Hossain is an Associate Professor in the Department of Engineering and Design at Eastern Washington University, Cheney, WA. His research interests involve the computational and experimental analysis of lightweight space structures and composite materials. Dr. Hossain received M.S. and Ph.D. degrees in Materials Engineering and Science from South Dakota School of Mines and Technology, Rapid City, South Dakota.

Prof. Martin William Weiser, Eastern Washington University

Martin Weiser is an Assistant Professor in the Engineering and Design Department at Eastern Washington University. He earned his BS in Ceramic Engineering from the Ohio State University and his MS and PhD in Materials Science and Mineral Engineering from the University of California at Berkeley. He then joined the Mechanical Engineering department at the University of New Mexico where he taught Materials Science, Thermodynamics, Manufacturing Engineering, and Technical Communication. Martin then joined Johnson Matthey Electronics/Honeywell Electronic Materials where he held positions in Technical Service, Product Management, Six Sigma, and Research & Development. He is an inventor on a dozen patents and patent applications and has published over 30 papers and book chapters on topics including ceramic processing, Pb-free solder development, experimental design, and biomechanics. His current research focuses on rocket propellant characterization, fin flutter, and heat transfer.

TEACHING AN UNDERGRADUATE INTRODUCTORY MATLAB COURSE: SUCCESSFUL IMPLEMENTATION FOR STUDENTS LEARNING

Kyle Larsen, Awlad Hossain and Martin Weiser
Department of Engineering and Design
Eastern Washington University

Abstract

In our institution, we offer a one-quarter long MATLAB class for the Mechanical Engineering (ME) and Mechanical Engineering Technology (MET) curriculum. This course teaches computational methods to solve engineering problems using the program MATLAB. The coursework involves teaching fundamental programing along with engineering principles to build the concept, analyzing simple structural problems using matrix algebra and then solving a wide variety of engineering problems dealing with statics, dynamics, fluid mechanics, and heat transfer. Students enrolled in this class solve a variety of problems by setting them up analytically then programing them and solving them in MATLAB. As we are in a quarter system, it is challenging to solve multidisciplinary complex engineering problems in regular class lectures. Therefore, students enrolled in this class are required to program a variety of engineering problems within a short time. These problems must have adequate engineering complexity and themes conveying interesting knowledge or technical concepts, and at the same time be concise enough to be completed during the course. While a course in MATLAB could be a common offering in many universities, the authors of this paper presents the pedagogical approaches undertaken to successfully implement the course objectives to the undergraduate engineering students. The topics and techniques applied to teach different topics of engineering problems to enhance students learning outcomes are addressed in this paper. The paper also presents how different topics taught in this class fulfill the targeted course objectives, which are mapped with ABET Engineering criteria.

Introduction

At Eastern Washington University (EWU) the faculty continually strives to improve the engineering program; this was a big motivation for this study. The purpose of this paper is to provide information that will help to optimize a curriculum in MATLAB. Engineering universities are often trying to improve their curriculum, while at the same time attempting to not increase the number of credit hours for their degree. Therefore, optimizing the curriculum by teaching students how to program many different types of engineering problems in a relatively short time is very important.

A big challenge in teaching a MATLAB course is the fact that many of the students haven't had a lot of math or engineering prior to taking the class. This is because most of the students taking the class are in their sophomore year. Therefore, it is essential that the problems being programed are carefully setup in a step by step manner in which the students can understand. Furthermore, the math and engineering used in the problems needs to be presented in such a way

that it is very understandable to the students. This requires starting with simple problems and then increasing the complexity of the problems as the course continues. Whether the class is taught as a 15-week semester course or a 10-week quarter course, the limited time makes it very difficult to cover all the material. This is especially true for schools that teach only a generic MATLAB course that is not intended for a specific engineering discipline and is therefore supposed to cover a little bit of everything.

The purpose of this paper is to provide information that will help optimize the curriculum in a MATLAB course and provide examples that are effective in teaching the students. This paper accomplishes this by studying some of the methods used by other colleges and universities, along with a discussion of the techniques used by the authors that have been found to be effective in teaching a course in MATLAB. Additionally, the importance of MATLAB as a pre-requisite for two different courses taken later by the students: Engineering Numerical Methods and Robotics are also addressed.

Review of the Teaching Methods used by Other Schools

Many engineering institutions use MATLAB as a technical computing language. Teaching and training students to become competent and efficient programmers continues to be a challenge. Engineering faculty from Armstrong Atlantic and Georgia Southern University[1] implemented several pedagogical approaches to address this challenge, including the use of Virtual Learning Environments (VLEs). This work presented the use of the VLE – MATLAB Marina as a supplement in computing for engineering courses.

Using a VLE as a supplement to the classroom study can greatly help the students to learn the material quicker. This could even be incorporated with an online tutorial that covers both programing techniques and the engineering concepts necessary to solve various problems in engineering.

At Grand Valley State University[2] the instructors used MATAB to learn different computing concepts in an inverted classroom. The inverted classroom showed 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.

At the University of Cincinnati[3], a series of courses called Engineering Models I and II are offered to help students make the connection between the math and science courses typically taken during the first year to engineering applications. This is done by introducing students to

computing, through MATLAB, to allow the students to work on a variety of complex problems. In an effort to improve the learning of students in the course, three sets of two teaching assistants (TAs) tested a different educational method in the lab portion of the courses. These methods were identified by the TAs after teaching students for several weeks and analyzing the common pitfalls encountered by the students. In the first method, the TAs focus on complimenting students on their performance and encouraging them to break up the problem into smaller, more manageable steps. This fosters an engineering mindset that assists the students intimidated by programming, or helps those that struggle with starting the problem. The next method tested involved the TAs asking questions about the coding concepts the students were using to encourage a deeper level of thought and understanding with the assignment. Lastly, the TAs questioned the students on the commands used in the lab activity and provided some initial guidance in starting the code for the assignment.

It has been shown that students learn best by visuals and examples[4]. These methods include using pictures, and realistic problems to show the students how to apply the concepts with the theory. If a student can visually see how all the pieces fit together, they will be able to master the subject and apply it to real world applications.

Projects with real world applications are another tool used in teaching MATLAB. Many schools use projects to teach their students a variety of different subjects.

At Northern Illinois University[5] they used MATLAB to design and simulate a sun tracking solar power system. The simulation consists of four modules: solar tracking cells, signal conditioning circuit, controller, and motor. The simulation provides an excellent platform for undergraduate engineering technology students to study the design and theory of a sun tracking solar power system using the MATLAB platform.

The University of Maryland, Eastern Shore[6] introduced an undergraduate level in-class project exercise of solving 2-3 extensive problems that required developing MATLAB subroutines and SIMULINK. MATLAB and SIMULINK were used for modeling to present the method of solution and grasp the theoretical ideas in practice to use it further for nonlinear application of a real-world problem. The authors of this paper presented a study of students' assessment, grasping capabilities and challenges to make it thorough and rewarding for undergraduate research experiences in Systems Dynamics & Controls and Aerospace Engineering.

The University of Cincinnati[7] has used MATLAB to teach different components of electric machines classified as Direct Current (DC) and Alternating Current (AC). Due to the time constraint in one quarter, seven (7) important AC and DC laboratory assignments pertaining to electric machines are selected in one required course. Using MATLAB enables the in-depth teaching of these topics during the ten-week quarter period. MATLAB was used as a tool that made it possible to teach important electromechanical topics in electrical engineering curriculums, during a limited time, in one quarter or one semester.

Use in Additional Courses

At EWU the Mechanical Engineering (ME) students take a course in Engineering Numerical Methods while both the ME and Mechanical Engineering Technology (MET) students take a course in Industrial Robotics. Both the MATLAB course discussed here and a computing course in the C language are accepted as prerequisites for these two courses. This is necessary because EWU has a significant number of transfer students that take a course in C at the community college or another University.

The Engineering Numerical Methods class covers a wide variety of methods used by Engineers ranging from basic tools such as least squares regression through solutions to partial differential equations. The course addresses the background mathematics and then requires the students to set up and solve an engineering problem both starting from the background math and using provided functions in both MATLAB and Excel. In most cases the students may solve the problem from the background math using their choice of programming language, occasionally choosing to use C++ or even Python. The use of MATLAB alongside Excel has been very well received by the students, even those who took a different programming language. Most of them pick up the MATLAB syntax quite quickly, as we solve more difficult problems and find the immediate feedback on both mistakes and successes very rewarding. It is understood that in their future work they will most likely use the functions provided in various software packages, but the goal is to make sure they understand how to use them. However, some of the students have reported back that they have used what they learned to quickly solve some fairly difficult problems using MATLAB or Excel when their employer did not have more sophisticated software such as ANSYS or SolidWorks Simulation.

The Industrial Robotics course is taken by both the ME and MET students at the same time and focusses on developing innovation[8]. The course starts with using a Robix kit to introduce the students to programming a basic robot. It then moves on to programming industrial robots (ADPET Cobra 600 or FANUC LR MATE 200ir robots) to perform a more complex task of the student's choice as featured on the EWU Robotics channel on YouTube. It finishes with a section programming a PLC to run the bulk delivery component of a model industrial plant. Each of these systems uses a different, proprietary programming language which the students must learn. However, we have found that the students who have taken MATLAB, rather than C++, pick up the new languages significantly faster since these industrial languages have the same basic structure as MATLAB.

Example Problems in the MATLAB Course

As mentioned earlier, most of the students do not have an extensive math, engineering and programing background. The prerequisite for our MATLAB course is Pre-Calculus; however, we teach various engineering concepts where the math ranges from Pre-Calculus through Differential Equations. Therefore, it is necessary to start with the basics, using simple problems, and increasing their complexity as the course progresses and encouraging our students with a deeper mathematics background to use their knowledge to make the problem more realistic.

We require and use one text book as a reference for the course along with examples from the internet and other books. Examples we use to teach the basics in our MATLAB course relate to the following subjects:

1. Built in functions.
2. MATLAB matrices and arrays.
3. User defined functions.
4. Loops, logical functions and selection structures.
5. Graphing and plotting.
6. Matrix algebra.
7. User-controlled input and output.
8. Symbolic Mathematics.
9. Graphical user interface.
10. Simulink.

A simple example to teach students about "*built in*" functions is to use a program to generate a sine wave with noise. This problem combines the random number function along with the *sine* function to model random noise such as `noise = rand(size(t));` `y = 5*sin(2*t) + noise.` The output is then plotted showing the results in figure 1.

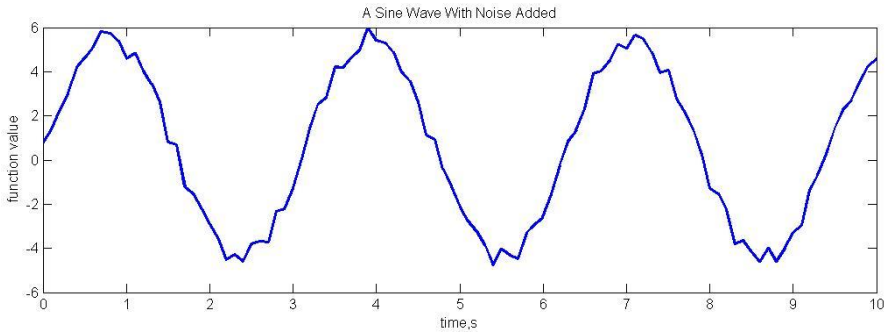

Figure 1 Sine Wave with Noise Example

This same random noise problem is then used to show the students how SIMULINK works. This is modeled using the set of blocks shown in figure 2. Here a sine wave function block and a uniform random number block are added along with three scopes. One scope shows the *sine* wave, the second shows the random noise, and the third shows the sum of the two which is the *sine* wave with the noise added.

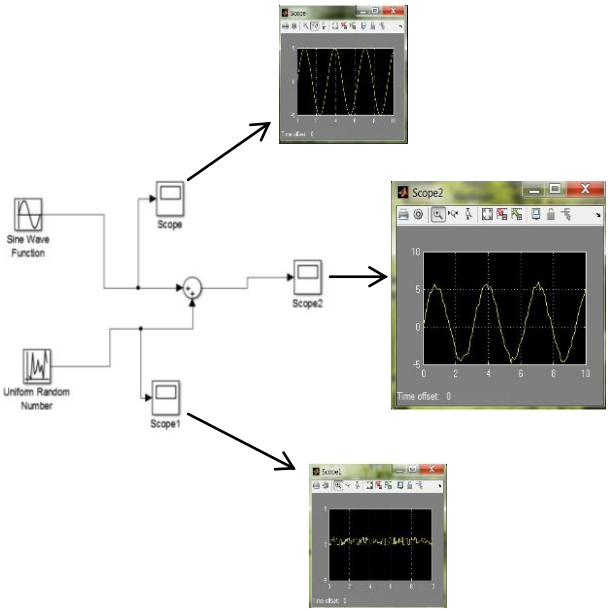

Figure 2 Sine Wave with Noise Solved using Simulink

Another problem useful in teaching the students about loops and logical functions is that of calculating the terminal velocity of a particle falling in a fluid. In this example, many iterations must be made using different Reynolds numbers as the velocity changes to calculate the varying drag coefficient until the terminal velocity is reached. Since the students at this stage have not had enough concepts on fluid mechanics, a detailed handout showing all the background of the mechanics of this problem is given. Figure 3 shows the diagram of the force balance and the apparatus.

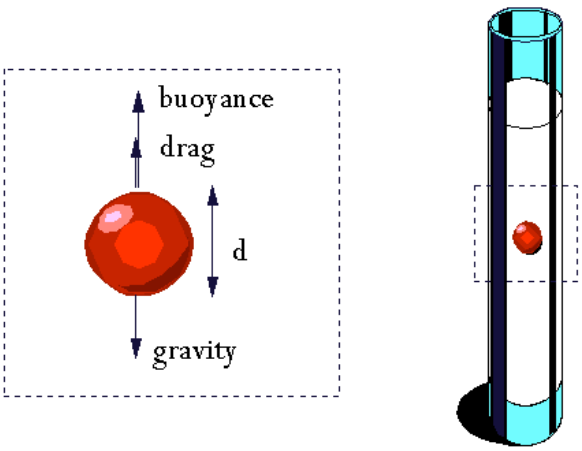

Figure 3 Terminal Velocity Problem

An example of a simple statics problem used is with a crate held in equilibrium as it rests on a frictionless inclined plane making an angle θ with the horizontal. The objective of this problem is compute the tension in the cable and the normal force at the surface for θ at 5˚increments

increasing from 0˚ to 90˚. The tension in the cable and the normal force is then plotted. Figure 4 shows the configuration of the crate on the inclined plane.

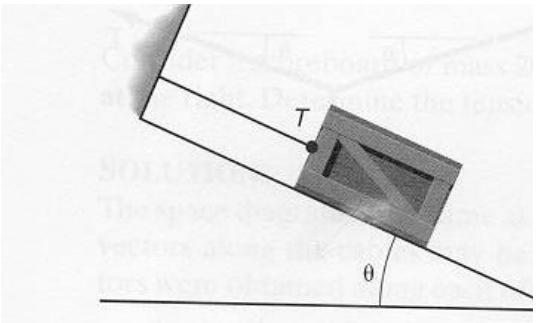

Figure 4 Crate on Incline Plane Problem
(Statics by R.W. Soutas-Little and D.J. Igman)

In addition to these examples many others are used to cover the topics mentioned earlier. These include 3-D graphing, graphical user interface problems including a commonly used READY_AIM_FIRE[9] example, an economics problem with compound interest, and circuits problem requiring matrix algebra to solve for the voltages and currents.

One of the more complicated problems that is solved in the introductory MATLAB course is the trajectory of a projectile with air drag. This is a second order, inhomogeneous, partial differential equation with non-constant coefficients. Since the prerequisite for the course is only pre-calculus this is well beyond what students have experienced in the past. In fact, it is beyond the scope of the mathematics requirement for the ME degree which terminates with ordinary differential equations. In spite of this, most of the students do very well with the project and leave the course understanding a bit of the true power of MATLAB. Based upon their reports, they are far better prepared when they take the rest of their calculus and differential equations classes.

The keys to making this project work are not to present the true complexity of the problem at the start, but to clearly present the students with the set of finite difference equations to be solved, and to break it into a series of projects that build to the true complexity of the problem. Most of the students have been introduced to the trajectory problem in their college physics class where they solve for the optimal launch angle without air drag of 45^o above the horizon for maximum distance. The first project in this sequence is for them to plot the trajectory of the projectile as a function of the launch angle starting from the basic equations of $F_x = ma_x = 0$ and $F_y = ma_y = -mg$ which are then used to calculate the velocity and position in both the x and y directions via the recursion formulae $v_{j+1} = v_j + \Delta t\, a_j$ and $d_{j+1} = d_j + \Delta t\, v_j$ based upon Euler's method. The students have to read in the initial velocity, angle, and Δt using the ***input*** function. They are given a great deal of latitude in their choice of Δt and the details of how they program this project. One of the biggest issues that arises is that they pick a value of Δt which results in the altitude being less than 0 after 1 time step so that their while loop only runs once,. i.e. $\Delta t\, g > v_{y,j+1}$.

The next step is to have the students add the drag force to their program and calculate the acceleration at each time step. The drag force is proportional to the velocity squared and opposes the velocity. This latter fact generally causes problems in the vertical direction as the projectile passes apogee. They are told about the ***sign*** function in MATLAB, but a significant number confuse it with the ***sine*** function and end up with some rather interesting, but not particularly useful results. The third project uses ***xlsread*** to read in data for several different projectiles and then read in the projectile type and launch parameters using the ***input*** function. They also have to plot the trajectory and velocity as animations on fixed axes that are a bit larger than the maximum of the *x* and *y* values rounded to a reasonable multiple such as 100m.

Often times we have the students calculate the projectile motion using Excel first, both without drag and then with drag so they can get a better understanding of the math. Then we have them use MATLAB to do the same thing, showing how much easier and versatile it is to do it with MATLAB. Figure 5 shows a plot of projectile motion without and with drag using Excel and figure 6 shows a similar plot using MATLAB.

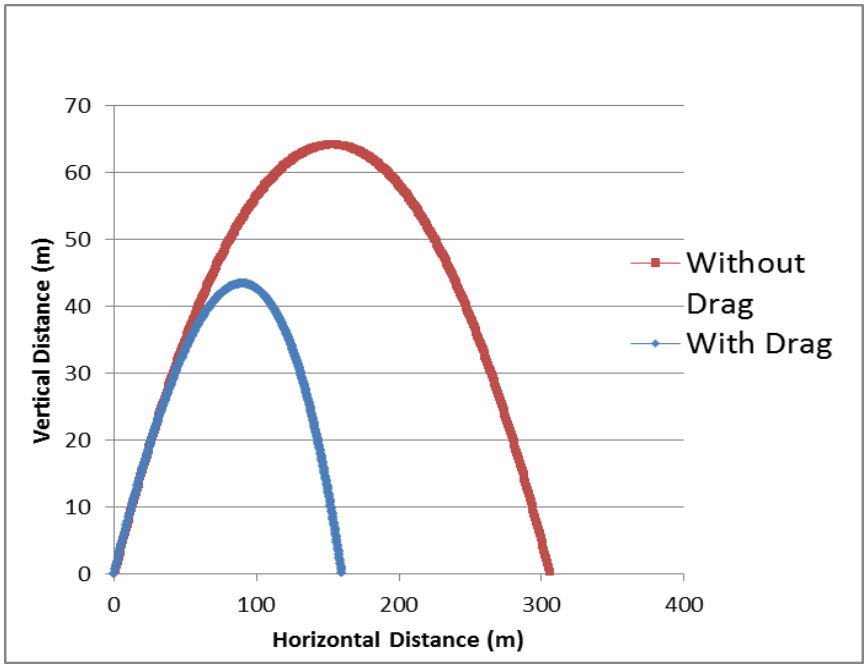

Figure 5 Projectile Motion using Excel

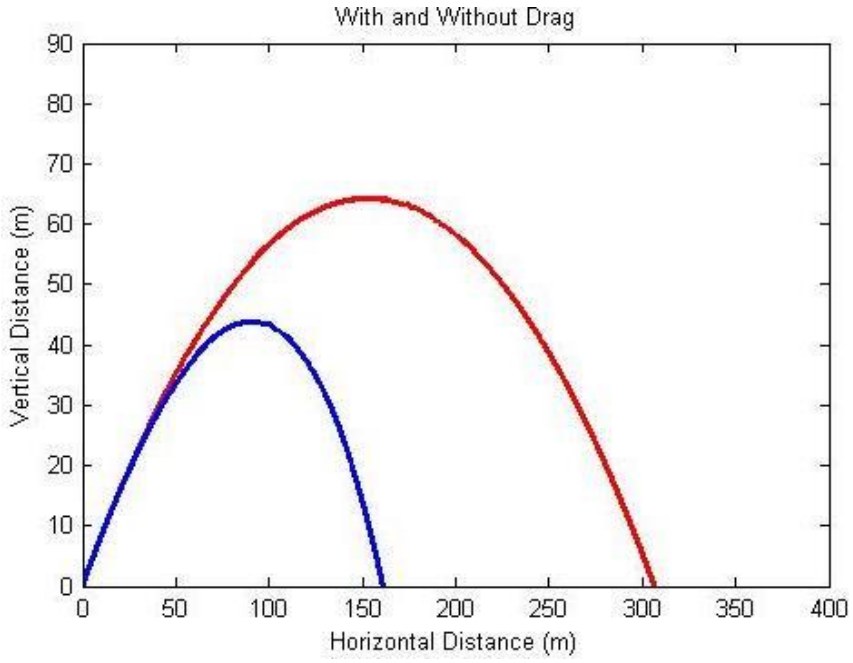

Figure 6 Projectile Motion using MATLAB

In the Engineering Numerical Methods course this project is generally extended by converting the projectile to a large model rocket where there is an initial motor thrust to start the flight that depends upon time as well as the deployment of both a drogue parachute at apogee and a main parachute at a lower elevation that dramatically changes both the C_d and projected drag area. The students that have completed the projectile project in the introductory MATLAB course generally do not have too many issues with this extension of the problem. However, nearly everyone has issues in converting from the simple Euler's method to the more complex Runge-Kutta 4 (RK4) method, particularly since it is the relationship between acceleration and velocity for drag that is not as straight forward.

Evaluation of the Course

For the ABET evaluation of the ME program, the student performance in our MATLAB class is specifically assessed against ABET criterion 3k "*An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice"* for the **MET** program a very similar ABET criterion 3a *"An ability to select and apply the knowledge, techniques, skills, and modern tools of the discipline to broadly-defined engineering technology activities"* is used. As of the Fall 2014 term, we have defined a Performance Indictor (PI) for each of the criterion, which are listed in Table I.

Evaluation of student performance versus the performance indicators is somewhat subjective as used for the ME/MET programs at EWU. The instructor sorts the chosen assignments into four categories based upon the rubric. This may or may not be directly tied to the student's score on the assignment since most of the assignments in this course also include writing a short report about how they did the assignment and presenting their results. It is not uncommon for the

students to demonstrate that they understand the use of the tool as defined by the rubric, but lose a significant number of points on the assignment.

Table I Performance Indicator for ABET Criterion 3k (ME) and 3a (MET).

Performance Indicator	Unsatisfactory **(Usat)**	Below Average **(BA)**	Satisfactory **(Sat)**	Exemplary **(Ex)**
Student must be able to use a mathematical software to analyze a problem.	Student did not know how to use any math software.	Student could navigate through the software but is unable to use it to solve.	Student could navigate through the software and is able to use it to solve a simple problem.	Student has a good understanding of the software and is capable of solving complex problems.

This PI was used to assess student performance during the Fall 2014, Winter 2015, and Summer 2015 terms when the course was taught by 3 different instructors. As a result three related, but different problems were used for the assessment as listed below.

- Fall 2014 – Use the Symbolic Toolbox to solve systems of linear equations composed of up to cubic polynomials and find the integral and derivative of such polynomials. The *ezplot* function was then used to plot the polynomial and its integral and derivative as subplots. Polynomials were chosen since the students were familiar with them and the expected results so they could focus on using the symbolic toolbox.
- Winter 2015 – The function $x^2\sin(x)$ was numerically integrated from 7 to 12 using a Riemann Sums. The initial value of Δx was 0.5 and the program had to decrease Δx until the Riemann Sum was within 1% of the true area as determined using the Symbolic Toolbox.
- Summer 2015 – Determine the terminal velocity of a sphere falling in a viscous fluid as described earlier in the paper. This assignment focused on using loops and branching to calculate the velocity as a function of the Reynolds number and plot it.

Although these problems are different, they all use MATLAB to teach the use of software to solve a problem. However, we recognize that the winter 2015 assignment is probably the most difficult (Symbolic Toolbox, loops, and evaluation of when the results were good enough) while the summer 2015 assignment was the easiest (loops with different ranges of Re), but it was taught earlier in the term than the other assignments. The results from this assessment are presented in Table II and graphically in Figure 7.

Table II Student Performance vs. the Rubric for the Three Terms.

	MET Students				ME Students				
	Usat	BA	Sat	Ex	Usat	BA	Sat	Ex	Total
Fall 14	0	0	3	0	0	3	11	7	24
Winter 15	2	2	1	0	0	4	4	7	20
Summer 15	0	1	1	2	0	0	3	11	18
Total	2	3	5	2	0	7	18	25	62

The student performance levels were then assigned scores of 0, 1, 2, and 3 from unsatisfactory to exemplary. The weighted means and standard deviations were first calculated for each of these different classes for both the MET and ME students, and then they were calculated by combining the data for all three classes for the MET and ME groups. The results are shown in Table III.

Table III Mean and Standard Deviation of the Student Performance versus the Rubric

MET Students

	Fall 14	Winter 15	Summer 15
Count	3	5	4
Average	2.00	0.80	2.25
SD	0.00	0.75	0.83

ME Students

	Fall 14	Winter 15	Summer 15
Count	21	15	14
Average	2.19	2.20	2.79
SD	0.66	0.83	0.41

	MET Students	ME Students
Count	12	50
Average	1.58	2.36
SD	0.95	0.71

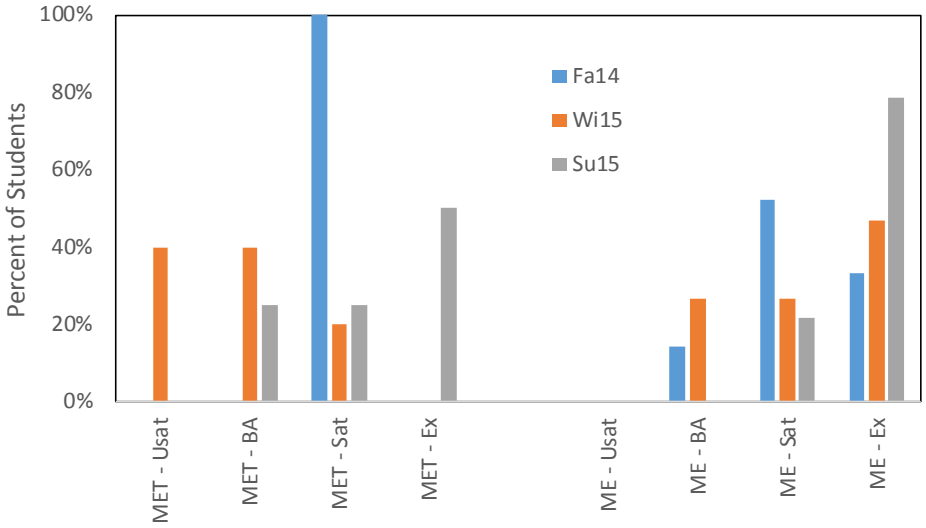

Figure 7 Results of the Assessment versus PI 3k (ME) and 3a (MET).

Examination of Figure 7 and Table III shows that in general the MET students did not perform as well as the ME students on these assignments. This is expected since most of the ME students are more comfortable with mathematics and some have taken courses beyond Calculus II which is the last mathematics course required for the MET students. Spending a little more time explaining the math would most likely help to alleviate this difference. The second thing is that the assessment results are quite variable from term-to-term, which is a result of the type of problem that was assigned and the expectations of the instructor. Analysis of these results indicate that we need to better define the type of problem used for this assessment and how it is to be assessed to allow better use of the tool. Having each instructor use the same problem in their class to excess the students learning would help to better achieve this outcome.

Conclusions and Recommendations

We have found that by having the students solve many different problems, first starting with the simpler ones and then increasing their complexity, is the best way for them to learn MATLAB. In addition, even though most of the students haven't had much math, engineering or programing prior to taking the course, our experience has been if the problems are well-organized and explained to the students, they can better grasp the material. Having students solve various problems not only teaches the students how to program and use MATLAB, but also teaches them more about engineering and math in general which prepares them for their future classes.

We have also found that our MET students have a more difficult time then our ME students in learning to solve problems in MATLAB. This was indicated by the results of our assessments by the problems we used in our classes. This is probably because MET students are generally less comfortable with math than ME students. While the results of our assessment methods did support this conclusion, the assessment of our students learning could be better improved by using at least one problem that is the same in every class taught. However, there is still some variability even when this is the case due to the different expectations and interpretations of the instructor when performing the assessments.

Finally, we have provided a number of good examples that can be used in an introductory MATLAB course along with our recommendation for those subjects that should be taught. These include the following:

1. Built in functions.
2. MATLAB matrices and arrays.
3. User defined functions.
4. Loops, logical functions and selection structures.
5. Graphing and plotting.
6. Matrix algebra.
7. User-controlled input and output.
8. Symbolic Mathematics.
9. Graphical user interface.
10. Simulink.

References

1. P. T. Goeser, W. Johnson, S. L. Bernadin, and D. A. Gajdosik-Nivens, "Work-in-Progress: The Impact of MatLab Marina - A Virtual Learning Environment on Student Learning in a Computing for Engineers Course", ASEE Annual Conference and Exposition, 2013.
2. R. Talbert, "Learning MATLAB in the Inverted Classroom", ASEE Annual Conference and Exposition, 2012.
3. J. J. Heeg, K. Flenar, J. A. Ross, T. Okel, T. A. Deshpande, G. W. Bucks, and K. A. Ossman, " Effective Educational Methods for Teaching Assistants in a First-Year Engineering MATLAB Course", ASEE Annual Conference and Exposition, 2014.
4. M. Cook, "Teaching with Visuals in the Science Classroom," Science Scope, vol. 35, no. 5, pp. 64 – 67, January 2012.
5. L. Guo, J. Han, and A. W. Otieno,"Design and Simulation of a Sun Tracking Solar Power System", ASEE Annual Conference and Exposition, 2013.
6. R. Sharma, and A. Nagchaudhuri, "Implementing Problem-Based Learning Projects to Synthesize Feedback Controllers Using MATLAB/Simulink and Students Assessment", ASEE Annual Conference and Exposition, 2014.
7. M. Rabiee, "Using MATLAB to Teach Electric Energy Courses", ASEE Annual Conference and Exposition, 2012.
8. J.K. Durfee, D.C. Richter, M.W. Weiser, N.M.A. Hossain, and H.S. Saad, "Using Course Projects to Infuse Innovation Throughout the Undergraduate Experience in the Engineering and Engineering Technology Curriculum", paper 8984, 121st ASEE Annual Conference, Indianapolis, IN June 2014
9. Moore, Holly. 2015. MATLAB for Engineers 4th edition. Pearson.