Assessment of Student's Programming Skills in a Dynamics Systems and Control Course

Dr. Arjumand Ali, Grand Valley State University

Dr. Ali is an Assistant Professor in the School of Engineering at Grand Valley State University. She received her Ph.D. in Mechanical Engineering from the University of Wisconsin Milwaukee in 2013. Her areas of interest and expertise include Dynamics, Controls, Vibrations, Mathematical Optimization, Multilevel Algorithms and Game Theory. She has taught courses in statics, dynamics, vibrations, kinematics, dynamic systems and controls.

Dr. Ryan W Krauss, Grand Valley State University

Dr. Krauss received his Ph.D. in mechanical engineering from Georgia Tech in 2006. His research interests include modeling and control design for flexible robots, feedback control, and microcontroller-based implementation of feedback control systems. In addition to the freshmen introduction to engineering design course, he has taught courses in mechatronics, controls, vibrations, dynamics and robotics as well as senior design.

Assessment of Student's Programming Skills in Dynamics Systems and Control Course

Abstract

This work-in-progress paper presents an assessment of the programming skills of students who have completed a junior-level dynamic systems and control course. At Grand Valley State University, this course is taken by both Mechanical Engineering (ME) and Product Design and Manufacturing Engineering (PDM) majors and is taught by two different professors. The course content in the two versions is very similar but two different programing languages are used. ME students learn Matlab and Simulink whereas PDM students learn Python to simulate systems. There is a lab associated with the course. In lab, both ME and PDM students use Arduino microcontrollers for dynamic systems and control experiments. However, tasks such as simulation or data analysis are done in Matlab or Python respectively.

This paper seeks to assess the programming skills of students in both versions of the course as well as see if there are any differences in student learning between Matlab and Python. The students were given two programming assignments as part of a take-home exam at the end of the course. Students are asked to design a controller for stability, predict the response and simulate the closed-loop step response of a system.

Introduction

Computer programming plays an important role in any dynamic system and control courses. Study of dynamic systems involve modeling and analysis of system to different inputs. When analyzing a complex (higher order) dynamic system, hand calculations would be tedious and is not a feasible option and some analytical tools are necessary. The students today are computer literate and use graphing calculators very effectively for solving their homework problems. When the students were asked to solve a simple, second-order, homogeneous differential equation without any review of differential equation in class, the students just used WolframAlpha to find their solution. They do not really understand the mathematical concepts that they manipulate. The analytical tools available to engineers these days are very powerful and the students must learn to use them but after learning the concepts “the hard way”. The analytical power of the tools that they use do not require the understanding to produce a correct solution. Computer programming on the other hand require the students to think and write their code to obtain the solution. Algorithms such as Euler’s Method and Runge Kutta method can be used to solve differential equations using any programming language like Matlab or Python. These tools can also be used to simulate the responses of such systems, and also to analyze experimental data.

Dynamics Systems Modeling and Control is a required course in any mechanical engineering program. This course is a challenging course for the students as well as for the instructors. Students often find the material too abstract, too mathematical, and difficult to master when it is presented in lectures [7]. Since this course is very abstract and mathematically intense, it is not easy to teach as well. The instructors find it hard to keep students engaged all the time. When class consists of an instructor simply talking to the students, they are not focused and their

attention is more likely to drift, especially if the class period is long [5]. Some researchers believe that there were no significant differences between traditional classroom teaching and the use of technology in teaching [3], while others think that students learn and retain more when they can apply the concepts and not just read or hear about them [4]. Different methods have been tried to improve students learning and engagement in this course. Some instructors have used extensive simulations [1],[7],[8] and haptics [9] to enrich this courses. Others [2],[6],[10] have tried physical experiments to give more hands-on experience to the students.

At Grand Valley State University, this course is one of the most challenging junior level courses as it combines dynamic systems, controls and mechatronics content into one single course. This is a required course for ME and PDM students. The prerequisites for this course for ME students are Dynamics, Differential Equations, Circuit Analysis I and Introduction to Digital Systems. There are some electives that needs this course as a prerequisite which includes Robotics, Vibrations and Manufacturing Controls. No other required course needs this course as a prerequisite in ME program. The prerequisite for this course for PDM students are Differential Equations and Circuit Analysis. A follow up course is Manufacturing Controls which is a required course in PDM program. Also Robotics is an elective for PDM program too which needs dynamic systems and control as prerequisite. The ME section of the course used Matlab and Simulink whereas the PDM section used Python along with the python-control module and the Jupyter notebook interface. It should be noted that the students have been introduced to Matlab/Simulink and Python in this course for the first time and they have not used them in any of the previous courses. This paper seeks to investigate whether or not there is any difference in students' skills related to programming tasks associated with dynamic systems and control if they use Matlab and Simulink as opposed to Python.

Softwares Description

Simulink is a graphical editor for modeling, simulating and analyzing dynamic systems. It consist of a library of blocks (Figure 1) representing the models.

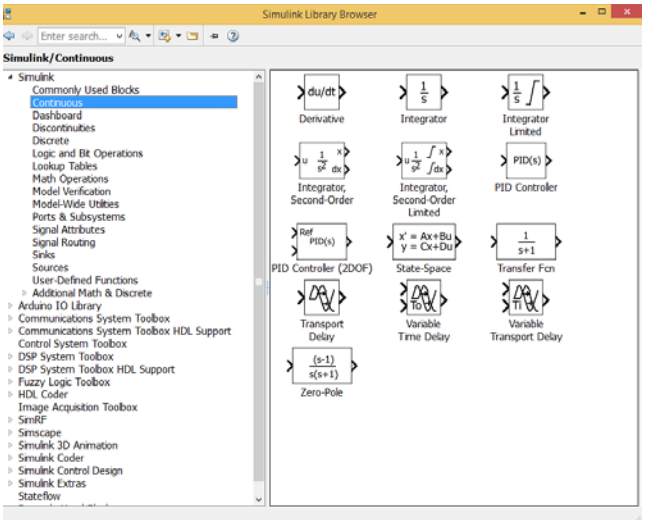


Figure 1- Simulink Library of Blocks

The students need to build a model of the dynamics system (a differential equation) by using blocks for integration, summation and multiplication. So they are not blindly inputting the differential equation but rather creating a model for it and then simulate it. A simple example of a spring mass damper system shown in Figure 2. The Simulink model of this system and the response to a step input is also shown in Figures 3 and 4. There is also a transfer function block and a state space block in Simulink library which can be used to obtain the response of the system as shown in Figures 5 and 6.

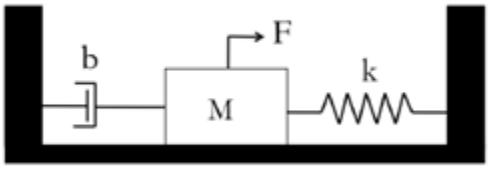


Figure 2- Spring mass damper system

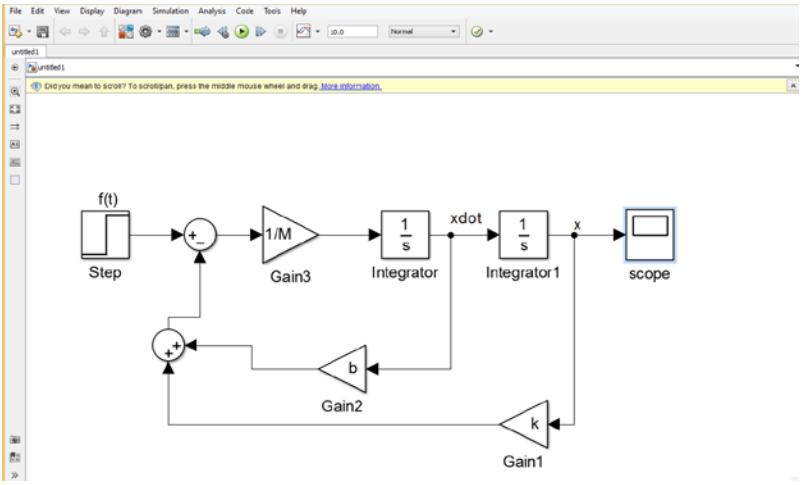


Figure 3- Simulink model for the spring mass damper system

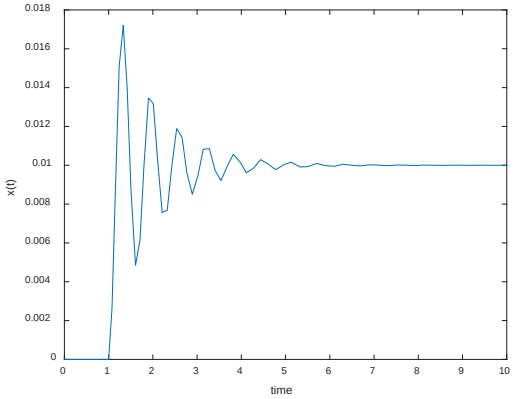


Figure 4 System response for the spring mass damper system

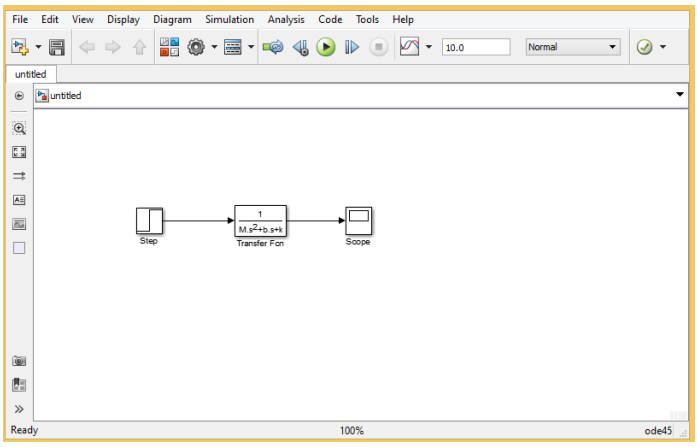


Figure 5 Simulink model using transfer function block

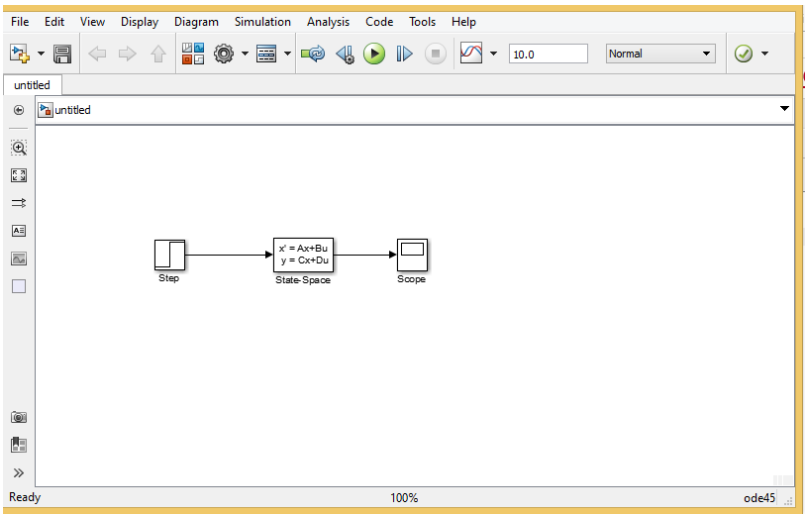


Figure 6 Simulink model using state space block

The PDM section used Python along with the python-control module and the Jupyter notebook interface. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics [11]. Python is open-source and its syntax is very clean. The python-control module allows dynamic systems to be represented in transfer function and state space form and facilitates simulation of dynamic systems and design of controllers in both the time and frequency domains. The Jupyter notebook provides a web interface that allows students to type code into cells and then see any results immediately in an output cell. The notebook merges the traditional editor and command line into one interwoven interface, providing students immediate feedback on each chunk of code they enter. Additionally, plots can be shown immediately following the cells that generates them, making it easier to associate the figure with the corresponding code. Figure 7 shows a screenshot of the Jupyter notebook.

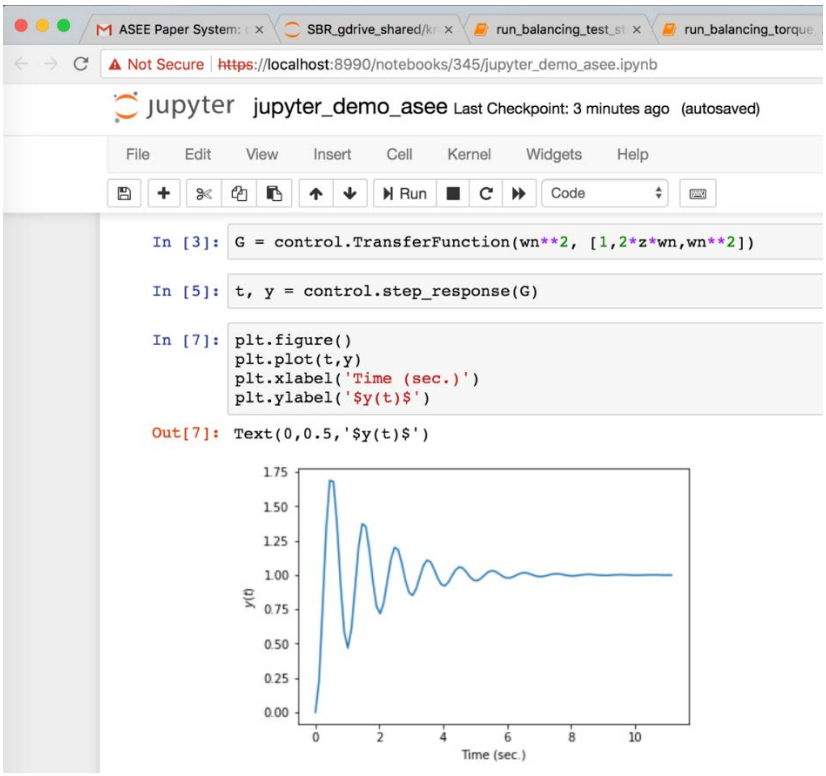


Figure 7 Screenshot of the Jupyter notebook showing the step response of a second-order system

Lab Experiments

Both versions of the course include a lab, both ME and MPD students used C to program the microcontroller for lab activities. The Arduino software is used to run experiments and collect data .The first lab taught in ME section was an introduction to Matlab and in PDM section was an introduction to Python. A tutorial on Simulink is also presented as a follow up lab in the ME section. The students then in the later labs were also asked to simulate the response of the systems and to compare the simulations with experimental results.

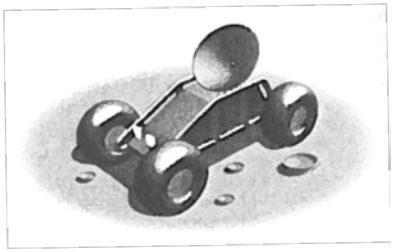
Assessment

The assessment of student's programming skills was done by comparing student learning outcomes between the two versions of the course through common final take home exam questions. This was the only way the authors think of to assess students' abilities to model and simulate complex systems using different softwares. The in class final exam is two hour long and is comprehensive. There is a lot of material to test the students' global comprehension of the material. Therefore the authors decided to make a separate take home exam which should be assigned two days before the final in class exam. Students were asked to choose the control gain that would give a stable system, predict the response of the system (using a reduced order model) and to simulate the actual response of the closed loop system. Two questions were assigned as take home exam to the students in both versions of the course. First problem assigned is a fourth

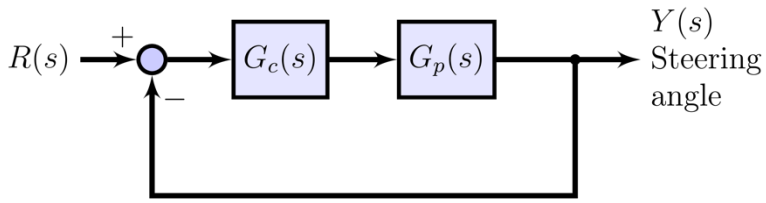
order system and second problem is a third order system. To simulate the system in Matlab, the students need to obtain the transfer function and/or differential equation. Since the resulting differential equations are coupled, they need to obtain the state equations which are a set of four/three first order differential equations and then write a program to solve those equations using either Euler's or Runge Kutta method. In Simulink, after obtaining the transfer function and or state space model, the students can directly use the blocks in Simulink to obtain the response. There were 23 students enrolled in one of the ME sections and 33 students in PDM section. The students were not allowed to collaborate or use Google/You Tube or any other online help. However, they were allowed to use Matlab and Python help menu. They were given 48 hours to complete the exam.

Question 1 Problem Statement

A rover vehicle designed for use on the planets and moons is shown below.



The block diagram of the steering control is also shown below :



where $Gp(s) = \frac{s+1.5}{(s+1)(s+2)(s+4)(s+10)}$

(a)When $G_c = K_P$, find the value of K_P for stable system.

(b) Find the roots for K_P=100, 300 and 600.

(c) Predict the overshoot and settling time for a step input.

(d) Determine the actual response time for the step input for the three values of K_P and compare the actual results with the predicted ones.

Question 1 Assessment:

Overall 57% of ME students and 61% of PDM students got 100% in this question. The distribution of grades in each part is listed below:

22 out of 23 ME students and 28 out of 33 PDM students got 100% in part a.
21 out of 23 ME students and 26 out of 33 PDM students got 100% in part b.
19 out of 23 ME students and 22 out of 33 PDM students got 100% in part c.
15 out of 23 ME students and 29 out of 33 PDM students got 100% in part d.

The percentage of correct answers given by ME and PDM students in problem 1 for parts a-d are shown in Figure 8. It should be noted that since both Matlab and Simulink were taught in ME section therefore some students used Matlab while others used Simulink to complete their exam. The remaining students who did not score 100% either did some calculation mistake or just did part (a) wrong and the mistake is carried over in the remaining parts. The average score in problem 1 for ME students is 93% and for PDM students is also 93%.

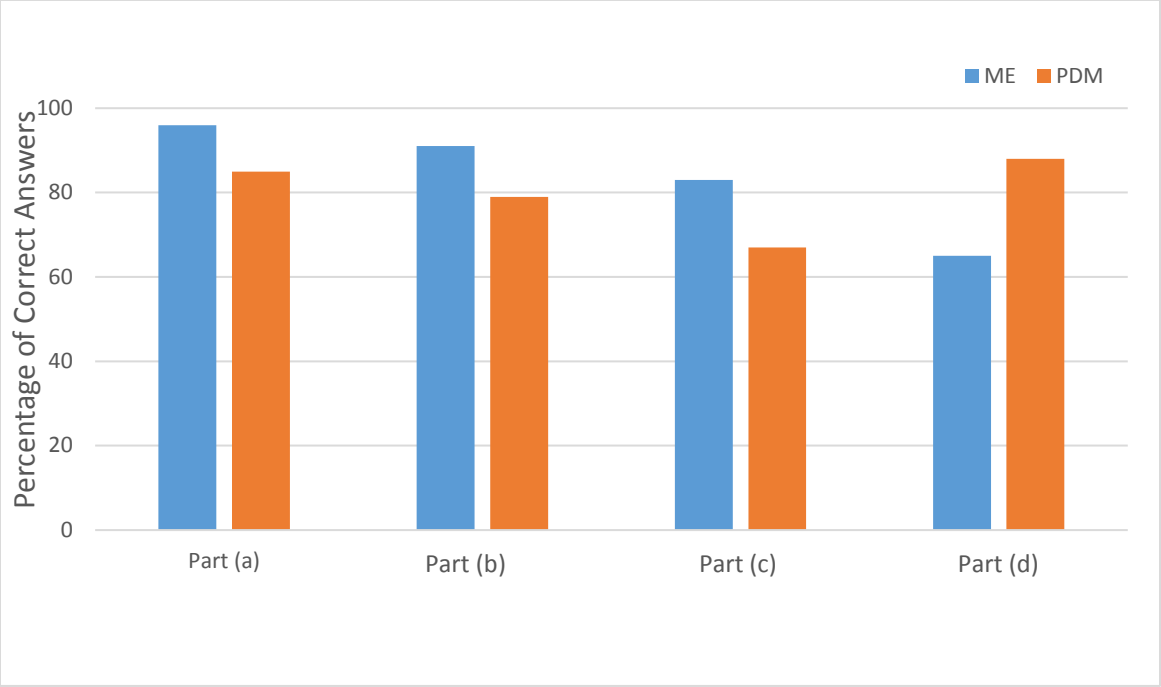


Figure 8- Bar graph of take home exam problem 1

Question 2 Problem Statement

Arc welding is one of the most important areas of application for industrial robots. In most manufacturing welding situations, uncertainties in dimensions of the part, geometry of the joint, and the welding process itself require the use of sensors for maintaining weld quality. Several systems use a vision system to measure the geometry of the puddle of the melted metal, as shown in Figure 9 below. This system uses a constant rate of feeding the wire to be melted.

(a) Calculate the maximum value of K for the system to be stable, K_{max}.

Use $K = K_{max}/2,$ i.e. half of the maximum value of found in part (a) for the rest of this problem.

With $K = K_{max}/2$:

(b) Determine the roots of the characteristic equation.

(c) Estimate the overshoot of the system when it is subjected to a step input.

(d) Plot the unit step response of the system.

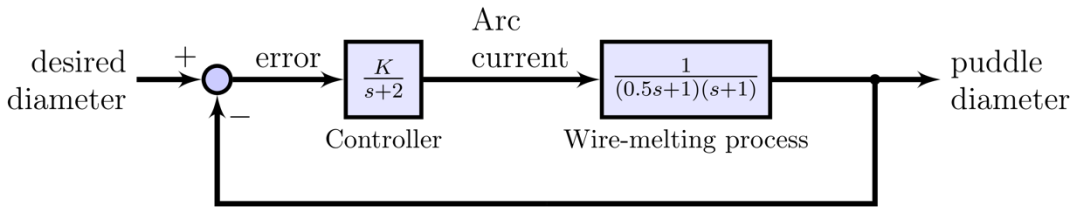


Figure 9- Block diagram of take home exam for problem 2

Question 2 Assessment

Overall 17% of ME students and 33% of PDM students got 100% in this question. The distribution of grades in each part is listed below:

10 out of 23 ME students and 21 out of 33 PDM students got 100% in part a.
23 out of 23 ME students and 28 out of 33 PDM students got 100% in part b.
16 out of 23 ME students and 22 out of 33 PDM students got 100% in part c.
21 out of 23 ME students and 30 out of 33 PDM students got 100% in part d.

The percentage of correct answers given by ME and PDM students in problem 2 for parts a-d are shown in Figure 10. Here also, for the remaining students who did not score 100%, the common mistake was finding an incorrect value of K in part (a) and then that mistake is carried over in the remaining parts. The average score in problem 2 for ME students is 91% and for PDM students is 88%.

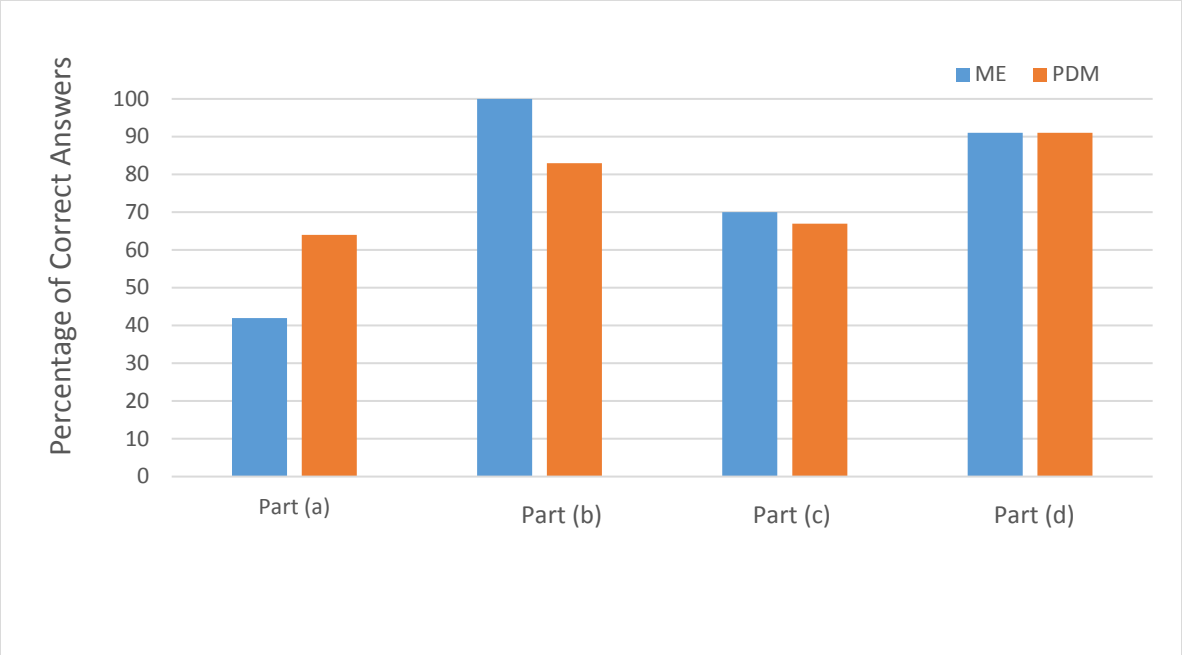


Figure 10- Bar graph of take home exam problem 2

Conclusion

Looking at the data, there seems to be no significant difference between the programming skills of ME and PDM students. The programming skills of ME students are equally good as PDM students. Also the type of software used does not have any effect on students learning. The slight variation in grades depend on the way the material is covered. The authors attempted to determine if there are any differences in student programming skills between using Matlab and Python in simulating complex dynamic systems. But based on the results, there seems no significant difference. This could be due to problems selected which were not difficult enough to show any difference between the two groups of students. Also the two versions were taught by two different instructors, therefore the amount of time spent on different topics was different. It should be noted that the ME majors require dynamics as a prerequisite whereas PDM majors does not have dynamics as a prerequisite. Therefore it was expected that ME students understand the dynamic systems in more depth than PDM students. However, based on the assessment presented in this paper, the PDM students are no less than ME students in understanding and simulating complex dynamic systems.

References

1. Ali, A., “Methods to Improve Students Learning in Dynamic Systems and Control Course,” *2017 ASEE Zone II Conference*, San Juan, Puerto Rico.
2. Bernstein, D., “Control experiments and what I learned from them: a personal journey,” Control Systems Magazine, *IEEE*, Vol. 18, No. 2, April 1998, pp. 81–88.
3. Clark, R. E., “Reconsidering Research on Learning from Media,” Review of Educational Research, Winter, 1983, Vol. 53, No. 4, pp. 445-459.

4. Felder, R. M. and Silverman, L. K., “Learning and Teaching Styles in Engineering Education,” Engineering Education 78(7), pp. 674-681, April 1988.
5. Jensen, E. (1998). “Teaching with the Brain in Mind,” Alexandria, VA: Association for Supervision and Curriculum Development.
6. Krauss, R., Ali, A., & Lenz, A., “Teaching Dynamic Systems and Control without Dynamics,” *2017 ASEE Annual Conference and Exposition*, Columbus, Ohio.
7. Lee, K.-M., Daley, W., and McKlin, T., “An interactive learning tool for dynamic systems and control,” *International Mechanical Engineering Congress & Exposition*, Anaheim, CA,1998.
8. Mann, H, and M. Sevcenko, “Course On Dynamics Of Multidisciplinary Controlled Systems,” *6th IFAC Symposium on Advances in Control Education*, Oulu, Finland, June 16-18, 2003.
9. Okamura, A. M., Richard, C., Cutkosky, M., et al., “Feeling is believing: Using a force-feedback joystick to teach dynamic systems,” *Journal of Engineering Education*, Vol. 91, No. 3, 2002, pp. 345–349.
10. Shiakolas, P. and Piyabongkarn, D., “Development of a Real-Time Digital Control System with a Hardware-in-the-Loop Magnetic Levitation Device for Reinforcement of Controls Education,” *IEEE Transactions on Education*, Vol. 46, No. 1, 2003, pp. 79–87.
11. https://www.python.org/doc/essays/blurb/