Impact of a Modeling Intervention in an Introductory Programming Course

Dr. Kelsey Joy Rodgers, Embry-Riddle Aeronautical University, Daytona Beach

Kelsey Rodgers is an Assistant Professor in the Engineering Fundamentals Department at Embry-Riddle Aeronautical University. She teaches a MATLAB programming course to mostly first-year engineering students. She primarily investigates how students develop mathematical models and computational models. She also conducts research around effective feedback and nanotechnology education. She graduated from the School of Engineering Education at Purdue University with a doctorate in engineering education. She previous conducted research in Purdue University's First-Year Engineering Program with the Network for Nanotechnology (NCN) Educational Research team, the Model-Eliciting Activities (MEAs) Educational Research team, and a few fellow STEM education graduates for an obtained Discovery, Engagement, and Learning (DEAL) grant. Prior to attending Purdue University, she graduated from Arizona State University with her B.S.E. in Engineering from the College of Technology and Innovation, where she worked on a team conducting research on how students learn LabVIEW through Disassemble, Analyze, Assemble (DAA) activities.

Dr. Jaqi C. McNeil, University of Louisville

J.C. McNeil is an Assistant Professor for the Department of Engineering Fundamentals at University of Louisville. Research interests include diversity in engineering, persistence, retention, and transitions to and from co-op experiences. Contact email: j.mcneil@louisville.edu

Dr. Matthew A. Verleger, Embry-Riddle Aeronautical University, Daytona Beach

Matthew Verleger is an Associate Professor of Engineering Fundamentals at Embry-Riddle Aeronautical University in Daytona Beach, Florida. His research interests are focused on using action research methodologies to develop immediate, measurable improvements in classroom instruction and the use of Model-Eliciting Activities (MEAs) in teaching students about engineering problem solving. Dr. Verleger is an active member of ASEE. He also serves as the developer and site manager for the Model-Eliciting Activities Learning System (MEALearning.com), a site designed for implementing, managing, and researching MEAs in large classes.

Dr. Farshid Marbouti, San Jose State University

Farshid Marbouti is an Assistant Professor of General (interdisciplinary) Engineering at San Jose State University (SJSU). He is currently the chair of SJSU Senate Student Success Committee. Farshid completed his Ph.D. in Engineering Education at Purdue University. His research interests center on First-Year Engineering student success and engineering design.

Impact of a Modeling Intervention in an Introductory Programming Course

Abstract

This complete research paper describes the impact of a modeling intervention on first-year engineering students' modeling skills in an introductory computer programming course. Five sections of the first-year engineering introductory programming course at a private, STEM+Business institution were revised to center around modeling concepts. These five sections made up the experimental group for this study. The comparison group consisted of four sections of the course that were not revised. Students in all these sections were given two different versions of a modeling problem two times in the semester to test their progress in gaining modeling skills. Each version required two submissions – a written solution and a coded solution. The assessment of these four submissions based on the three established dimensions of modeling were quantitatively analyzed in this study. The three dimensions within mathematical modeling that were the focus of this study were mathematical model complexity, modifiability, and reusability. Mathematical model complexity is being able to address the complexity of the problem. Modifiability addresses the generalizability of the model solution. Reusability is showing an understanding of the problem and the user. Statistical analysis showed that students in the experimental group had more gains in their demonstrated modeling abilities across all three dimensions than the students in the comparison group. This study demonstrated that intentional and explicit instructional strategies targeting model development resulted in greater gains in students' demonstrated modeling skills and both their written and coded solutions to a complex modeling problem.

Introduction

Engineers must understand how to build, apply, and adapt various types of models, including mathematical and computational models, to be successful. Throughout undergraduate engineering education, modeling is fundamental for many core concepts, though it is rarely explicitly taught [1]. There are many benefits to explicitly teaching modeling, particularly in the first years of an engineering program [1-3]. There are some well-developed pedagogies that demonstrate the successes of doing this. Model-eliciting activities (MEAs) are an impactful example of a pedagogical approach used in first-year engineering to teach mathematical modeling skills [3]. There is still a significant need for more meaningful ways of explicitly teaching modeling throughout the engineering curricula and especially in first-year engineering [1, 4].

There has been an extensive amount of research around modeling interventions within the Computational Adaptive Expertise (CADEX) [2, 5] and Models and Modeling Perspective (M&MP) [6] frameworks that have proven successful. Carberry, McKenna, Linsenmeier, and Cole [7] conducted research within the CADEX framework and found that explicit modeling interventions caused a significant shift in senior engineering students' concepts of models and modeling. Carberry and McKenna [1] expanded their research within the CADEX framework to gain a greater understanding of modeling conceptions and found more students discussed mathematical and predictive models when they were taught an explicit mathematical modeling module. Research efforts within the M&MP have focused around a mathematical modeling intervention called MEAs [3]. Some of this research has focused on how students develop mathematical model solutions to MEAs (e.g., [8, 9]), MEA implementation strategies within engineering courses (e.g., [10, 11]), and the improvement of MEA implementation strategies in large first-year engineering courses (e.g., [12]) and upper division courses (e.g.,[13, 14]).

According to Lesh and Harel [15] "models are conceptual systems that generally tend to be expressed using a variety of interacting representational media, which may involve written symbols, spoken language, computer-based graphics, paper-based diagrams or graphs, or experience-based metaphors" (p. 159). Physical models (e.g., prototypes, drawings) are the most common type of models that engineering students identify as a type of model and discuss as a model used in the design process [1, 16]. Although physical models are one type of model that engineers use, there are other critical types of modeling, including mathematical models [3, 6, 17] and computational models [18, 19]. Within engineering curricula, various types of models need to be explicitly introduced to enhance students' definitional knowledge and implemented to develop students' modeling skills. Interventions need to focus on types of models, model application, and model development in engineering curricula, especially on mathematical and computational models [1, 3, 4, 19]. Rodgers [20] found that although students participated in well-developed mathematical modeling activities, they still were unable to demonstrate an ability to define mathematical models. Rodgers [20] and Carberry and McKenna [1] discussed the importance of incorporating explicit language that identifies various types of models throughout engineering courses that incorporate modeling.

For this paper, two types of models were explored; mathematical models and computational models:

- *Mathematical Models:* Mathematical models utilize mathematics to explain and enable exploration of a real-world system. A pedagogical approach that has been extensively studied to enable students to develop mathematical modeling skills is the use of MEAs [17]. MEAs are a type of open-ended problem where students create a mathematical model to meet particular criteria and constraints for a stakeholder based on provided, relevant data [3].

- *Computational Models:* Computational models, in the context of this work, are often seen in the form of simulations (e.g., [21-23]). Where computational modeling is often taught is in the context of programming courses (e.g., [18]), but rarely is it referred to as computational modeling or simulation. For example, having students develop code that can respond to a variety of input conditions is a form of simulation, but is typically referred to as "programming logic".

Problem solving, design, and introductory computer programming are some fundamental course content integrated into most first-year engineering courses [4, 24, 25]. All three of these involve modeling, although it may not be clearly discussed. Mathematical modeling is essential to engineering problem solving and design in the workforce [4]. Teaching students how to develop an algorithmic solution (a type of model) is fundamental to programming, although sometimes there is greater focus on syntax [25]. Most engineering education studies on computer programming focus on paired programming (e.g., [26]), extreme programming (XP) (e.g., [27]), or active learning teaching pedagogies [28]. This study focused specifically on modeling development in a first-year programming course.

Research Purpose and Questions

The focus of this study is on the opportunities that concentration on modeling can present in a computer programming course, since there are many challenges in teaching students to program. Introductory programming courses face many challenges in engineering, such as low retention rates, students entering with extremely different backgrounds in programming and computers, and overemphasis on syntax over learning programming skills [24, 25].

In this study, we investigated the following research questions: (1) In a computer programming course, how does the assessment of students' written and coded solutions to a complex open-ended problem change over the course of multiple submissions? and (2) How do students' solutions compare across sections that have an emphasis on modeling and others that do not?

Methods

Setting and Participants

In Fall 2018, 302 students enrolled in an introductory computer programming course for engineers across 16 sections at a medium-sized, private, STEM+Business only university. The programming language for the course is MATLAB. All mechanical, civil, and aerospace engineering students are required to take this course. The electrical and computer engineering students take a similar course that uses Java as the programming language. This course is open to other students at the university and required by some non-engineering degree programs, such as the astronomy and astrophysics program and the unmanned aircraft systems science degree. Even with these additional requirements, sections are made up of mostly engineering students and more specifically aerospace engineering students. The focus of the course is to teach engineering students how to develop effective computer programs for solving problems. The learning objectives of the course are to:

1. Demonstrate understanding of the role of software design when solving problems using the computer.
2. Apply knowledge of mathematics and computer programming to communicate ideas when solving computational problems.
3. Design and implement algorithmic solutions to problems requiring user input/output (I/O), data processing, control structures, arrays, and file input/output (I/O).
4. Solve problems of intermediate complexity requiring the use of non-numerical data such as characters and strings.
5. Apply a top-down design methodology to problems of intermediate complexity using functions.

Five of the 16 sections were taught by two instructors that explicitly incorporated modeling throughout the course materials by developing various levels of modeling problems for all the homework assignments and in-class activities. The revised course was designed to scaffold students from solving more close-ended modeling problems to more open-ended modeling problems across the semester. The language of the class focused on mathematical and computational models, in addition to the more common computing language of algorithms and computer programs. Both instructors used the same homework assignments and exams, but had some variations in their in-class activities and presentations. There was a total of 82 students enrolled in these five sections. The other 11 sections taught by three other instructors were not significantly modified from previous iterations of the course. There was a total of 220 students enrolled in these 11 sections. These sections focused more on syntax and used close-ended problems with one correct answer for most of their assignments.

All the students from the 16 sections of the courses were given one common modeling problem with two different versions to complete individually. Each version consisted of two submissions – the first submissions were students' written solutions potentially with tables, figures, and flowcharts (i.e. their mathematical model) and the second submission were students' MATLAB code with comments, as specified (i.e. their computational model).

The modeling problem was developed using the six design principles of the M&MP theoretical framework [6]. The second version of the problem contained more data than the first version to prompt the students to reevaluate their solutions. The intention of the second version was also to ensure more iterations in their model development and capture how their demonstrated modeling skills developed across the course. The first submission for the problem was similar to a MEA [3] in that the students had to interpret a problem and data to develop a model. The second submission followed the concept of a model-adaptation activity (MAA) [3] in that the students had to apply the model they developed through the first submission.

The modeling problem challenged students to develop a model to predict the probability of a student team (user) to win in the upcoming Design, Build, Fly competition based on both the past performance data of teams that competed in previous competitions and the student team's (user) tested values for three different missions (i.e., successful flight and landing, fastest flight time, and largest product of laps flown and cargo carried within a set time frame). The program also had to provide recommendations to help improve the student team's (user) chances of winning, when appropriate based on the team's inputted tested values. In the first version of the problem, the students were provided a table of the top four teams' scores for each mission and overall for the past four years. In the second version of the problem, the students were provided an additional data set in Excel that consisted of the top 50 teams' scores for the most recent competition. This additional data set complemented the previous one they received. An example of the problem provided to the students for the first version is provided in Appendix A. Some minor modifications were made to the problem provided for distribution purposes.

Data Collection

The 82 students in the five sections with the revised modeling curriculum make up the experimental group for this study. Out of the 11 sections taught by the three instructors (i.e. the comparison group), only one instructor both implemented the assignment consistent with the way it was developed and ensured the assessment of the assignment was consistent. This instructor's four sections consisting of 88 students is the comparison group that was used for this study. The data collected for this study were the students submitted solutions for all four of the modeling problem submissions (summarized in Table 1). Within the experimental group, 79 students submitted at least one submission for the assigned modeling problem. Within the comparison group, 85 students completed at least one submission for the assigned modeling problem. All 164 of these students' submissions (ranging from 1 to 4 submissions per student) were used to investigate the established research questions.

Table 1: Description of the four submissions for the modeling problem

Submission	Modeling Problem Assigned	Submission Type
Submission 1	Version 1 (table of data)	Written solution
Submission 2	Version 1 (table of data)	Coded solution (.m file)
Submission 3	Version 2 (Excel file with a larger data set)	Written solution
Submission 4	Version 2 (Excel file with a larger data set)	Coded solution (.m file)

Data Analysis

To assess the quality of students' models in each of the submissions, a rubric was established. The rubric consisted of items developed based on categories established for assessing students' solutions to MEAs [8, 29]. The three recognized dimensions focused on were mathematical model complexity, modifiability, and reusability. The purpose of the mathematical model complexity dimension is to assess students' ability to address the complexity of the problem in an elegant solution [8, 29]. Items related to this category consisted of proper implementation of relevant equations, appropriately incorporated data, and meaningful potential outputs from the model. There was a total of nine items developed within this category. The purpose of the modifiability dimension is to ensure a student's model is generalizable by evaluating the student's rationale for each step in their model [8, 29]. There was a total of five items created that focused on students' rationale for five major components of their models. The purpose of the reusability dimension is to assess students' understandings of the given problem and their client or user (i.e., problem scoping) [8, 29]. There was a total of three items developed that focused on students' assumptions and understandings of needed information from their client/user. Shareability, the fourth dimension from the literature [8, 29], was not focused on in the developing rubric items because strict formatting was not required for the students' submissions. These 17 modeling rubric items were assessed in all four submissions.

Additional rubric items were developed to assess the quality of students' programs for each of the modeling problems. The coding rubric items were only used to assess the second submission of each problem (i.e. submissions 2 and 4). This portion of the rubric focused more on syntax errors and was only used for grading purposes in the class. It was not used for this study.

The entire rubric used for the fourth submission is provided in Appendix B. All the scores for the rubric items shown in the appendix are based on the fourth submission scoring used for assigning students' grades. For the data analysis all the rubric items for were assigned a possible score of 1 point. Most of the rubric items consisted of full points for demonstrated or no points for not demonstrated. For the three rubric items relevant to this study with three possible scoring levels, the scores were converted to 1 (demonstrated), 0.5 (attempted), and 0 (not demonstrated).

The students' solutions to the modeling problem that were included in this study were graded by three student graders. These graders completed a required training that consisted of assessing multiple sample student solutions using the rubrics to ensure consistency across grading – there was a separate training for each version of the modeling problem where the application of the rubric for both the model and code submissions was explained in detail.

The training was completed in an iterative process of the graders evaluating a student's solution individually and then discuss the grades until all the graders assessed a student's work individually with at least 90% agreement across the three categories and 80% agreement for each individual category. To capture the graders' scores and improve agreement across the graders, each rubric item was discussed one at a time. First the graders' evaluations were documented and then their selections were deliberated – all uncertainties and/or disagreements about selections were discussed until a firm agreement was made. Prior to the training, two researchers had assessed five potential sample solutions and came to an agreement on the grade for each rubric item to help guide the graders in the training. There were some cases in the first few rounds of grading during training for the first version where two graders selected a grade that differed from the one the researchers agreed upon. In all cases, the graders' responses were revised to match the researchers' conclusion. The graders were quickly able to understand the application of the rubric. Most of the items only had a yes or no (all or nothing) possible score; this seemed to make the training process much easier. The first training consisted of four rounds of grading sample solutions and resulted in 100% agreement by the final round (refer to Table 2). The second training only consisted of grading two sample solutions since the graders demonstrated a proficient understanding of the process (refer to Table 3). After each training, the three student graders each graded students' solutions for three sections. A grader typically graded all four submissions for each student, but there were some instances where students received grades from two different graders across their submissions.

Table 2: Percent agreement for the first version of the modeling problem

Round	Mathematical Model Complexity	Modifiability	Reusability	Overall
1	0.667	0.867	0.556	0.706
2	0.778	0.867	0.556	0.765
3	0.778	0.867	0.778	0.804
4	1.000	1.000	1.000	1.000

Table 3: Percent agreement for the second version of the modeling problem

Round	Mathematical Model Complexity	Modifiability	Reusability	Overall
1	0.778	0.867	0.556	0.765
2	1.000	0.867	1.000	0.961

The resulting assessments of students' works based on the 17 modeling items of the rubric was analyzed using descriptive statistics and t-tests [30]. The t-tests were used to determine statistically significant differences between the experimental and comparisons groups and between the first and last submission within each group. The t-tests were also used to determine statistically significant differences between the two instructors in the experimental group.

Findings

Comparing Instructors in Experimental Group

All the students in the experimental group were first compared across the two instructors. Based on all the students' submissions, there were no statistically significant differences between the students' modifiability and reusability scores for the two different instructors, as show in Table 4. There were statistically significant differences between their students' mathematical model complexity scores. Since there were statistically significant differences between their mathematical model complexity scores, the instructors' students were not grouped together for the analysis of this dimension. The instructors' students were grouped together for the analysis of the other two dimensions – modifiability and reusability.

Table 4: Experimental group – Difference between instructors

Dimension	Instructor	N	Mean	Standard Deviation	p-value
Mathematical Model Complexity	1	188	7.582	1.6835	0.000*
	2	95	6.342	1.8655	
Modifiability	1	188	2.090	1.6145	0.808
	2	95	2.042	1.4941	
Reusability	1	188	2.277	0.6359	0.314
	2	95	2.195	0.6616	
* statistically significant difference between two different instructors					

Mathematical Model Complexity – Comparing Instructors

The average scores for all the mathematical model complexity rubric items across all four submissions for all three instructors (the two making the experimental group and one making the comparison group) are presented in Figure 1. Along the mathematical model complexity dimension, the students in both instructors' courses for the experimental group and the students in the comparison groups improved their written solutions from the first to the third submissions, although it is not as visually apparent for the comparison group as it is for both the experimental group instructors. All the students in both the experimental groups and the comparison groups also improved their coded solutions from the second to the last submissions, along the mathematical model complexity dimension. Although the students from the comparison group received the highest average score on the first submission, the students who took Instructor 1 from the experimental group outperformed the comparison group students on all other submissions. Instructor 1's students average scores increased by 1.251 points compared to the average scores of the students from the comparison group that only increased by 0.384 points. The students who took Instructor 2 from the experimental group received the lowest average score on all four submissions, but had the greatest gains across the submissions (with the average increasing by 1.780 points).

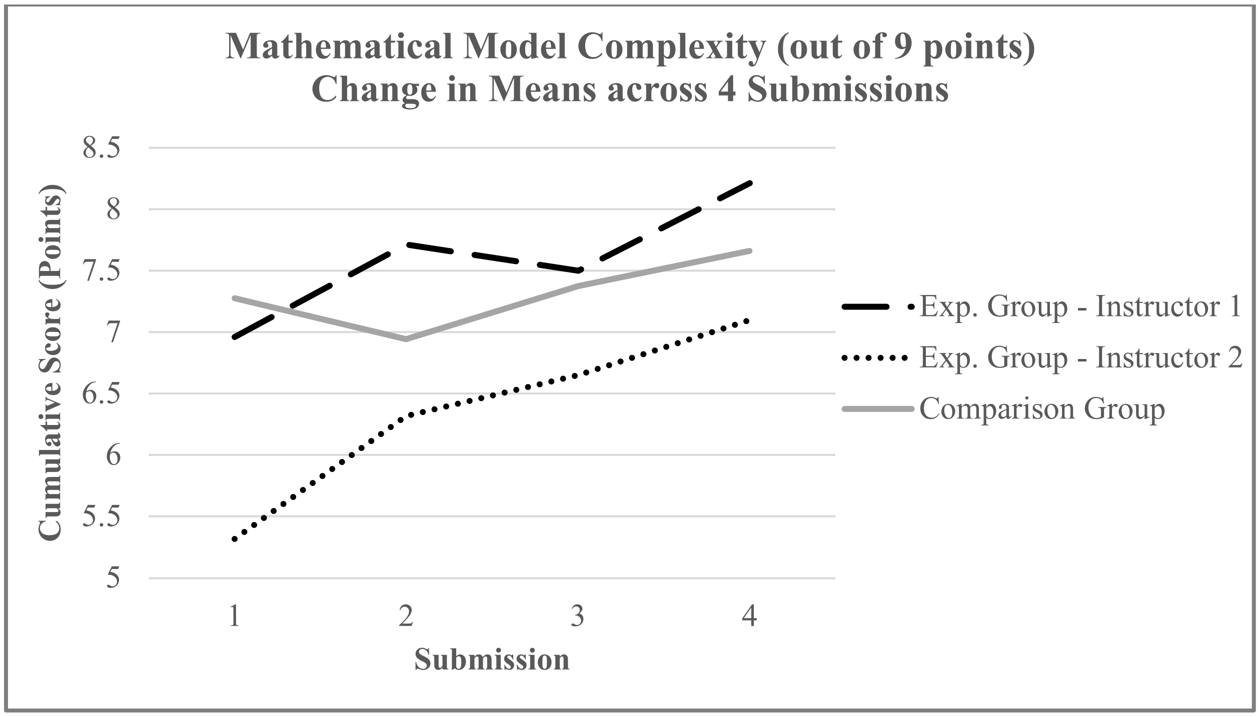

Figure 1: Average of students' scores on Mathematical Model Complexity items

As demonstrated in Table 5, the students in the comparison group started out with the highest scores, but they did not maintain the highest scores nor have as much gains as the students from the experimental group. The students' scores for the mathematical model complexity dimension in the experimental group had a statistically significant difference from the first to the last submission, as shown in Table 5. The students' scores for the mathematical model complexity dimension in the comparison group did not have a statistically significant difference from the first to the last submission, as shown in Table 5.

Table 5: Mathematical model complexity – All instructors – Difference between submissions

Exp. Group Instructor/ Comparison Group	Submission	N	Mean	Standard Deviation	p-value
Exp. Group Instructor 1	1	52	6.962	1.6828	0.000*
	4	47	8.213	1.3259	
Exp. Group Instructor 2	1	25	5.320	1.1715	0.000*
	4	25	7.100	1.9632	
Comparison Group	1	81	7.278	1.5632	0.191
	4	68	7.662	2.0064	
* statistically significant difference between first and last submissions					

Based on the students' mathematical model complexity scores in the first submission, there was a statistically significant difference between the two instructors in the experimental group and between Instructor 2 in the experimental group and the comparison group, as shown in Table 6. There was not a statistically significant difference between Instructor 1 in the experimental group and the comparison group for either the first or the last submission. There was still a statistically significant difference between the two instructors in the experimental group in the last submission. There was no longer a statistically significant difference between Instructor 2 in the experimental group and the comparison group in the last submission, as shown in Table 6.

Table 6: Mathematical model complexity – Differences between instructors

Exp. Group Instructor/ Comparison Group	Submission	p-value
Exp. Group Instructor 1 vs. Exp. Group Instructor 2	1	0.000*
	4	0.015*
Exp. Group Instructor 1 vs. Comparison Group	1	0.271
	4	0.079
Exp. Group Instructor 2 vs. Comparison Group	1	0.000*
	4	0.232
* statistically significant difference between instructors' students		

Modifiability – Experimental Group vs. Comparison Group

The average scores for all the modifiability rubric items across all four submissions for both the experimental and comparison groups are presented in Figure 2. The students from the control group received higher scores than the students from the experimental group on all their submissions. Although the comparison group outscored the experimental group on every submission, the experimental group had significant gains between most of their submissions. The students in the experimental group improved their written solutions from the first to the third submissions and their coded solutions from the second to the fourth submissions, along the modifiability dimension. The students in the comparison group on the other hand had declining scores across their written and coding solutions along the modifiability dimension.

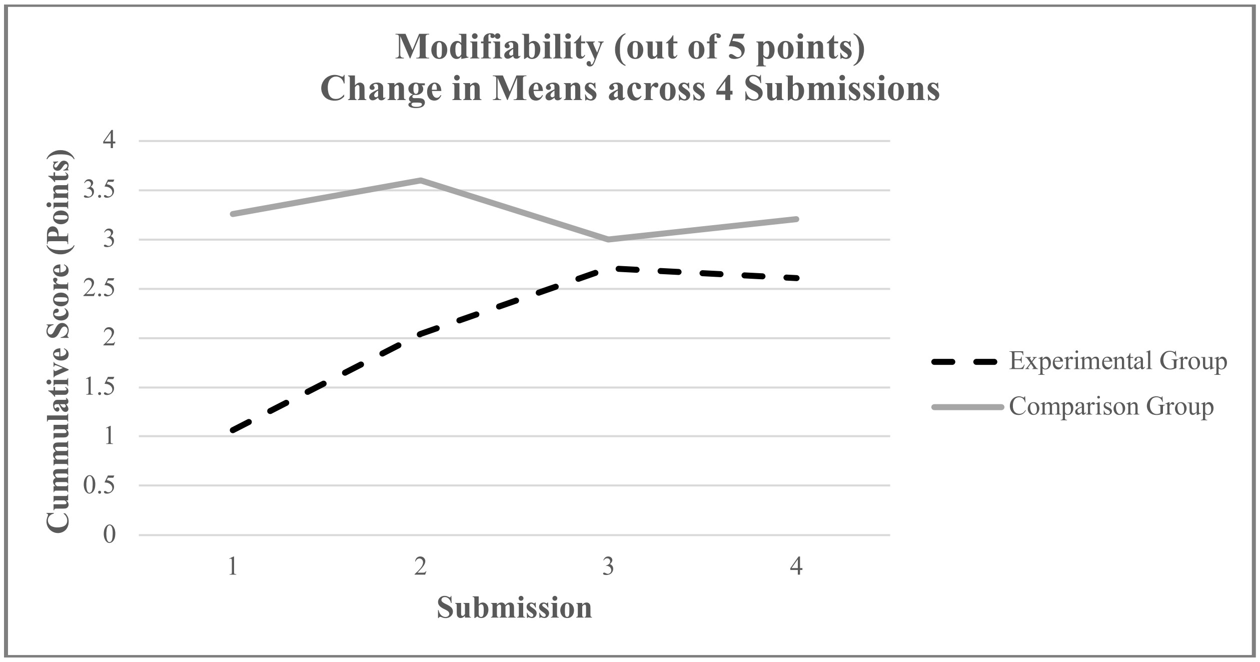

Figure 2: Average of students' scores on Modifiability items

As demonstrated in Figure 2, the students in the comparison group maintained higher scores on the modifiability dimension than the students in the experimental group across all four submissions. Although their scores were higher, the students in the experimental group had greater gains across the course of the semester. There was a statistically significant difference between students' scores on the first and last submissions for the experimental group, as shown in Table 7. There was not a statistically significant difference between the students' scores on the first and last submission for the comparison group.

Table 7: Modifiability – Differences between submissions

Exp./Comp.	Submission	N	Mean	Standard Deviation	p-value
Experimental Group	1	77	1.065	1.0312	0.000*
	4	72	2.611	1.5157	
Comparison Group	1	81	3.259	1.5065	0.815
	4	68	3.206	1.2762	
* statistically significant difference between first and last submissions					

The students' scores for the modifiability dimension in the experimental and comparison groups had a statistically significant difference for the first submission and maintained a statistically significant difference in the last submission, as shown in Table 8.

Table 8: Modifiability – Differences between experimental and comparison groups

Submission	p-value
1	0.000*
4	0.013*
* statistically significant difference between experimental and comparison groups	

Reusability – Experimental Group vs. Comparison Group

The average scores for the reusability rubric items across the submissions for both experimental and comparison groups are presented in Figure 3. The students from the comparison group received higher scores than the students from the experimental group on three submissions. Although the comparison group outscored the experimental group on most submissions, the experimental group had greater gains across their submissions. Like the modifiability dimension, the students' scores improved across their written and coded solutions for the experimental group, while students' scores declined across both solutions for the comparison group.

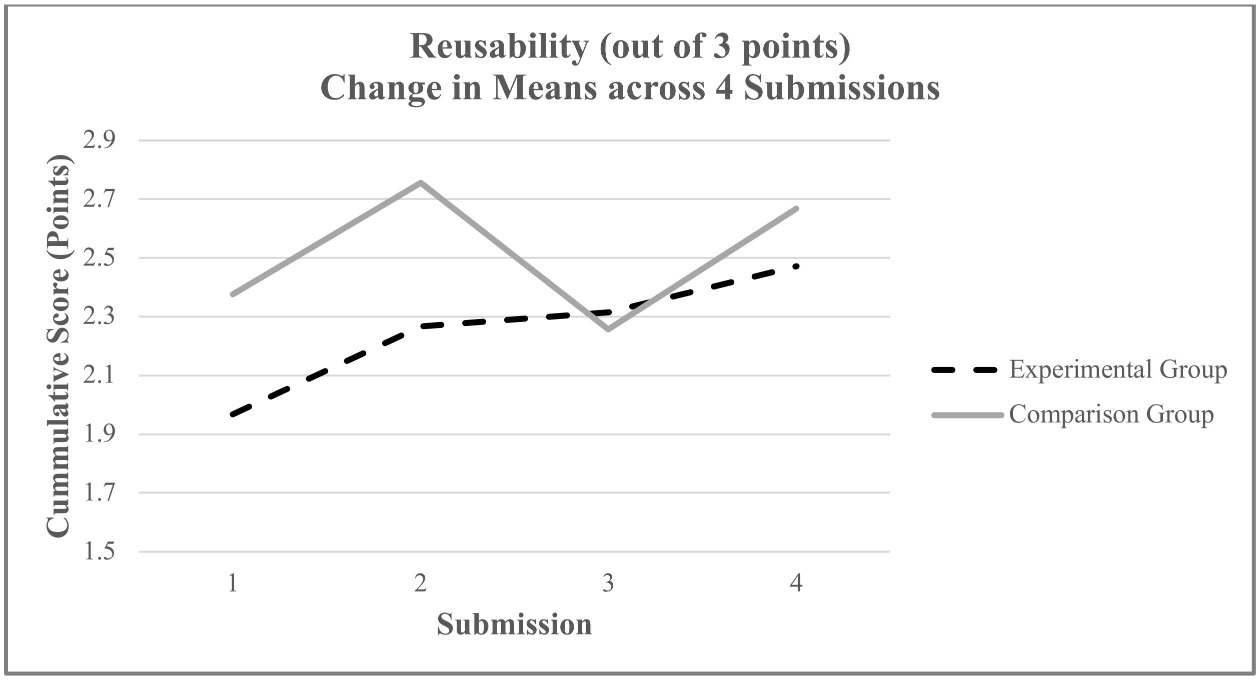

Figure 3: Average of students' scores on Reusability items

As demonstrated in Figure 3, the students in the comparison group maintained higher scores on the reusability dimension than the students in the experimental group across most of the submissions (all except the third submission). Although their scores were higher for most of submissions, the students in the experimental group had greater gains across the course of the semester. There was a statistically significant difference between students' scores on the first and last submissions for both the experimental and comparison groups, as shown in Table 9.

Table 9: Reusability – Differences between submissions

Exp./Comp.	Submission	N	Mean	Standard Deviation	p-value
Experimental Group	1	77	1.968	0.6404	0.000*
	4	72	2.472	0.5746	
Comparison Group	1	81	2.377	0.6733	0.005*
	4	68	2.669	0.5834	
* statistically significant difference between first and last submissions					

The students' scores for the modifiability dimension in the experimental and comparison groups had a statistically significant difference for the first submission and maintained a statistically significant difference in the last submission, as shown in Table 10.

Table 10: Reusability – Differences between experimental and comparison groups

Submission	p-value
1	0.000*
4	0.046*
* statistically significant difference between experimental and comparison groups	

Summary

As demonstrated in the figures above, the students from the experimental group were consistently receiving higher scores on each of the rubric categories across their submissions. For the experimental group, there was a statistically significant difference between students' scores on the first and last submissions across all three rubric dimensions.

As demonstrated in the figures above, the students' scores in the comparison group were not consistently improving nor declining; their scores typically fluctuated across the four submissions. For the comparison group, there was a statistically significant difference between students' scores on the first and last submissions across the reusability dimension.

Conclusions

The results identified differences that exist in the mathematical and computational model development between students who were exposed to the revised modeling language and examples (experimental group) compared to those students who were not (comparison group). There were three dimensions in the modeling rubric that were analyzed for this study, mathematical model complexity, modifiability, and reusability [8, 29]. Each dimension of the rubric was analyzed separately to capture differences in experimental and comparison groups development. Students in the experimental group, who were exposed to the revised modeling language and assignments, had significant gains in mathematical model complexity, modifiability, and reusability. Even though the students in the two different instructors' courses within the experimental group started with significantly different scores on the mathematical model complexity dimensions, both groups had statistically significant gains. This study showed that the experimental groups did have increases in demonstrated modeling abilities through the intentional and explicit instructional strategies targeting model development.

The students who were exposed to the revised modeling language and materials had greater gains across the course of the semester in their ability to address the complexity of the problems (mathematical model complexity), understand the application of the model and potential for adaptation (modifiability), and address the users' anticipated needs (reusability). The students who were not exposed to the modeling language nor given the opportunity to engage in modeling activities beyond the assigned modeling problem did not demonstrate as much improvement. Their completion of the two versions of the one modeling problem did not seem to provide enough impact on their modeling abilities; the additional course materials seemed to be fundamental in leading to this success. The comparison group did demonstrate statistically significant improvement in their ability to understand the user (reusability). This improvement in both groups could be attributed to the fact that emphasizing writing code for a given user and understanding your assumptions in solving a problem to know when the solution is applicable is embedded in the course. These findings help inform other researchers and instructors how to help students develop modeling skills, specifically modeling skills related to mathematical model complexity, modifiability, and reusability dimensions. This research will build upon this idea by further analyzing impact of the revised modeling language in more courses and covering more types of modeling, including physical and business models.

Acknowledgements

This work was made possible by a collaborative research grant from the National Science Foundation (DUE 1827392; DUE 1827600; DUE 1827406). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

References

[1] A. R. Carberry and A. F. McKenna, "Exploring student conceptions of modeling and modeling uses in engineering design," *Journal of Engineering Education,* vol. 103, no. 1, pp. 77-91, 2014.

[2] A. McKenna, R. Linsenmeier, and M. Glucksberg, "Characterizing computational adaptive expertise," in *2008 ASEE Annual Conference and Exposition*, 2008.

[3] J. S. Zawojewski, H. A. Diefes-Dux, and K. J. Bowman, *Models and modeling in engineering education: Designing experiences for all students*. Sense Publishers, 2008.

[4] J. Gainsburg, "Learning to model in engineering," *Mathematical Thinking and Learning,* vol. 15, no. 4, pp. 259-290, 2013.

[5] A. F. McKenna, "Adaptive expertise and knowledge fluency in design and innovation," in *Cambridge handbook of engineering education research*: Cambridge University Press, 2015.

[6] R. Lesh, H. M. Doerr, G. Carmona, and M. Hjalmarson, "Beyond constructivism," *Mathematical thinking and learning,* vol. 5, no. 2-3, pp. 211-233, 2003.

[7] A. R. Carberry, A. F. McKenna, R. A. Linsenmeier, and J. Cole, "Exploring senior engineering students' conceptions of modeling," in *118th ASEE Annual Conference and Exposition*, 2011.

[8] H. A. Diefes-Dux, M. A. Hjalmarson, and J. S. Zawojewski, "Student Team Solutions to an Open-Ended Mathematical Modeling Problem: Gaining Insights for Educational Improvement," *Journal of Engineering Education,* vol. 102, no. 1, pp. 179-216, 2013.

[9] H. A. Diefes-Dux, K. Bowman, J. S. Zawojewski, and M. Hjalmarson, "Quantifying aluminum crystal size part 1: The model-eliciting activity," *Journal of STEM Education: Innovations and Research,* vol. 7, no. 1/2, p. 51, 2006.

[10] H. A. Diefes-Dux, M. A. Hjalmarson, T. K. Miller, and R. Lesh, "Chapter 2: Model-eliciting activities for engineering education," *Models and modeling in engineering education: Designing experiences for all students,* pp. 17-35, 2008.

[11] E. Hamilton, R. Lesh, F. Lester, and M. Brilleslyper, "Model-Eliciting Activities (MEAs) as a Bridge between Engineering Education Research and Mathematics Education Research," *Advances in Engineering Education,* vol. 1, no. 2, p. n2, 2008.

[12] H. A. Diefes-Dux and P. Imbrie, "Chapter 4: Modeling activities in a first-year engineering course," *Models and modeling in engineering education: Designing experiences for all students,* pp. 37-92, 2008.

[13] R. M. Clark, L. J. Shuman, and M. Besterfield-Sacre, "In-Depth Use of Modeling in Engineering Coursework to Enhance Problem Solving," in *Modeling Students' Mathematical Modeling Competencies*: Springer, 2010, pp. 173-188.

[14] T. P. Yildirim, L. Shuman, M. Besterfield-Sacre, and T. Yildirim, "Model eliciting activities: assessing engineering student problem solving and skill integration processes," *International Journal of Engineering Education,* vol. 26, no. 4, pp. 831-845, 2010.

[15] R. Lesh and G. Harel, "Problem solving, modeling, and local conceptual development," *Mathematical thinking and learning,* vol. 5, no. 2-3, pp. 157-189, 2003.

[16] A. R. Carberry and A. F. McKenna, "Engineering student's conceptions of model uses in design," in *2011 Research in Engineering Education Symposium, REES 2011*, 2011.

[17] R. Lesh, M. Hoover, and A. Kelly, "Equity, assessment, and thinking mathematically: Principles for the design of model-eliciting activities," *Developments in school mathematics education around the world,* vol. 3, pp. 104-130, 1993.

[18] O. B. J. Daniel K. Howe, "Developing an Interactive Computer Program to Enhance Student Learning of Dynamical Systems," in *2016 ASEE Annual Conference & Exposition*, 2017.

[19] A. J. Magana, "Modeling and Simulation in Engineering Education: A Learning Progression," *Journal of Professional Issues in Engineering Education and Practice,* vol. 143, no. 4, p. 04017008, 2017.

[20] K. J. Rodgers, H. A. Diefes-Dux, M. Zielinski, and K. Madhavan, "Students' definitional knowledge of mathematical models," Journal manuscript in preparation.

[21] A. J. Magana, S. P. Brophy, and G. M. Bodner, "Instructors' intended learning outcomes for using computational simulations as learning tools," *Journal of Engineering Education,* vol. 101, no. 2, pp. 220-243, 2012.

[22] A. Stefan, "A Computer Model of Cell Dynamics Using Agents," in *American Society for Engineering Education*, 2010: American Society for Engineering Education.

[23] J. P. A. Omer Farook, Athula Kulatunga, Ashfaq Ahmed P.E., Wangling Yu, Yoonill Lee, Hassan Abdullah Alibrahim, "Freshman Experience Course in Electrical and Computer Engineering Technology Emphasizing Computation, Simulation, Mathematical Modeling, and Measurements," in *2017 ASEE Annual Conference & Exposition*, 2017.

[24] A. El-ZEin, T. Langrish, and N. Balaam, "Blended Teaching and Learning of Computer Programming Skills in Engineering Curricula," *Advances in Engineering Education,* vol. 1, no. 3, p. n3, 2009.

[25] H. Fangohr, "A comparison of C, MATLAB, and Python as teaching languages in engineering," in *International Conference on Computational Science*, 2004: Springer, pp. 1210-1217.

[26] C. McDowell, L. Werner, H. E. Bullock, and J. Fernald, "The impact of pair programming on student performance, perception and persistence," in *Software Engineering, 2003. Proceedings. 25th International Conference on*, 2003: IEEE, pp. 602-607.

[27] L. Williams and R. Upchurch, "Extreme programming for software engineering education?," in *Frontiers in Education Conference, 2001. 31st Annual*, 2001, vol. 1: IEEE, pp. T2D-12.

[28] J. McNeil, A. Thompson, and N. Hawkins, "A Comparison of Students Learning Programming with Online Modules, Instruction, and Team Activities " presented at the 2018 ASEE Annual Conference & Exposition Salt Lake City, Utah, 2018. [Online]. Available: https://peer.asee.org/29665.

[29] H. A. Diefes-Dux, J. S. Zawojewski, and M. A. Hjalmarson, "Using educational research in the design of evaluation tools for open-ended problems," *International Journal of Engineering Education,* vol. 26, no. 4, p. 807, 2010.

[30] J. M. Stonehouse and G. J. Forrester, "Robustness of the t and U tests under combined assumption violations," *Journal of Applied Statistics,* vol. 25, no. 1, pp. 63-74, 1998.

Appendix A. Modeling Problem Assignment (Version 1 – Submission 1 and 2)

Modeling Problem – Design, Build, Fly – Version 1

Problem:

At this institution, there are many different organizations and competitions that you can participate in. Design, Build, Fly is one competition that our institution regularly competes in. In this competition, students must design, document, fabricate, and demonstrate the aircraft they determine to be capable of achieving the highest score on the three specified missions. Mission scores are based on the demonstrated mission performance obtained during the contest and calculated based on given formulas. You can find out more information about the actual competition at: https://www.aiaadbf.org/.

In this problem you will develop a computational model that will output to the user (a team using their test flight data) their final predicted scores, their chances of winning the upcoming competition, and recommendations of how to increase their odds of winning based on how the team's aircraft is preforming (data from the user) and how the winning teams preformed at previous competitions (data provided in tables in theory section). *Please note there are some modifications to how the competition works for this problem. There are some assumptions that are not valid to the real competition rules.*

Submissions:

Please note that the submission for this assignment may be different from any of your homework assignments. A written model (or algorithm) will be due for this problem in about one week and then your code for your solution will be due about one week later *(refer to online course materials for exact dates)*.

- Your **model** (or algorithm) – you must complete the steps below on paper. Keep in mind assumptions will be very important for your user to understand the constraints and limitations for using your solution. For your solution step (Step 5), you must write out clear directions of how you will code your solution. This can be in the form of a flow chart, bullet point steps, or paragraph format. Be sure that any method you choose is clear. In addition to your steps, you must justify each step – meaning you must explain your reasoning for designing the step the way that you did *(refer to the example provided in the bullet point below)*. In addition to your written model, you must provide the outputs of your model for three test cases (set of sample inputs).
  - Example of some steps and *justifying them* (keep in mind some parts of this example are not good solutions to this problem): (EXAMPLE 1) First I am going to ask the user their empty aircraft weight and wing span to calculate their RAC using the provided formula. If their RAC is less than 10, then I am going to end the program and tell them they have a 0% chance of winning and they need to increase their aircraft's weight and/or wing span. *I think any plane that has too small of a weight and/or wing span will not be capable of flying and completing the missions, so I don't want to waste time asking the rest of the questions and doing any more calculations.* (EXAMPLE 2) I am going to use the user's time flown and the fastest time flown from the most recent competition to calculate the user's mission 2 score using the provided equation. *I am going to use the fastest time flown from the most recent competition because I believe all other times are outdated and not relevant.*

- Your **code** (or computational model) will be based on the model that you submit. It is natural to modify your model throughout the process of coding it. Be sure to document changes and comment in justifications for new steps implemented in your revised model.

Solution: *Below are the 7 steps of the engineering process discussed in the course. Remember these steps are used to guide you through solving a problem. Keep in mind sometimes these steps are iterative.*

1. **Decipher Problem Statement**

The first step is to decipher the problem and identify the information (or variables) given and what information you need to find (or display to the user at the end of your program).

Givens (Inputs):

Keep in mind that you may not have to ask the user all these questions every time. Think about what you need to ask when and why. Think about how you will use these inputs.

- The team's mission 1 results:
  - Successful landing?
  - Completed three laps in 5 minutes?
- The team's mission 2 results:
  - Successful landing?
  - How many seconds to fly 3 laps?

- The team's mission 3 results:
    - Successful landing?
    - How many passengers?
    - How many payload blocks?
    - How many laps flown in 10 minutes?
- The team's Rated Aircraft Cost:
    - Empty Aircraft Weight (oz.)
    - Aircraft Wing Span (in.)

Finds (Outputs):

- The team's predicted final score. (Number)
- The team's probability of winning based on their test flight data. (Percentage)
- Recommendations to improve the team's odds of winning. (Statements)

2. Draw a Diagram

Sometimes the problem will include a diagram; be sure to still draw your own diagram(s). This step will help you better decipher the problem by visualizing it.
No diagram required with this submission, but draw one if it helps you visualize your solution.

3. Identify Relevant Theory

After determining the information that you know and need to find and drawing a diagram, you should start to have an idea what theory and/or background information you need to solve the problem. In this step you will identify the information (e.g., formulas, data, conversions) that you will need to create a solution to the given problem.

Units of Measure: The units of measure for scoring will be based on the US English system (time – seconds, length – inches, weight – ounces or pounds). All times or physical measurements will be rounded to two decimal places. Conventional rounding will be implemented (<0.5 → round down, >= 0.5 → round up).

Final Score: Final score is calculated based on all the mission scores and the rated aircraft cost (RAC) based on weight and wing span; it is calculated using the following equations.

$$\boldsymbol{Final\ Score = \frac{M1 + M2 + M3}{Rated\ Aircraft\ Cost\ (RAC)}}$$

$$\boldsymbol{RAC = Empty\ Aircraft\ Maximum\ Weight\ (lbs.) * Wing\ Span\ (ft.)}$$

Maximum aircraft empty weight recorded after each successful mission in ounces (oz.); aircraft empty weight does not include the payload but does include any payload supports or restraints and batteries. The highest of these three weights is used to calculate the RAC. For this competition, the maximum legal weight for the aircraft including any cargo (e.g., payload, passengers) is 55 pounds. Wing span is the longest distance between wingtips measured perpendicular to the axis of the fuselage in inches (in). For this competition, the maximum wing span is seven feet.

Mission requirements for all missions: Must complete a successful landing to get a score. A lap is complete when the aircraft passes over the start/finish line in the air (the landing is not part of any time requirements). Any incomplete missions are a 0. The score for each mission is calculated as described in the equations below. Some final scores that teams received in past competitions are shown in Table 1. The top scores for missions 2 and 3 in past competitions are shown in Table 2.

Mission 1 (M1) – no payload – 3 lap timed flight (timed in seconds)

- Teams must complete 3 laps within the flight window (5 minutes)

$$\boldsymbol{M1 = 0.0\ for\ unsuccessful\ mission}$$

$$\boldsymbol{M1 = 1.0\ for\ successful\ mission}$$

Mission 2 (M2) – set payload – 3 lap timed flight (timed in seconds)

- The ***fastest time flown*** is based on the team that received the fastest flight time that year (*this is unknown until all of the teams finish this mission*). Therefore, the team that receives the fastest time that year will receive the highest possible score on Mission 2 of 4.

$$\boldsymbol{M2 = 4 * \left[\frac{fastest\ time\ flown}{time\ flown}\right]}$$

Mission 3 (M3) – long haul of passengers and additional payload

- The score will be a function of the number of passengers and number of payload blocks carried times the number of laps completed (within 10 minutes).
- Each payload unit is a one-ounce block.

- The ***largest total of cargo and laps*** is based on the team that received the highest total that year (*this is unknown until all the teams finish this mission*). Therefore, the team that receives the highest total that year will receive the highest possible score on Mission 3 of 6.

$$\boldsymbol{M3 = 4 * \frac{\#\ of\ passengers * \#\ of\ payload\ blocks * \#\ of\ laps}{(largest\ total\ of\ cargo\ and\ laps)} + 2}$$

Table 1. Top 4 teams' scores over the past 4 years

Team Name:	Mission 1 Scores	Mission 2 Scores*	Mission 3 Scores*	RAC (lb. in.)	Final Score
2018:					
2018_Fly	1.0	1.62	0.0	4.84	0.54
2018_Air	1.0	2.94	2.53	28.71	0.23
2018_Plane	1.0	3.35	4.46	43.56	0.20
2018_Cargo	1.0	3.01	6.00	53.28	0.19
2017:					
2017_Captain	1.0	2.49	4.32	16.53	0.47
2017_Crew	1.0	2.71	2.53	20.95	0.30
2017_Flight	1.0	3.96	3.99	32.78	0.27
2017_AE	1.0	2.93	5.39	42.66	0.22
2016:					
2016_Aerospace	1.0	2.93	1.89	18.48	0.31
2016_Engineer	1.0	3.63	2.43	25.02	0.28
2016_Build	1.0	2.52	4.85	35.26	0.24
2016_Design	1.0	2.79	3.65	36.00	0.21
2015:					
2015_Airplane	1.0	2.74	3.27	19.85	0.35
2015_Flying	1.0	3.33	4.65	28.71	0.31
2015_Jet	1.0	3.78	5.43	42.66	0.24
2015_Aero	1.0	4.00	4.26	39.56	0.23

Table 2. ***Fastest time flown*** **for Mission 2 and** ***largest total of cargo and laps*** **for Mission 3 for past 4 years**

Year	**Mission 2 Fastest Time Flown** (seconds)	**Mission 3 Largest Total of Cargo and Laps** (# of passengers * # of payload * # of laps)
2018	110.12 s.	(1 pass*1 block*9 laps) = 9
2017	106.80 s.	(1 pass*2 block*5 laps) = 10
2016	121.23 s.	(2 pass*1 block*4 laps) = 8
2015	115.36 s.	(1 pass*1 block*8 laps) = 8

4. Assumptions

In this step you will communicate some ideas that you have assumed to simplify the problem. These are things that you may try to address later on to make your solution address a more realistic scenario. It is okay if you cannot envision how you would solve the problem without the assumption, but try to think about this for each assumption that you write.

Write **at least 3** assumptions; most likely you will have more.

5. Solution Steps

This step requires solving for the finds using the givens and theory. For this step you will create equations to solve the problem, but you will not plug in any numbers yet. Be sure to base your equations on variables and NOT numeric values (unless they are a constant, such as 4 in the second mission equation). Be sure to also refer to your diagram and assumptions to help you through this step. At some points in this step you may find it useful to go back and draw another diagram or necessary to use another formula.

Be sure to make your solution adaptable as possible. How can you modify it if the missions get changed? How can you modify it, if the data set gets larger or changes in any other way?

6. Identify Results and Verify Accuracy

Now that you have solve the problem without plugging in values, you will plug in values for this step to verify if your problem is accurate or not.

For submission 1: Test your model for various potential inputs. Do the results make sense? If not, how can you modify your model?

- Submit outputs of your model for **at least the 3 test cases** provided.

For submission 2: Test your code for various potential inputs. Do the results make sense? If not, how can you modify your code?

- Submit outputs of your code for **at least the 3 test cases** provided. You can add these outputs as comments at the bottom of your code or a separate file.

Three Test Cases: *I recommend using more test cases than just the ones provided, especially for testing your code.*

Test Case 1:

- The team's mission 1 results – successful landing and completed the 3 laps within 5 minutes.
- The team's mission 2 results – successful landing and flew the 3 laps in 160.92 seconds
- The team's mission 3 results – successful landing and flew 1 passenger and 1 payload block for 2 complete laps in 10 minutes
- The team's Rated Aircraft Cost:
    - Empty Aircraft Weight (oz.): 30.26 oz.
    - Aircraft Wing Span (in.): 14.25 in.

Test Case 2:

- The team's mission 1 results – successful landing and completed the 3 laps within 5 minutes.
- The team's mission 2 results – successful landing and flew the 3 laps in 118.56 seconds
- The team's mission 3 results – successful landing and flew 1 passenger and 1 payload block for 5 complete laps in 10 minutes
- The team's Rated Aircraft Cost:
    - Empty Aircraft Weight (oz.): 51.26 oz.
    - Aircraft Wing Span (in.): 20.08 in.

Test Case 3:

- The team's mission 1 results – successful landing and completed the 3 laps within 5 minutes.
- The team's mission 2 results – successful landing and flew the 3 laps in 128.34 seconds
- The team's mission 3 results – unsuccessful mission
- The team's Rated Aircraft Cost:
    - Empty Aircraft Weight (oz.): 63.8 oz.
    - Aircraft Wing Span (in.): 29.68 in.

7. Algorithm and Code

Your solution steps will lay out the process that you will need to code. In complex problems that require conditionals and/or repetition, it may be beneficial to draw out a flowchart, concept map, etc. or write out bullet point or numbered steps. Doing this step can ensure you understand the flow of your code before you start writing code in MATLAB.

Your first submission will be all about showing your work through the first six steps and the developed algorithm for this submission.

Keep in mind coding will be required for the second submission, but not the first submission. When you are coding this solution, you can use any coding techniques that you see fit. If you want to stick to if statements and loops you can. If you want to use arrays or programmer-defined functions, you can also do that. Use the code necessary to develop your computational model. *Keep in mind you must validate (or error check) all user inputs for appropriate values; be sure to comment the values that you consider valid and why you chose these values.*

Appendices B. Modeling Problem Rubric (Submission 4)

Criteria	Ratings			Pts
(MODEL) **Reusability:** User Inputs How well does the solution address what user inputs are needed for the model? all addressed: M1 result (either pass/fail, yes/no, land/not land and/or time for flight - some way to evaluate pass/fail), M2 result (time for flight - not score since fastest time for 2019 competition is unknown), M3 result (number of passengers, payload, and number of laps - not score since largest total for 2019 competition is unknown), wingspan and empty aircraft weight (used to calculate RAC)	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(CODE) User Inputs Are the user inputs coded correctly?	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(MODEL) **Reusability:** User Inputs - Error Check Does the student state what values will be considered valid and what will not? Does the student explain why they have determined these values acceptable?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Error Check Are the error checks coded correctly?	**3.0 pts** **Full Marks**	**1.5 pts** some errors in code	**0.0 pts** **No Marks**	3.0 pts
(CODE) Share-ability: User Inputs and Error Check Does the program clearly communicate to the user what data they need for each input (with the units, where relevant)? Does the program clearly communicate the range of valid inputs in input or error messages?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(MODEL) **Reusability:** Assumptions Does the solution have at least 3 clearly communicated assumptions? There should be significantly more, but 3 is the minimum. (Reminder: Assumptions are not facts. Assumptions are things assumed to solve the problem - information that needs to be addressed to move forward, but was not given.)	**1.0 pts** **Full Marks**	**0.5 pts** include 3 bad "assumptions", OR include less than 3 assumptions, but have at least 1 valid assumption	**0.0 pts** **No Marks**	1.0 pts
(MODEL) **Mathematical Model Complexity:** Mission 1 Equation M1 = 1.0 for successful mission (Teams must complete 3 laps and land within 5 minutes to be successful for this mission.)	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Mission 1	**1.5 pts** **Full Marks**	**0.0 pts** **No Marks**		1.5 pts
(MODEL) **Mathematical Model Complexity:** Mission 2 Equation M2 = 4 * ((fastest time flown)/(time flown)) (The time it takes a team to complete 3 laps and land is timed in seconds. The fastest time recorded for that year's competition is used for fastest time flown.)	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts

Criteria	Ratings			Pts
(MODEL) **Mathematical Model Complexity:** Mission 2 Equation - Fastest Time Flown The solution addresses what time they will use for fastest time flown. (Is this based on one fastest time flown? Is this based on an array of possible fastest times flown? Is this based on a predicted future fastest time flown based on the data set?)	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(MODEL) **Modifiability:** Mission 2 Equation - Fastest Time Flown The student clearly communicates where the fastest time flown used came from and why they chose that fastest time. If it was one selected, which one and why? If it was a calculated value, how and why?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Mission 2 – Equation implemented correctly and successfully calculates a Mission 2 score.	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(MODEL) **Mathematical Model Complexity:** Mission 3 Equation M3 = 4*(((number of passengers)*(total payload - oz.)*(number of laps))/((the highest overall value)))+2 (The number of laps completed within 10 minutes is multiplied by the number of passengers and the total payload the plane carries. This highest overall value is the largest value calculated for that year's competition.)	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(MODEL) **Mathematical Model Complexity:** Mission 3 Equation – largest total for cargo and laps (highest overall value) The solution addresses what total they will use for max total or highest overall value. (Is this based on one max total? Is this based on an array of possible max total? Is this based on a predicted future max total based on the data set?)	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(MODEL) **Modifiability:** Mission 3 Equation - Max Total (highest overall value) The student clearly communicates where the max total used came from and why they chose that value. If it was one selected, which one and why? If it was a calculated value, how and why?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Mission 3 – Equation implemented correctly and successfully calculates a Mission 3 score.	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(MODEL) **Mathematical Model Complexity:** Final Score Final Score = (M1 + M2 + M3)/((empty aircraft maximum weight)*(wing span))	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Final Score – Equation implemented correctly and successfully calculates the final score.	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(MODEL) **Mathematical Model Complexity:** Data Set The student took all the available data into account. A significant amount of the data was used to help them evaluate the probability of the user winning.	**3.0 pts** **Full Marks**	**1.5 pts** Minimal data used (less than half of the data was taken into account or at least discussed).	**0.0 pts** **No Marks**	3.0 pts

Criteria	Ratings			Pts
(MODEL) **Modifiability:** They clearly communicated what data they were using and what data they were not. The student clearly communicated why they were or were not using different parts of the data. They student clearly communicated why they decided to use the data the way they did.	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) data file implemented in code correctly Data set is uploaded into code correctly (used xlsread function with correct return variables). The data types are handled in code appropriately (example: working with cell arrays).	**3.0 pts** **Full Marks**	**1.5 pts** some minor errors	**0.0 pts** **No Marks**	3.0 pts
(MODEL) **Mathematical Model Complexity:** Calculate Probability There is a system of determining the odds of the user winning. When evaluating the odds of past winners, the results make sense. There is a fairly large range of possible outcomes based on different inputs. (Not just 0, 50, 100 or something way too over simplified.) There is not ever more than 100% probability of winning. There probably shouldn't ever be 100% probability of winning because there is no guarantee that someone else won't show up and perform better at next year's competition.	**3.0 pts** **Full Marks**	**1.5 pts** Simplified solution. OR Probability can be over 100% OR less than 0%.	**0.0 pts** **No Marks**	3.0 pts
(MODEL) **Modifiability:** Calculating Probability Does the student clearly communicate how and why they chose the various steps for calculating probability?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Calculate and Report out Probability (sample possible errors: at some point two probability values are added together, but one is in a decimal form and the other a percentage so it throws off the final probability; the equation the student developed in their model is clearly different than what they implemented without explanation and it appears the equation was coded wrong because the values are way off.)	**5.0 pts** **Full Marks**	**2.5 pts** some errors in calculating probability.	**0.0 pts** **No Marks**	5.0 pts
(MODEL) **Mathematical Model Complexity:** Recommendations for Improvement There is a system to present recommendations based on the user's provided inputs. What if the team fails a mission? What if the aircraft weight is heavy or wing span long? What if the team has great probability of winning? What about almost none?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(MODEL) **Modifiability:** Recommendations and Improvements Does the student clearly communicate how and why they chose the various recommendations they did based on the user data?	**1.0 pts** **Full Marks**	**0.0 pts** **No Marks**		1.0 pts
(CODE) Determine Recommendations and Report them out	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
(CODE) Share-ability: Outputs Does the program clearly communicate to the user what the results are (predicted final score, probability, and recommendations) (with the units, where relevant)?	**2.5 pts** **Full Marks**	**0.0 pts** **No Marks**		2.5 pts
(CODE) Sample Calculations/Outputs (CODE) The sample outputs for the test cases are provided as comments in the code.	**2.0 pts** **Full Marks**	**0.0 pts** **No Marks**		2.0 pts
				Total Points: 50.0