2015

Does choice of programming language affect student understanding of programming concepts in a first year engineering course?

Benjamin D. McPheron
Roger Williams University, bmcpheron@rwu.edu

Stephanie M. Gratiano
Roger Williams University, sgratiano952@g.rwu.edu

William J. Palm
Roger Williams University, wpalm@rwu.edu

Does Choice of Programming Language Affect Student Understanding of Programming Concepts in a First Year Engineering Course?

Benjamin D. McPheron, Stephanie M. Gratiano, and William J. Palm IV
Roger Williams University, bmcpheron@rwu.edu, sgratiano952@g.rwu.edu, wpalm@rwu.edu

Abstract - **Most undergraduate engineering curricula include computer programming to some degree, introducing a structured language such as C, or a computational system such as MATLAB, or both. Many of these curricula include programming in first year engineering courses, integrating the solution of simple engineering problems with an introduction to programming concepts. In line with this practice, Roger Williams University has included an introduction to programming as a part of the first year engineering curriculum for many years. However, recent industry and pedagogical trends have motivated the switch from a structured language (VBA) to a computational system (MATLAB). As a part of the pilot run of this change, the course instructors felt that it would be worthwhile to verify that changing the programming language did not negatively affect students' ability to understand key programming concepts. In particular it was appropriate to explore students' ability to translate word problems into computer programs containing inputs, decision statements, computational processes, and outputs. To test the hypothesis that programming language does not affect students' ability to understand programming concepts, students from consecutive years were given the same homework assignment, with the first cohort using VBA and the second using MATLAB to solve the assignment. A rubric was developed which allowed the investigators to rate assignments independent of programming language. Results from this study indicate that there is not a significant impact of the change in programming language. These results suggest that the choice of programming language likely does not matter for student understanding of programming concepts. Course instructors should feel free to select programming language based on other factors, such as market demand, cost, or the availability of pedagogical resources.**

Index Terms – First year engineering, Programming, MATLAB, VBA

INTRODUCTION

Many engineering curricula require computer programming to some degree, with the intent of fostering both problem solving skills and the development of engineering procedure. There is little doubt that engineering students benefit from learning to write instructions that a computer can follow, as it can develop students' own understanding of the problem [1]. Various engineering schools include programming using a structured language, such as C++ or Visual Basic for Applications (VBA), or include programming as an integral part of a computational system, such as MATLAB or Mathcad [2]. Computer programming is often included in first year engineering courses, to introduce both the engineering method and simple engineering problems [3]-[4].

For these reasons, the Computer Applications for Engineering course at Roger Williams University (RWU) has long included a programming component. Until recently, this was accomplished using VBA, as it was an extension of Excel, which is the other computer application emphasized in the course [5]. Other motivations for using VBA were the ease of access and the need for no further license beyond Microsoft Office. Key topics presented in the programming unit included input and output, decision points, variable assignment, syntax, and executing correct calculations.

Although structured programming languages like VBA can facilitate teaching these skills, computational systems are becoming more popular in first year engineering courses, and are commonly used by industry professionals [3]-[4]. In addition, VBA is used infrequently in the rest of the RWU engineering curriculum. Furthermore, there is a very small selection of VBA textbooks aimed at engineering education [6]. In short, the VBA unit was not meeting students' needs in terms of jobs, internships, or the other courses in their curriculum.

As a result, other options were explored, and MATLAB emerged as the tool to teach programming concepts for a number of reasons. MATLAB is employed in many later courses in the curriculum, including Circuit Theory, Control Systems, and Finite Element Analysis, among others, as it is at many other institutions [3], [5]. MATLAB has grown in popularity in first year engineering curricula for its utility as both a programming language and a computational system [3]-[4], [7]-[8]. In addition, several studies have explored

the use of MATLAB in engineering education, concluding that MATLAB use is now common practice in first year engineering courses [9]-[11]. One primary benefit of MATLAB is that it allows for regular programming methods to be utilized while providing high level functions like advanced visualization and matrix mathematics [1]. The final motivating piece is that MATLAB is now available to all engineering students at Roger Williams University through a university supported cloud-based virtual desktop service, which removes the barrier of personal software licenses.

The switch to MATLAB was met with reservation from one senior engineering faculty member. He expressed concern that students had struggled with MATLAB in past offerings of the course. Because of this, he had changed from MATLAB to VBA over a dozen years ago. Other faculty members countered that MATLAB has gotten easier to use, and the caliber of our students has improved, in the intervening years. After further discussion, the majority of the faculty favored switching back to MATLAB, and so a pilot trial was given the green light.

As a result of these thoughtful discussions, the course instructors felt that it would be worthwhile to study the impact of the change in programming language on student understanding of programming. Other studies of this sort have been done [12]-[15], but none has considered this exact change of languages. In addition, none of these studies is recent, reflecting the updates to programming interfaces and languages. The most closely related study was performed by Cortina, et al. in 1997 [14], nearly twenty years ago. Another benefit to the present study is that it provides the instructors with feedback on the efficacy of teaching programming using MATLAB. Finally, this study can provide a service to faculty members at other institutions who are unsure of which programming language or computational system to employ in introductory classes.

To study the effect of changing programming languages, students from the spring 2015 offering of Computer Applications for Engineering were taught programming concepts using MATLAB, and were given several of the same programming problems from the spring 2014 offering, in which students were taught VBA. Two evaluators independently scored the assignments of both cohorts using a rubric developed to assess student ability in key areas while remaining agnostic of programming language. The key hypothesis for this study was that switching from VBA to MATLAB would not negatively impact students' ability to learn key programming concepts.

METHODS

The course examined in this study is the second semester engineering course at Roger Williams University. The 2014 cohort learned programming concepts using VBA as an extension of Excel, while the 2015 cohort used MATLAB. The students in these courses were given the same engineering programming problems, which required application of the programming concepts of input and output, decision points, variable assignment, syntax, and correctly executing calculations. Figure 1 shows an excerpt of one of these programming problems, which asks students to perform calculations related to *Water Treatment*. The second problem, displayed in Figure 2, asks students to write a program used for compression *Spring* design.

Create a program that does the following:

a) Asks the user to enter the initial concentration of microorganisms, the time at which they would like to calculate the microorganism concentration, and whether they would like to determine the concentration using a zero, first, or second order equation. Your program should work no matter whether they enter "Zero" or "zero" or "ZERO" or "zErO", etc.
b) Computes the concentration of organisms using the appropriate equation:

Zero Order: $N(t) = N_0 - 21.8 * t$

First Order: $N(t) = N_0 * e^{-0.125*t}$

Second Order: $\frac{1}{N(t)} = \frac{1}{N_0} + 0.00349 * t$

Where the INPUTS are the following:

- Initial number of organisms per liter of water (N_0)
- Time in minutes (t)
- Which equation they would like to use (zero order, first order, or second order)

And the OUTPUT is the following:

- Concentration of organisms N at time t

FIGURE 1
WATER TREATMENT PROGRAMMING PROBLEM USED IN THIS STUDY.

Create a program to aid an engineer in designing a spring.

Inputs:

- The wire diameter, d (a real number, where 0.01 <= d <= 0.25 inches)
- The mean coil diameter, D (a real number, must be > 0 inches)
- The total number of coils, N_t (a real number, must be >= 3)
- The free length L_{free} (a real number, must be > 0 inches)
- The spring end type (could be *plain*, *closed*, *ground*, or *closed & ground*)
- The spring material (could be *steel* or *stainless steel*)
- The deflection, x (a real number, must be >= 0 inches)

Constants:

- The shear modulus, G (11,800,000 psi for steel, or 10,000,000 psi for stainless steel)

Results:

- The spring constant, k
- The solid height, L_{solid}
- The shear stress, τ
- The factor of safety, n

FIGURE 2
SPRING PROGRAMMING PROBLEM USED IN THIS STUDY.

To assess students' ability to apply these programming concepts, a rubric was developed to score student work. The rubric was developed to be independent of the programming language used, and value assigned to each rubric topic was based on the problem requirements. The maximum available score for correct completion of both problems was 35 points. This rubric is displayed in Table 1. Two evaluators used the rubric to independently score the work of each student. One evaluator was a course instructor for two sections in 2015, while the second investigator was not a course instructor. The use of a non-instructor is

TABLE I
RUBRIC USED BY EVALUATORS.

Water Treatment					
Program runs		2 if yes	0 if no		
Correct number of inputs		3 if 3	2 if 2	1 if 1	0 if 0
Number of decisions		3 if 3	2 if 2	1 if 1	0 if 0
Decisions correct		4 if all	2 if 2	1 if 1	0 if 0
Correct calculation		2 if yes	0 if no		
Correct number of outputs		2 if yes	0 if no		
Spring					
Program runs		2 if yes	0 if no		
Correct number of inputs		3 if all 7	2 if 4-6	1 if 1-3	0 if 0
Number of decisions		3 if all 7	2 if 4-6	1 if 1-3	0 if 0
Decisions correct		7 if all 7	4 if 3-6	2 if 1-3	0 if 0
Correct calculation		2 if yes	0 if no		
Correct number of outputs		2 if yes	0 if no		
Total Score					

intended to reduce the chance of scoring bias. The evaluators' scores were tabulated and compared to assess inter-rater reliability. The Pearson's *r* correlation between the two evaluators' scores was 0.68 (0.69 for the MATLAB assignments and 0.67 for the VBA), indicating good agreement. The two evaluators' scores were averaged to yield each student's overall score.

SAMPLE DEMOGRAPHICS

The 2014 cohort was composed of 83 students from four course sections, of whom 67 completed both programming problems examined in this study. The 2015 cohort included 96 students from four course sections, of whom 84 completed both problems. Possible reasons why some students did not complete both problems could include difficulty with programming, but also time management issues or apathy. Because we are specifically interested in the effect of programming language on student understanding, and not on student motivation or interest, we chose to include only the students who completed both problems in this study. Further analysis, including even the students who did not complete both problems, did not change the principal findings.

In terms of academic preparation, the two cohorts are very similar. The average high school GPA for students in 2014 was 3.45, while that of the 2015 students was 3.43. The average Math SAT score was 594 for each cohort. In addition, the average college GPA of students entering the course in 2014 was 3.18, compared to 3.15 in 2015.

Despite these striking similarities, there are some demographic differences in the cohorts. The 2014 group was composed of 85.1% male students and 14.9% female students, while the 2015 sample included 77.4% male students and 22.6% female students. This difference is not statistically significant (Fisher's exact test, $p = 0.30$). Additionally, the 2014 cohort had 16.4% international students, versus only 8.3% in 2015. This difference too is not statistically significant ($p = 0.14$). Nonetheless, the potential impact of these differences is discussed in the next section.

RESULTS & DISCUSSION

The results from scoring the MATLAB homework problems from the 2015 cohort, and the same problems completed in VBA by the 2014 cohort, show that MATLAB had a higher overall mean score than VBA (Table 2). Use of Student's *t*-test suggests that this difference in means is not statistically significant (*p*-value of 0.095, two-tailed). This result supports our hypothesis that changing the programming language from VBA to MATLAB would not adversely impact student understanding of key programming concepts taught in the course. Indeed, it may have had a positive effect.

The distribution of overall scores for both MATLAB and VBA is shown in Figure 3. Only one student scored

TABLE 2
OVERALL MEANS, STANDARD DEVIATIONS AND *T*-TEST RESULTS FOR MATLAB AND VBA SCORING.

	Mean Score	**Standard Deviation**
2015 Cohort (MATLAB)	29.15	4.28
2014 Cohort (VBA)	28.00	4.10
***p*-value for *t*-test for difference of means**	0.095	

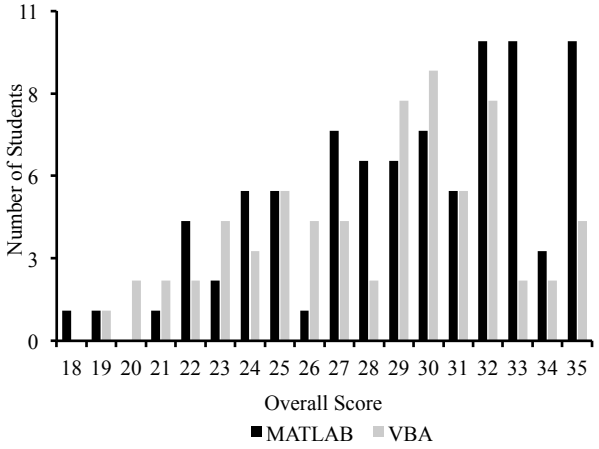

FIGURE 3
DISTRIBUTION OF OVERALL SCORES FOR STUDENTS USING MATLAB AND VBA.

TABLE 3
MEAN SCORES GIVEN FOR EACH RUBRIC TOPIC BY EACH EVALUATOR FOR EACH PROBLEM IN BOTH MATLAB AND VBA.

		Water Treatment							*Spring*							
		Program runs	Ask for the correct number of inputs	Correct number of decision statements	Decision statements correct	Correct calculation	Correct number of outputs	Total – Water Treatment	Program runs	Ask for the correct number of inputs	Decision statement for each input	Decision statements correct	Correct calculation	Correct number of outputs	Total – Spring	Overall Score
MATLAB	**Evaluator 1**	1.60	2.91	2.97	2.79	0.74	1.68	11.00	1.10	2.97	2.88	5.39	0.65	1.72	14.70	27.42
	Evaluator 2	1.71	2.94	2.98	3.70	1.27	1.74	12.60	1.51	2.88	2.98	5.88	1.59	1.58	16.42	30.89
VBA	**Evaluator 1**	1.42	2.86	2.95	1.69	0.49	1.76	9.40	1.69	3.00	2.99	5.02	0.74	1.77	15.20	26.72
	Evaluator 2	1.85	2.55	2.82	3.44	1.47	0.99	12.13	1.66	2.97	2.88	5.39	1.25	1.84	15.98	29.28

below 18 points in either MATLAB or VBA. More students received a perfect score of 35 points in MATLAB than in VBA. In addition, 39% of students using MATLAB scored greater than 31 points, compared with 24% of those using VBA.

Figure 4 shows the mean score for each rubric topic on a percentage basis. Student performance on the various rubric topics was generally consistent between both homework problems and both programming languages. For both MATLAB and VBA, students had difficulty getting their program to make the "Correct calculations," most likely due to not having "Decision statements correct." For both programming languages, students excelled in providing the "Correct number of inputs" as well as having the "Correct number of decision statements."

For the *Water Treatment* problem, students using MATLAB received scores greater than or equal to students using VBA on all rubric topics. This trend is also visible in the *Spring* problem; however, students faced more difficulty in getting their program to run in MATLAB for this problem. This is likely the result of very little time spent teaching students to debug their code in the 2015 offering of the course, coupled with the fact that the *Spring* problem was more complex than the *Water Treatment* problem.

Table 3 presents the mean scores assigned by each evaluator to each rubric topic. Results for overall scores show that both evaluators rated student performance higher for MATLAB than VBA. Evaluator 1 consistently rated lower than Evaluator 2, but both evaluators' scores show the same trends with respect to rubric topic, programming language, and homework problem. For two rubric topics, ("Decision Statements Correct" and "Correct Calculation"), Evaluator 1 scored noticeably lower than Evaluator 2. This is probably due to the assignment of lower scores for "Decision Statement Correct" by Evaluator 1 and the assumption that "Decision Statements Correct" is directly related to "Correct Calculation".

One question that emerges when reviewing these results is whether the improved performance observed with MATLAB might be due to differences in student characteristics, instructor, or other factors that differed between years. As noted in the sample demographics section, the 2015 cohort had more female students and fewer international students than the 2014 cohort. In addition, the course instructors were not the same from 2014 to 2015. One instructor was common, teaching two sections of the course in each year. A second instructor taught two sections in 2014, while a third taught two sections in 2015.

To address this question, ordinary least squares regression was used to determine the effect of programming language on homework problem performance, while controlling for student gender, nationality, and instructor.

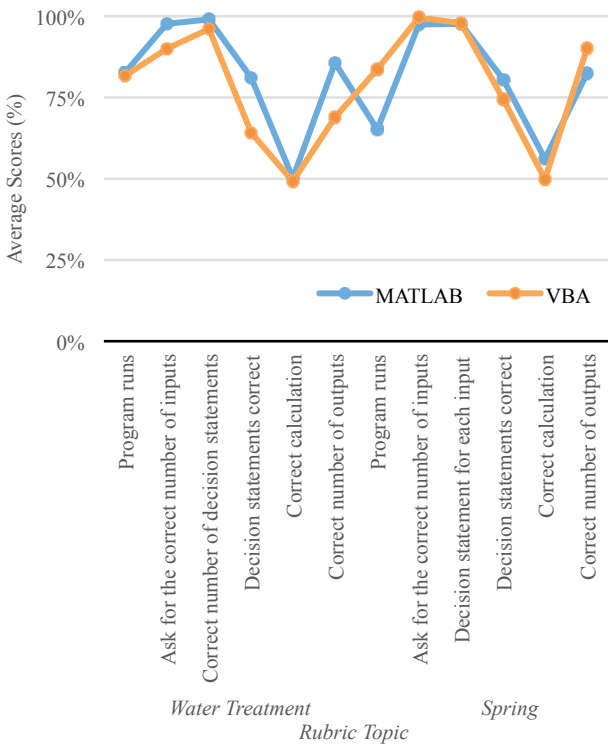

FIGURE 4
MEAN SCORE FOR EACH RUBRIC TOPIC ON A PERCENTAGE BASIS

TABLE 4
RESULTS OF REGRESSION MODEL USED TO CONTROL FOR STUDENT DEMOGRAPHICS AND INSTRUCTOR

	b	*t*	*p*
Intercept	30.318	29.7	< 0.001
MATLAB	0.535	0.58	0.565
Gender (1 = male)	-2.286	-2.66	0.009
International	-1.686	-1.59	0.115
Instructor 2†	-0.224	-0.22	0.828
Instructor 3†	0.441	0.49	0.627
$R^2 = 0.084$, $N = 151$, $F = 2.65$ ($p = 0.025$)			

† The instructor who taught in both 2014 and 2015 is the base case, so for him $b = 0$.

Results of the regression model are shown in Table 4. The *b*-values in the second column represent the coefficients in a linear equation to estimate a student's overall score. The *p*-values indicate the likelihood that each coefficient differs from 0 due to random sampling error. The results indicate that even after controlling for differences in student demographics and instructors, students still performed slightly better using MATLAB, though the effect is not significant ($p = 0.565$). Male students performed significantly worse than female students, while international students performed somewhat worse but not significantly so. The course instructor had little impact.

CONCLUSIONS

This study examined the question of whether switching from VBA to MATLAB to introduce programming in a first-year computer applications course would affect student understanding of key programming concepts. The results suggest that switching languages had no significant effect.

This finding is supported by several aspects of the study design. First, it included all students who completed the relevant assignments from two years of the course. Second, an objective rubric and two independent evaluators were used to score the students' programs. The evaluators' scores were well-correlated, and the average of their scores was used in analysis. Finally, although the two cohorts were fairly similar in terms of demographics, a regression analysis was used to control for the subtle differences that did exist, as well as the different instructors teaching the course. This analysis confirmed the lack of a significant impact of programming language.

The primary limitation of the study is that it was conducted at a single institution, with a relatively modest sample size. It is possible that the slight improvement seen when using MATLAB is a real effect, but a larger sample size would be needed to confirm this. The second limitation is that the study was not a perfectly controlled experiment. Although the course design and lessons were generally similar between years, there were some minor variances. For example, in 2014 the *Water Treatment* and *Spring* problems appeared on separate homework assignments, whereas in 2015 they appeared on the same assignment. In addition, the students used different textbooks for VBA and MATLAB. It is possible that differences such as these could mask a true difference inherent in the programming languages themselves. However, given that our primary objective is assessment and improvement of our engineering curriculum, the finding that the entire 2015 course design (programming language, textbook, lesson content, etc.) performed no worse than the 2014 course is valuable.

This study inspires several questions that could be addressed in future work. Perhaps the most interesting would be to evaluate student perceptions of the programming languages. Our 2015 course evaluations hint that the students may have found MATLAB more challenging than the students found VBA in 2014, despite the fact that their performance was actually slightly better. Did the perceived difficulty prompt them to work harder to learn the programming concepts? Or are perceived difficulty and programming performance independent of each other? Another question would be whether the students retain their understanding of programming concepts and language syntax better in one language or another. For most students, their next significant use of programming will occur in Circuit Theory during the fall of the junior year, 1.5 years after they took the Computer Applications course. Circuit Theory currently uses MATLAB; will the students who learned MATLAB as freshmen perform significantly better than those who learned VBA? We would like to think so, but most readers are familiar with students' ability to "unlearn" material if not used recently. We made the switch to MATLAB in part because it is used frequently in the upperclass curriculum, but if the introduction to MATLAB in the freshman year is not retained by the junior year then this motivation may be immaterial.

The results found in this study suggest that choice of programming language likely does not matter for student understanding of programming concepts. It is probable that instructors of First Year Engineering courses can choose a programming language based on external factors, such as availability, use in other courses, or instructor proficiency, and see no measurable dip in student understanding. Instructors looking to make a change in programming language should be encouraged to do so, provided they are willing and able to develop materials for the new language.

ACKNOWLEDGMENT

We would like to thank our colleague Nicole Martino for her assistance with data collection, as well as our students for the use of their homework submissions.

REFERENCES

[1] Herniter, M.E. Scott, D.R. and Pangasa, R., "Teaching programming skills with MATLAB", *Proceedings of the 2001 American Society for Engineering Education Conference & Exposition*, 2001.

[2] Hodge, B.K. and Steele, W.G. "A survey of computational paradigms in undergraduate mechanical engineering education", *Journal of Engineering Education*, October 2002 (415-417).

[3] Attaway, S., "Using MATLAB to teach programming to first-year engineering students at Boston University", *MathWorks News & Notes* 2010.

[4] Yokomoto, C.F., Rizkalla, M.F, O'Loughlin, C.L., El-Sharkawy, M.A. and Lamm, N.A., "Developing a motivational freshman course in using the principle of attached learning", *Journal of Engineering Education*, January 1999 (99-106).

[5] Clough, D.E., Chapra, S.C. and Huvard, G.S., "A change in approach to engineering computing for freshmen-similar directions at three dissimilar institutions", *Proceedings of the 2001 American Society for Engineering Education Conference & Exposition*, 2001.

[6] Chapra, S.C., "Introduction to VBA for Excel", 2nd Edition, Prentice Hall, 2009.

[7] Collier, N. and Kaw, A., "On comparing computational systems – Maple, Mathcad, Mathematica & MATLAB", *Computers in Education Journal* Volume XIV, Number 1, January-March 2004.

[8] Liu, Y., "A programming course including C# and MATLAB for mechanical engineering students", *Computers in Education Journal* Volume XXI, Number 3, July-September 2011.

[9] Ibrahim, A., "Special issue – MATLAB and Simulink in engineering education (II)", International Journal of Engineering Education, 21 (5), 768-768.

[10] Wallin, H.P., Carlsson, U., Ross, U., El Gaidi, K., "Learning MATLAB: Evaluation of methods and materials for first-year engineering students", International Journal of Engineering Education, 21 (4), 692-701.

[11] Colgan, L. "MATLAB in first-year engineering mathematics", International Journal of Mathematical Education in Science and Technology. Vol. 31, Iss. 1, 2000.

[12] Ho, C. and Raubenheimer D., "Computing-related self-efficacy: the role of gender, academic performance, and computational capabilities", *Proceedings of the 2011 American Society for Engineering Education Conference & Exposition*, 2011.

[13] Thomas Jr., G.E., Minnick, M.V. and Gang, D. "Evolution of a freshman software tools class", *Proceedings of the 2005 American Society for Engineering Education Conference & Exposition*, 2005.

[14] Cortina, T.J., "A quantitative approach for choosing a procedural programming language in freshman programing", *Proceedings of the 1997 American Society for Engineering Education Conference & Exposition*, 1997.

[15] Brannan, K.P. and Murden, J.A., "From C++ to Mathcad: teaching an introductory programming course with a non-traditional programming language", *Proceedings of the 1998 American Society for Engineering Education Conference & Exposition*, 1998.

AUTHOR INFORMATION

Benjamin D. McPheron
Assistant Professor of Engineering, Roger Williams University, bmcpheron@rwu.edu

Stephanie M. Gratiano
Undergraduate Research Assistant, Roger Williams University, sgratiano952@g.rwu.edu

William J. Palm IV
Assistant Professor of Engineering, Roger Williams University, wpalm@rwu.edu