2023 Annual Conference & Exposition
Baltimore Convention Center, MD | June 25 - 28, 2023
The Harbor of Engineering Education for 130 Years
Paper ID #38288

Biomedical and Agricultural Engineering Undergraduate Students Programming Self-Beliefs and Changes Resulting from Computational Pedagogy

Ms. Joreen Arigye, Purdue University

Joreen Arigye is a Ph.D. student in the School of Engineering Education at Purdue University. She holds a M.S. in Information Technology from Carnegie Mellon University and a B.S. in Software Engineering from Makerere University. Her research interests include computational modeling, data analytics, and computation in STEM Education.

Dr. Alejandra J. Magana, Campbell University

Alejandra J. Magana, Ph.D., is the W.C. Furnas Professor in Enterprise Excellence in the Department of Computer and Information Technology with a courtesy appointment at the School of Engineering Education at Purdue University. She holds a B.E. in Informa

Mr. Joseph A. Lyon, Cornell University

Joseph A. Lyon is a Lecturer for the College of Engineering Honors Program at Purdue University. He holds a Ph.D. in Engineering Education. His research interests are computational thinking and mathematical modeling.

Elsje Pienaar

Evidence-based practice: ASEE

Biomedical and Agricultural Engineering Undergraduate Students Programming Self-Beliefs and Changes Resulting from Computational Scaffolding

Background: The growing demand for computing skills in all science and engineering-related fields begs the question of how college graduates in science and engineering can be best equipped with computational thinking and computer programming skills. Therefore, computational practices need to be integrated into the science and engineering curricula sooner and more often.

Purpose: This study investigated undergraduate students pursuing biomedical and agricultural engineering majors and the changes in their self-beliefs about programming for approaching engineering problems. Specifically, we wanted to understand if the students' self-beliefs changed as a result of implementing three two-week-long computational assignments throughout the semester facilitated through computational scaffolding. The computational scaffolding was embedded within computational notebooks and was grounded in evidence-based practices aligned with cognitive apprenticeship methods.

Methods: The study was conducted in a second-year thermodynamics course offered at a large Mid-Western University. The objective of the course was to understand and exploit basic principles of thermodynamics as they apply to biological systems and biological processes and model these processes using computer code. Pre and post-data were collected using a survey instrument at the beginning and the end of the course. The survey instrument captured students' perceptions toward five aspects related to their experience with programming, i.e., self-efficacy, self-concept, interest, anxiety, and aptitude mindset. 100 students who completed both surveys were considered for the final analysis.

Results: Based on the constructs used to capture students' programming experience, i.e., self-efficacy, self-concept, interest, anxiety, and aptitude mindset, results indicate an average positive increase in only programming self-efficacy. The rest of the constructs maintained a neutral or undecided position.

Implications: The study indicates that undergraduate engineering students reported a neutral or undecided experience during programming in the computational modeling course, specifically for programming self-concept, interest, anxiety, and aptitude mindset. These findings can be potentially useful for implementing course interventions to improve engineering students' experience.

1. Introduction

Engineering equips students with the ability to use their mathematical and scientific principles to build models of real-world systems and to simulate their behavior which allows them to understand complex phenomena, innovate around them, and even make predictions. Modeling and simulation then becomes a fundamental skill set across engineering disciplines. Multiple calls have been made for increased incorporation of modeling and simulation in science and engineering classrooms [1], [2]. Clark and Ernst [3] further emphasize that by having courses that link science and mathematics to technology through the development of both computation and physical models, STEM content integration can take place for students. Many times, however, these practices can be difficult for engineering students to learn [4] and for engineering faculty to teach [1]. As such, computational modeling skills and practices are often undertaught by instructors and underdeveloped among graduating students.

Fortunately, work in engineering and physics education has started to document effective ways for delivering computation instruction through scaffolding, e.g., [4]–[7]. Even with these strides, research has indicated that incorporating computational modeling and simulation can lead to "cognitive overload" from having to learn and model different representations, such as physical, mathematical, and algorithmic, on top of the programming challenges. [8], [9].

This study investigates the effects of computational modeling and simulation, where students reported their levels of caring and enjoyment before and after modeling exercises. In particular, the pre and post-survey data capture students' perceptions of their programming self-efficacy beliefs, self-concept beliefs, levels of anxiety, aptitude mindset, and interest. This leads to the following research question: Do students' perceptions of their own computational abilities change after participating in computational modeling and simulation projects?

2. Theoretical Framework

The theoretical framework that guided the design of the learning intervention and the focus of our research design was grounded in the theory of the Zone of Proximal Development (ZPD). The ZPD was proposed by Lev Vygotsky as a sociocultural theory that describes learning and development [10]. The ZPD conceives learning as the space between what a learner can do without assistance and what the learner can do with competent assistance. A common way to translate implications from the ZPD to the design of learning interventions is by providing students with scaffolding. Scaffolding refers to all types of support and guidance offered in the classroom either by the instructor or peers or supported by technology [11].

In the context of higher education, scaffolding refers to teaching techniques or tools that support students' learning. Students are provided with learning supports that help them accomplish tasks that they normally would not be able to accomplish on their own or will pose a significant challenge. As students acquire specific knowledge and skills, those supports are eventually removed as they can apply the learning skills independently.

In the context of engineering education practice, providing students with scaffolding is highly recommended when the faculty is not available to provide help (i.e. while solving a homework assignment or projects outside of the classroom). Specifically, in the context of computational assignments, scaffolding methods can involve (a) short video lectures explaining difficult concepts, (b) worked-out examples demonstrating and explaining difficult calculations or implementations of a particular function, (c) templates of code that can get students started with implementing their computational solutions, and (d) test cases for evaluating computational solutions [12].

Research studies have also explored the link between providing students scaffolding on difficult tasks and how those have enhanced students' self-efficacy beliefs [13]. An efficacy belief refers to the conviction of an individual that they can successfully execute a behavior required to achieve a specific goal, action, or outcome [14]. Self-efficacy beliefs are essential in the learning process because they result in agency, control, and intention to pursue courses of action [15].

The constructs used in this study are defined as follows based on Scott and G. Ghinea's instrument [16]. Self-efficacy captures "learners' cognitive self-assessment of whether or not they are confident in their ability to write and debug simple programs" [p. 125]. Self-concept is "a composite of self-perceptions that one can be a good programmer, which is formed through experience with and interpretations of one's environment" [p. 125]. Interest is "the extent to which an individual enjoys engaging with programming-related activities"[p. 124]. Anxiety is the "self-reflected state of experiencing negative emotions, such as nervousness or helplessness while writing and debugging programs" [p. 125]. The programming aptitude mindset represents "the strength of a learners' belief in the notion of a fixed programming aptitude (e.g., aptitude is inherent and cannot change)" [p. 125].

The implications of the theoretical framework for this study then relate to the integration of scaffolding approaches to support the development of computational practices and how those experiences may improve students' self-beliefs in their programming abilities.

3. Methods

This intervention study employed quantitative methods to answer the research question, which focused on identifying the changes in students' perceptions of their own computational abilities after a computational modeling activity.

3.1. Context and participants

The context of the study was a second-year thermodynamics course offered at a large Mid-Western University in the USA. The objective of the course was to understand and exploit basic principles of thermodynamics as they apply to biological systems and biological processes and model these processes using computer code. Pre and post-data were collected using a survey instrument at the beginning and the end of the course. The survey instrument captured students' perceptions toward five self-beliefs related to their experience with programming, i.e., self-efficacy, self-concept, interest, anxiety, and aptitude mindset.

The population considered for the final analysis consisted of 100 students that completed both surveys. According to institutional data, in 2021-2022, about 56% of the students pursuing biomedical engineering majors were women, and about 44% of the students were men. The majority of the students were White 67%, followed by Asian 21%, International 10%, more than two races 6%, Hispanic or Latino 3% and Black or African American 2%. The students were organized into a total of 24 teams, each with four or five members.

3.2. Learning Design

The intervention in this study aimed to facilitate the acquisition of both disciplinary knowledge and computational skills, consisting of three two-week-long computational projects implemented throughout the semester. The projects were titled; Could you outrun a Dinosaur (P1), Toxin-Antitoxin system design (P2), and Chemical Reactor Stability and Sensitivity (P3).

Computational notebooks were used to deliver scaffolding methods since they provide the platform to include detailed explanations, guidance, and scaffolding throughout the project solution. Computational notebooks are defined as computational essays that use text, along with code programs, interactive diagrams, and computational tools to express an idea [7]. The importance of computational notebooks is to provide programming environments for developing and sharing educational materials, combining different types of resources such as text, images, and code in a single document accessible through a web browser [17]. These are specific ways in which the projects were scaffolded to guide students:

- The tasks for each project were broken down into smaller sub-tasks. For example, as shown in Table 1 below, the sub-tasks included planning, collecting data, defining functions, performing calculations, and visualizing results.
- A detailed outline or a step-by-step guide was provided for each sub-task. This guide provided clear instructions on what the students needed to do along with examples such as code snippets and embedded video guides.
- Pre-written code was provided for necessary coding sub-tasks. The code would be partially complete, with some placeholders or comments for students to fill in. This aim was to help students understand the structure of the code and provide a starting point of what they needed to do at each step.
- Students were prompted to provide explanations for their visualizations to articulate their understanding and knowledge of the relevant sub-tasks. The aim was to reinforce students' learning and improve their ability to communicate these complex ideas.
- For the coding sub-tasks, students were required to add meaningful comments to their code. The aim was to communicate their thought process and code structures and to help students collaborate by ensuring that team members can easily read and follow the code.
- When applicable, sample data was provided.

The computational learning objectives for the projects were to:

- Organize and input data efficiently
- Visualize data by plotting arrays
- Perform simple calculations and computations using arrays
- Utilize linear modeling for data analysis
- Utilize built-in tools to numerically create, solve, and visually represent ordinary differential equations.

- Utilize for-loops to iterate over arrays of parameters and carry out computations.
- Calculate steady-state values for state variables by utilizing built-in tools and functions.

Table 1 presents specifics of the learning objectives and the high-level tasks for each of the projects. The projects contained planning, coding, task reflection, and assignment reflection.

Table 1. Overview of learning objectives and tasks for each of the projects

	P1. Could you outrun a dinosaur	**P2. Toxin-antitoxin system design**	**P3. Chemical reactor stability and sensitivity**
Objectives	Collect and visualize data accounting for noise and uncertainty Compute and interpret dimensionless quantities Interpret and analyze data and resulting dimensionless quantities	Describe complex biological systems using models of genetic circuits Characterize and describe dynamics in a given system of biological Interactions Evaluate and test possible system structures to achieve a stated goal	Construct and analyze mass and energy balances Incorporate endo- and exothermic reactions into mass and energy balances Interpret and characterize systems at, and away from, steady state Predict operating conditions to achieve a stated goal in a bioreactor
Planning	Before you start - plan your solution	Before you start - plan your solution	Before you start - plan your solution
Task #1	Collect data.	System of differential equations that describe the dynamics of the biological system	Mass balance at steady state
Task #2	Plot velocity as a function of stride length	Predict what the dynamics of receptor, toxin, and antitoxin levels are over time Reflection on results from Task 2	Energy balance at steady state
Task #3	Plot velocity as a function of s/l	Include 1 or 2 regulatory modules for activation of anti-toxin production Reflection on results from Task 3	Characterizing the steady state behavior of the system for an isothermic reaction Reflection on steady-state analysis of isothermic reactions
Task #4	Transform velocity to a dimensionless form	Find a combination of up to 3 gene regulatory modules that can meet the design criteria Reflection on results from Task 4	Characterizing the dynamic behavior of the system for an isothermic reaction Reflection on not in steady-state analysis of isothermic reactions
Task #5	Plot dimensionless velocity as a function of s/l		Characterizing the steady state behavior of the system for exothermic reactions Reflection on steady-state analysis of exothermic reactions
Task #6	Fit a line through your data		Dynamics toward different steady states with a fixed value of τ Reflections on dynamics and multiple steady state observations
Task #7	Calculate the velocity for your dinosaur		
Task #8	Advise your fellow group members on operation dino egg		
Reflection	Post assignment reflection	Post assignment reflection	Post assignment reflection

3.3. Data Collection

A pre-survey and post-survey regarding caring and enjoyment of computation were administered at the beginning and end of the semester, respectively. The specific survey instrument used was the Scott and Ghinea's [16] instrument to assess student self-beliefs in CS1. The survey consisted of a 5-point Likert scale ranging from strongly disagree (1 point) to strongly agree (5 points), capturing students' perceptions of their programming self-efficacy beliefs, self-concept beliefs, level of anxiety, aptitude mindset, and interest.

3.4. Data Analysis

The study used descriptive and inferential statistics to analyze the data in order to answer the research questions. The survey questions were analyzed quantitatively by deriving the difference between the individual pre and post-test scores for each student for each construct, i.e., self-efficacy, self-concept, interest, anxiety, and aptitude mindset. The differences between the pre and post-test scores were then compared using a t-test to infer any significant differences. 100 students that filled out both the pre-and post-course survey were considered for the analysis.

4. Results

The results of the analysis indicate that statistically significant differences were observed in all the questions under student's reported self-efficacy, two questions under the reported self-concept, one question under reported anxiety, and one question under reported aptitude mindset as shown in Table 3 below.

In total 19 different tests were conducted and 9 of them were statistically significant. One limitation of running multiple hypothesis tests can be an increased chance of making a type I error. But since we primarily wanted to identify individual significant results and we did not have a strict requirement for maintaining an overall level of significance, we did not implement the Bonferroni correction.

The results indicate that reported self-efficacy had the most significant changes. Two questions under reported self-concept i.e. "I am just not good at programming" and "In my programming labs, I can solve even the most challenging problems" were significant. One question under reported interest i.e. "I am interested in the things I learn in programming activities" was significant. One question under reported anxiety i.e. "I get nervous when trying to solve programming bugs" was significant and one question under reported aptitude mindset i.e. " To be honest, I do not think I can really change my aptitude for programming" under reported aptitude mindset was significant.

Table 3. Pretest and Post-test transition tags for self-belief constructs

Self-beliefs Construct		Pre Mean, SD	Post Mean, SD	T statistic P-value
Self-efficacy	I am confident that I can understand Python scripts	3.11, 1.14	3.87, 0.88	6.21, <0.0001
	I am confident I can solve simple problems with my programs	3.77, 0.84	4.14, 0.74	3.65, 0.0004
	I am confident I can implement a method from a description of a problem or algorithm	3.36, 0.92	3.83, 0.82	4.11, <0.0001
	I am confident I can debug a program	3.38, 0.94	3.87, 0.91	4.627, <0.0001
Self-concept	I am just not good at programming	2.72, 1.12	2.37, 1.00	-3.07, 0.002
	I learn programming quickly	3.36, 0.98	3.27, 1.03	-0.90, 0.368
	I have always believed that programming is one of my best subjects	2.31, 1.08	2.44, 1.16	1.28, 0.201
	In my programming labs, I can solve even the most challenging problems	2.49, 1.03	2.87, 1.12	2.29, 0.024
Interest	I enjoy reading about programming	2.43, 1.03	2.40, 1.04	-0.225, 0.822
	I do programming because I enjoy it	2.76, 1.11	2.94, 1.17	1.59, 0.114
	I am interested in the things I learn in programming activities	3.69, 0.77	3.38, 0.98	-2.97, 0.003
	I think programming is interesting	3.86, 0.73	3.76, 0.89	-1.13, 0.259
Anxiety	I often worry that it will be difficult for me to debug my program	3.39, 1.06	3.17, 1.12	-1.84, 0.067
	I often get tense when I have to debug a program	2.98, 1.11	2.97, 1.18	0.07, 0.940
	I get nervous when trying to solve programming bugs	3.04, 1.14	2.74, 1.20	-2.13, 0.035
	I feel helpless when trying to solve programming bugs	2.89, 1.09	2.64, 1.07	-1.87, 0.064
Aptitude mindset	I have a fixed level of programming aptitude, and not much can be done to change it	1.89, 0.72	2.05, 0.85	1.72, 0.088
	I can learn new things about programming, but I cannot change my basic aptitude for programming	2.32, 0.92	2.42, 0.96	0.82, 0.410
	To be honest, I do not think I can really change my aptitude for programming	1.86, 0.74	2.06, 0.76	2.09, 0.038

5. Discussion and Implications

This study investigated whether students' perceptions of their own computational abilities change after participating in computational modeling and simulation projects, which are captured as students' perceptions of their programming self-efficacy beliefs, self-concept beliefs, level of anxiety, aptitude mindset, and interest. The overall findings suggest only students' perceptions of their programming self-efficacy beliefs increased. For the rest of the constructs, although some changes were observed in specific questions within each of the constructs, they were not consistent. Thus, we can conclude that students’ perceptions of their self-concept, interest, anxiety, and aptitude mindset remained undecided or neutral after the computational modeling and simulation projects.

The self-efficacy beliefs construct consists of the student's confidence to understand Python scripts, the student's confidence to solve simple program problems, the student's confidence to implement a method from a description of a problem or algorithm, and the student's confidence to debug a program. Self-efficacy can be a key factor in students' academic success and future career choices in engineering. Self-efficacy defined as "one's self-judgment concerning capability", is an important mediating factor in cognitive motivation [18]. In engineering, students with high levels of self-efficacy tend to have better problem-solving skills, greater resilience in the face of challenges, and more positive attitudes toward their coursework and future careers [19].

Another important aspect of self-efficacy is its relationship to the retention of women in engineering. Self-efficacy can play an important role in the success and persistence of women in engineering. Research shows a mixed view of women's engineering self-efficacy and gender differences for engineering self-efficacy, even though researchers tend to agree that self-efficacy is an important concept in academic pursuits and career decisions [20].

The implications of this study relate to the use of scaffolding methods to support students in their learning processes, particularly as related to computational assignments. The findings suggest that the scaffolding delivered via the computational notebooks was sufficient to help students succeed in completing their computational projects and developed more confidence in their programming skills.

6. Conclusion, Limitations, and Future Work

This study found that participating in computational modeling and simulation projects can positively impact students' perceptions of self-efficacy in computational tasks. Improved confidence in programming during these projects can have a positive impact on students' attitudes toward engineering and potentially increase retention rates in the field. These are encouraging findings for engineering educators at all levels.

However, the study has some limitations, such as having a smaller sample size and focusing only on perception which is only part of the larger story. As part of future work, demographic information can be used as a covariate for further analysis of self-efficacy given that prior experiences play a heavy role in student self-efficacy. Future studies could consider alternative methods to the ones presented here to gain a more comprehensive understanding of the impact of different types of computational scaffolding on students' self-beliefs in engineering.

Acknowledgments
This work is based upon efforts supported by the EMBRIO Institute, contract #2120200, a National Science Foundation (NSF) Biology Integration Institute. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NSF or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] A. J. Magana and G. Silva Coutinho, “Modeling and simulation practices for a computational thinking-enabled engineering workforce,” *Computer Applications in Engineering Education*, vol. 25, no. 1, pp. 62–78, Jan. 2017, doi: 10.1002/cae.21779.

[2] D. E. Penner, “Chapter 1: cognition, computers, and synthetic science: building knowledge and meaning through modeling,” *Review of research in education*, vol. 25, no. 1, pp. 1–35, 2000.

[3] A. C. Clark and J. V. Ernst, “STEM-Based Computational Modeling for Technology Education.,” *Journal of Technology Studies*, vol. 34, no. 1, pp. 20–27, 2008.

[4] J. Gainsburg, “The mathematical modeling of structural engineers,” *null*, vol. 8, no. 1, pp. 3–36, Jan. 2006, doi: 10.1207/s15327833mtl0801_2.

[5] H. Fennell, J. A. Lyon, A. Madamanchi, and A. J. Magana, “Computational apprenticeship: Cognitive apprenticeship for the digital era,” 2019.

[6] R. P. Medeiros, G. L. Ramalho, and T. P. Falcão, “A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education,” *IEEE Transactions on Education*, vol. 62, no. 2, pp. 77–90, 2019, doi: 10.1109/TE.2018.2864133.

[7] T. O. Odden, E. Lockwood, and M. Caballero, “Physics computational literacy: An exploratory case study using computational essays,” *Physical Review Physics Education Research*, vol. 15, Dec. 2019, doi: 10.1103/PhysRevPhysEducRes.15.020152.

[8] A. J. Magana, M. L. Falk, and M. J. Reese, “Introducing Discipline-Based Computing in Undergraduate Engineering Education,” *ACM Trans. Comput. Educ.*, vol. 13, no. 4, Nov. 2013, doi: 10.1145/2534971.

[9] C. Vieira, A. J. Magana, R. E. García, A. Jana, and M. Krafcik, “Integrating Computational Science Tools into a Thermodynamics Course,” *Journal of Science Education and Technology*, vol. 27, no. 4, pp. 322–333, Aug. 2018, doi: 10.1007/s10956-017-9726-9.

[10] L. S. Vygotsky and M. Cole, *Mind in society: Development of higher psychological processes*. Harvard university press, 1978.

[11] N. Boblett, “Scaffolding: Defining the metaphor,” *Studies in Applied Linguistics and TESOL*, vol. 12, no. 2, 2012.

[12] C. Vieira, A. J. Magana, A. Roy, and M. Falk, “Providing students with agency to self-scaffold in a computational science and engineering course,” *Journal of Computing in Higher Education*, vol. 33, pp. 328–366, 2021.

[13] P. Yantraprakorn, P. Darasawang, and P. Wiriyakarun, “Enhancing self-efficacy through scaffolding,” *Proceedings from FLLT*, 2013.

[14] A. Bandura, “Self-efficacy: toward a unifying theory of behavioral change.,” *Psychological review*, vol. 84, no. 2, p. 191, 1977.

[15] R. M. Klassen and E. L. Usher, “Self-efficacy in educational settings: Recent research and emerging directions,” *The decade ahead: Theoretical perspectives on motivation and achievement*, vol. 16, pp. 1–33, 2010.

[16] M. J. Scott and G. Ghinea, “Measuring enrichment: the assembly and validation of an instrument to assess student self-beliefs in CS1,” in *Proceedings of the tenth annual conference on International computing education research*, 2014, pp. 123–130.

[17] A. Cardoso, J. Leitão, and C. Teixeira, “Using the Jupyter Notebook as a Tool to Support the Teaching and Learning Processes in Engineering Courses,” in *The Challenges of the*

Digital Transformation in Education, M. E. Auer and T. Tsiatsos, Eds., Cham: Springer International Publishing, 2019, pp. 227–236.

[18] M. K. Ponton, J. H. Edmister, L. S. Ukeiley, and J. M. Seiner, “Understanding the Role of Self-Efficacy in Engineering Education,” *Journal of engineering education (Washington, D.C.)*, vol. 90, no. 2, pp. 247–251, 2001.

[19] R. W. Lent, S. D. Brown, and G. Hackett, “Toward a unifying social cognitive theory of career and academic interest, choice, and performance,” *Journal of vocational behavior*, vol. 45, no. 1, pp. 79–122, 1994.

[20] R. M. Marra, K. A. Rodgers, D. Shen, and B. Bogue, “Women Engineering Students and Self-Efficacy: A Multi-Year, Multi-Institution Study of Women Engineering Student Self-Efficacy,” *Journal of engineering education (Washington, D.C.)*, vol. 98, no. 1, pp. 27–38, 2009.