Surveying civil engineering student attitudes toward the use of computational tools

Dr. Sotiria Koloutsou-Vakakis, University of Illinois at Urbana - Champaign

Dr. Sotiria Koloutsou-Vakakis holds a Diploma in Surveying Engineering (National Technical University of Athens, Greece), a M.A. in Geography (University of California, Los Angeles), and M.S. and Ph.D. degrees in Environmental Engineering (University of Illinois Urbana-Champaign). She is a Senior Lecturer and Research Scientist in the Department of Civil and Environmental Engineering, at the University of Illinois Urbana-Champaign. Her main interests are in air quality, environmental policy, and supporting student learning and professional preparation.

Dr. Megan L. Matthews, University of Illinois Urbana - Champaign

Dr. Megan L. Matthews is an Assistant Professor in the Department of Civil and Environmental Engineering at the University of Illinois Urbana-Champaign. Her work involves developing multiscale computational models to explore plant engineering strategies for sustainably improving the security and equity of global food, water, and environmental systems. Megan earned her bachelor's, master's, and doctorate in electrical engineering from North Carolina State University.

Prof. Jacob Henschen, University of Illinois Urbana-Champaign

Professor Henschen completed his B.S., M.S., and PhD. at the University of Illinois Urbana-Champaign in 2007, 2009, and 2018 respectively. He was an Assistant Professor at Valparaiso University until he moved to the University of Illinois Urbana-Champaign as a Teaching Assistant Professor in June 2020. He serves as the co-chair for the Teaching Methods and Education Materials Committee at ACI and the co-chair of the Committee on Faculty Development at ASCE.

Prof. John S Popovics P.E., University of Illinois at Urbana - Champaign

John Popovics is a Professor in the Civil and Environmental Engineering Department at the University of Illinois at Urbana-Champaign. He earned his B.S. and M.S. in Civil Engineering from Drexel University and his Ph.D. in Engineering Science and Mechanic

Dr. Ashlynn S. Stillwell, University of Illinois Urbana-Champaign

Dr. Ashlynn S. Stillwell is an Associate Professor and the Elaine F. and William J. Hall Excellence Faculty Scholar in Civil and Environmental Engineering at the University of Illinois Urbana-Champaign. Her research focuses on creating sustainable water and energy systems in a policy-relevant context. She earned a B.S. in Chemical Engineering from the University of Missouri (2006), and an M.S. in Environmental and Water Resources Engineering (2010), M.P.Aff in Public Affairs (2010), and Ph.D. in Civil Engineering (2013) from The University of Texas at Austin.

Cheryl Ann Cohen

Surveying civil engineering student attitudes toward the use of computational tools

Sotiria Koloutsou-Vakakis, Megan L. Matthews, Cheryl Cohen, Jacob Henschen, John S. Popovics, Ashlynn S. Stillwell

Department of Civil and Environmental Engineering, University of Illinois Urbana-Champaign, 205 N. Mathews, Urbana, IL 61801, sotiriak@illinois.edu

Abstract

The discussion about integrating computing into curricula of non-computer science engineering majors remains open. Such integration is desirable, as computing is an essential tool for engineering practice. In addition, computing can enhance engineering education through simulation and visualization, facilitating deeper understanding and learning. Integration of computing using high-level programming languages into courses presents challenges including achieving alignment with traditional disciplinary learning objectives, the potential need to shift from a traditional constructivist approach to a more student-driven model, and the necessity for faculty transition into using new tools and pedagogies which requires career-long training. This paper originates from a department-wide effort to integrate computing throughout the curriculum. The success of such an effort depends primarily on coordinated faculty involvement and student engagement. Focusing on the latter, in this paper, we report on our approach for integrating coding for data analysis and problem solving, and on surveyed student attitudes toward the integration of Python or R in undergraduate civil and environmental engineering courses. We report data from one required first-year course (serving as a reference point), one required second-year course and two required third-year courses within a four-year civil and environmental engineering undergraduate program. These courses are representative of the efforts to scaffold the integration of computing throughout the curriculum.

In the beginning of the effort to integrate Python and R coding into courses, student resistance and lack of engagement seemed to be major obstacles based on the signals we received from the student body. Therefore, we began with straightforward, guided coding assignments in the early weeks of a semester, gradually integrating coding into redesigned assignments on the core material of the courses. Our goal was to integrate coding without altering a course's learning objectives. At the same time, we surveyed student attitudes, as we implemented these changes in each participating course twice, once in the beginning and once at the end of a semester. In the beginning-of-semester surveys, many students identify coding as a major challenge. In the end-of-semester self-assessment responses, overall and over time, we see an increasing level of acceptance of computing as part of the civil and environmental engineering learning culture and experience. The responses indicate a moderately increasing trend for student willingness to independently choose Python or R for future courses and projects. The survey responses suggest that as computing becomes normalized, negative feelings among students become less of an impediment. Comparisons of grades among classes in semesters before and after the computing integration do not generally show statistically significant differences. Given the lack of relevant data available, both in our department and in the literature, these survey responses provide valuable insights into civil engineering students' attitudes toward coding for data analysis and problem-solving, which could assist others considering similar curricular changes.

1. Introduction

Rapid advances in computational capacity, ability to process massive amounts of data, artificial intelligence, automation, and new ways of visualizing and experiencing (virtual and augmented reality) disrupt technological, market, and societal norms. In the educational and professional environments these developments create a need for university "graduates with transferable skills, such as creativity and adaptability, as well as skills in teamwork, communication, conflict resolution, and problem solving" [1].

Translating this stated need into university curricula is challenging for teachers and students. It is common experience in recent years that updates to educational technology tools during a student's first year could be outdated by the time the student graduates. Beyond the ongoing effort to keep up, educational inequalities emerge across courses due to differences in instructors' skills, attributes, and attitudes, which range from pioneering creators to early adopters, to unwilling change-embracers [2]. It is evident that approaches to teaching engineering continue to evolve. The key question persists about how educators can overcome our own new knowledge deficit to best help engineering students cultivate a growth mindset for navigating technological disruptions, without inducing stress or anxiety about rapidly shifting norms.

In this paper, we present initial findings on student attitudes toward the use of interpreted, high-level object-oriented programming languages such as Python [3] and R [4] in the problem-solving process within coursework. This study is part of a departmental initiative to incorporate computation and computational thinking into the curriculum by integrating computational tools with course fundamentals. This effort commenced just before the rapid emergence of ChatGPT [5] in late 2022. Since we only have anecdotal evidence about AI's impact, we defer discussing this topic to a future study. The insights here are based on surveys designed to collect baseline information about student attitudes toward computational tools in their courses, and to explore whether these have changed over time in select courses, considering both lower level to higher level courses.

2. Background

The general framework for our effort to integrate computation and computational thinking is grounded in our department throughout the curriculum. The departmental curriculum committee serves as the primary community of practice (CoP) for ongoing department-wide curriculum innovation. The effort focuses on three key areas: computational thinking, communication, and teaching Civil and Environmental Engineering (CEE) in a societal context. To ensure scalability and sustainability, course-specific CoPs were established, with one instructor designated as the course director. The course director coordinates efforts to develop common learning objectives and pedagogies for a course, regardless of the instructor. Course materials are created through collaborative efforts within these course-specific CoPs and are continuously refined across semesters to adapt to evolving knowledge and technologies. In some courses, we have adopted a student-centered teaching model [6]. This initiative aligns with the Universal Design for Learning (UDL) guidelines, to foster purposeful and motivated learners through engagement, sustained effort, persistence, and self-regulation [7]. Moving toward a constructivist approach, where knowledge is actively constructed by the learner [8], alters traditional student-teacher interactions and may influence student affect and perceptions of learning outcomes [9, 10]. It also adds complexity to managing both in-class and out-of-class educational time. At this stage, the CoPs play a crucial role in addressing instructors' need for a support system [11]. Creating course-specific CoPs aims to sustain the involvement of all participants in the educational process.

Next, we summarize the scholarly framework within which this project develops. We could not find consensus in the literature on the meaning and use of terms 'computation' and 'computational thinking'. In this context, we use *computation* to refer to tools such as spreadsheets and high-level interpreted programming languages, which our students are taught to use for problem-solving. *Computational thinking* is a broader concept encompassing a variety of skills. A frequently cited definition is Wing's description of computational thinking as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" [12, 13]. Expanding on Wing's definition computational thinking has been described as a problem-solving process that includes (but is not limited to): (1) Formulating problems in a way that it is possible to use a computer and other tools; (2) Organizing and analyzing data logically; (3) Representing data through abstractions; (4) Automating solutions; (5) Identifying, analyzing and implementing possible solutions in order to obtain the most effective combination; and (6) Generalizing and applying this process to a wide variety of problems [14]. A 2021 National Academies of Science, Engineering, and Medicine report [15], emphasizes that computing is more than just coding or computer science, extending to "a larger set of foundational knowledge and competencies". The development of computational thinking skills is framed as inclusive of skills essential for students' professional preparation including communication, collaboration, creativity, critical thinking, and computing [16].

Although there is broad agreement that computational thinking is much more than coding, in our experience a fear of coding can be a roadblock for many students in non-computer science (CS) majors. As acknowledged in [15], even though authentic learning experiences in computing designed to closely mirror professional practices may engage some learners, "historical inequities in computing, biases, and stereotypes may also make these kinds of experiences unattractive to learners from communities that have typically been excluded from computing". It is therefore recommended to (1) Carefully design authentic experiences to ensure they are contextually appropriate, targeted at specific outcomes, and considerate of learners' prior interests and experiences in their homes and communities; and (2) Establish clear and explicit programmatic goals, with continuous refinement to maintain alignment with those goals. In addition, a deficit of studies is identified for assessing learner growth in computational knowledge [15, 17]. Magana et al., provide further insights [18], reporting that college students exposed to teaching that implements modeling and simulation, often encounter obstacles related to students' inability to map the relationships among the physical phenomena, the mathematical representations, and the computational representations. Based on research from the learning sciences, they recommend that learners need to engage in cognitive and metacognitive processes including reflection and evaluation of the importance of each step in the modeling or simulation process [18]. Surveying student attitudes, as in our work, motivates additional reflection.

3. Approach

We have reported on earlier efforts to integrate computational thinking in our two 2nd-year required civil engineering courses [19, 20]. The current efforts focus on the 3rd-year courses required for CEE areas that students major in.

Even though a course CoP determines how and when during a semester integration of R or Python commences, the general approach is that (1) the effort occurs stepwise over a number of semesters; (2) a learning platform is chosen that facilitates grading of code (e.g., Prairie Learn (https://www.prairielearn.com), Google Colab (https://colab.research.google.com), Gradescope (https://www.gradescope.com)); (3) coding is integrated with the subject matter of the course and assignments are redesigned to integrate the use of R or Python (depending on the kind of problems solved in a course).

We administer two student surveys, one at the beginning (early') and the second at the end of a semester. The surveys serve three purposes: (1) to learn about student attitudes toward computation; (2) to help us refocus on our educational goals and adjust our methods based on expressed student needs, and (3) to help us improve the survey questions for clarity and aligning them better with our objectives. In this paper, we report on student responses from the initial set of surveys, focusing on their attitudes toward the required use of computational tools. Negative emotions, such as fear or anxiety, can adversely affect student engagement and learning, making this a key area of interest. In the following section, we present data from one course each in the 1st and 2nd years of study and two courses from the 3rd year, highlighting insights from the student responses. The surveys are anonymous, approved as exempt by the Institutional Review Board. Based on the exempt protocol, students were required to sign and informed consent form before they took a survey.

4. Insights from student surveys about student attitudes toward coding

First transition to integrating coding for computational tasks

As mentioned earlier, the effort to introduce Python and R in the curriculum started with two required 2nd year courses. At that time, students entering these courses had previously taken a 1st year required CEE course where use of spreadsheets (Microsoft Excel or Google Sheets) is taught, and a computer science course (hereafter CS 101) where the basics of programming and Python are taught for non-CS engineering students. In our anecdotal experience, students expressed fear, dislike, and resistance to the idea of coding as something difficult and unnecessary for the CEE profession. We start this section with a review and comparison of indicative responses to two anonymous midsemester surveys taken in the last semester Excel was used as a required computational tool and the first semester when R was introduced in one of the 2nd year courses for probability and statistics calculations, data analysis and visualization. This course is introductory on risk and uncertainty, referred to as the RU course.

Focusing on the computation-relevant questions and responses, Table 1 contains summaries of anonymous responses for comparison of perceptions and student feedback, before (N=96) and after (N=87) R was introduced in the RU course. We present a qualitative summary of these comments which includes the most frequent comments we used in our continuing efforts to improve student sentiments and learning outcomes after coding integration. During the earlier semester, exams were on paper precluding the use of Excel for exams. Because of this assessment approach, some students considered homework assignments requiring use of Excel as an unnecessary waste of time. It was obvious that unless a computational tool was also part of the exam assessment process, some students would not consider it as an essential part of their learning in a course. Therefore, in our subsequent introduction of coding using R, we adopted an online platform that allowed code grading, and we redeveloped course assignments to integrate coding in the problem-solving process. Furthermore, from our experience, we were aware that coding would bring some student resistance, thus good preparation was essential. To transition to use of R coding, we revised course materials aiming at (1) balancing challenge and support, (2) improving CEE relevance by modifying assignments to have CEE context, and (3) streamlining course procedures for class time and study time. Specifically, we:

- Created handouts to clarify course expectations and workflows.
- Transitioned to a learning management system, that allows code submission for grading.
- Rebuild the class-time worksheets assignments and homework to include problems with CEE context.
- Created guided computational assignments in R.
- Added more practice problems.
- Introduced a class project requiring use of R for data analysis and visualization.
- Adopted a flipped class approach.
- Organized the class in groups working together during class and for the class project.

Table 1. Comparison of student feedback related to computation and computational tools between the two semesters before and during transition to R (RU course).

<table>
<tr><th colspan="2">Student responses - Focus on responses referring computational tools</th></tr>
<tr><th>(before R: MS Excel taught – exams on paper)</th><th>(R introduced: exams in an off-the-web environment, coding exam questions possible)</th></tr>
<tr><td>Excel
• Excel-based assignments were mentioned as less helpful by some students.
• Students requesting elimination of assignments requiring use of Excel, perceiving them as a waste of time, since they could not use Excel for exams.

Application
• Students desired more applied problems that directly connect computational skills to CEE.

Teamwork on Computational Problems
• Several students found working on worksheets or computational problems in groups beneficial.</td><td>R
• Difficulties with coding, using R and interacting with R, translating everyday wording to probability notation and to code.
• Probability-related computation identified as challenging and disorienting.
• R identified as useful for interval calculations and statistical computations.

Application
• Requests for more computational examples or guides to help understand concepts.
• Many responses emphasized the relevance of computational skills (e.g., statistics and coding) to engineering tasks.

Teamwork on Computational Problems:
• Students appreciated working in groups to resolve computational issues.</td></tr>
</table>

Characteristic student quotes (unedited)	Characteristic student quotes (unedited)
"Less excel, don't feel it's helping understanding the material just time consuming." "I wish we had shorter excel assignments, they always take a long time." "The homework questions have been tricky because there are no hints/examples to look at if I don't know how to do it." "I would like more time to work on the worksheet and less time on lecture slides and Excel tasks." "The computations are great, really helped my understanding of the material." "More focus on computation would be nice." "I understand the concept, but calculations still take too long for me to feel confident." "Working with teammates in class, Extending in-class teamwork." "Review sessions are really good for understanding the concepts." "The worksheets really support my learning." "Group work in class helped me a lot."	"I do not like R." "R is hard. Maybe some in-class examples of R?" "R is pretty useful when calculating intervals, but I still find it challenging." "The thing that would make my R experience the best is if there were more hands-on examples in class." "Example problems in class help me understand the material more clearly." "The difficulty of understanding how to use the concepts in real-world scenarios." "Understanding the details of probability was tough." "There are many small concepts that build up and can be overwhelming." "Review sessions are very helpful, and working on problems with peers helps me learn better." "Group work has helped me overcome challenges, especially when brainstorming and discussing solutions together.".

We performed a comment-level sentiment analysis of open-text student responses, using three lexicons AFINN [21], BING [22] and NRC [23]. Each lexicon has its own word collection and design rationale giving it certain strengths and limitations [24]. For example, AFINN provides intensity-based sentiment scoring, whereas BING provides binary polarity analysis (positive-negative), and NRC provides emotional granularity (e.g., anticipation, disgust, anger, joy). Figure 1 shows a visual comparison of the sentiment analysis scores for the questions shown at the top of each graph. Due to space limitations, we show simplified sentiment score graphs where a positive score indicates positive sentiment (e.g., joy, confidence, anticipation), while a negative score indicates a negative sentiment (e.g., anger, fear, anxiety). Despite the differences among lexicons, overall, it appears to be a less negative / more positive sentiment in the semester R was introduced compared to the previous semester, with indication of a change in students' perceptions of the course relevance to CEE, likely related to the intentional addition of problems with CEE context that occurred in parallel with the introduction of R. However, Figure 2 shows that answers to the question 'Are you feeling at ease and stimulated by the learning community in this class?' student NO/YES answers were 20/80% in the earlier semester versus 30/70% in the semester R was introduced. It is not possible to tell if this outcome was the effect of introducing coding or the effect of differences in the student communities in the two semesters, but this was informative feedback as we continued with material development and effort to make the course more interactive and engaging.

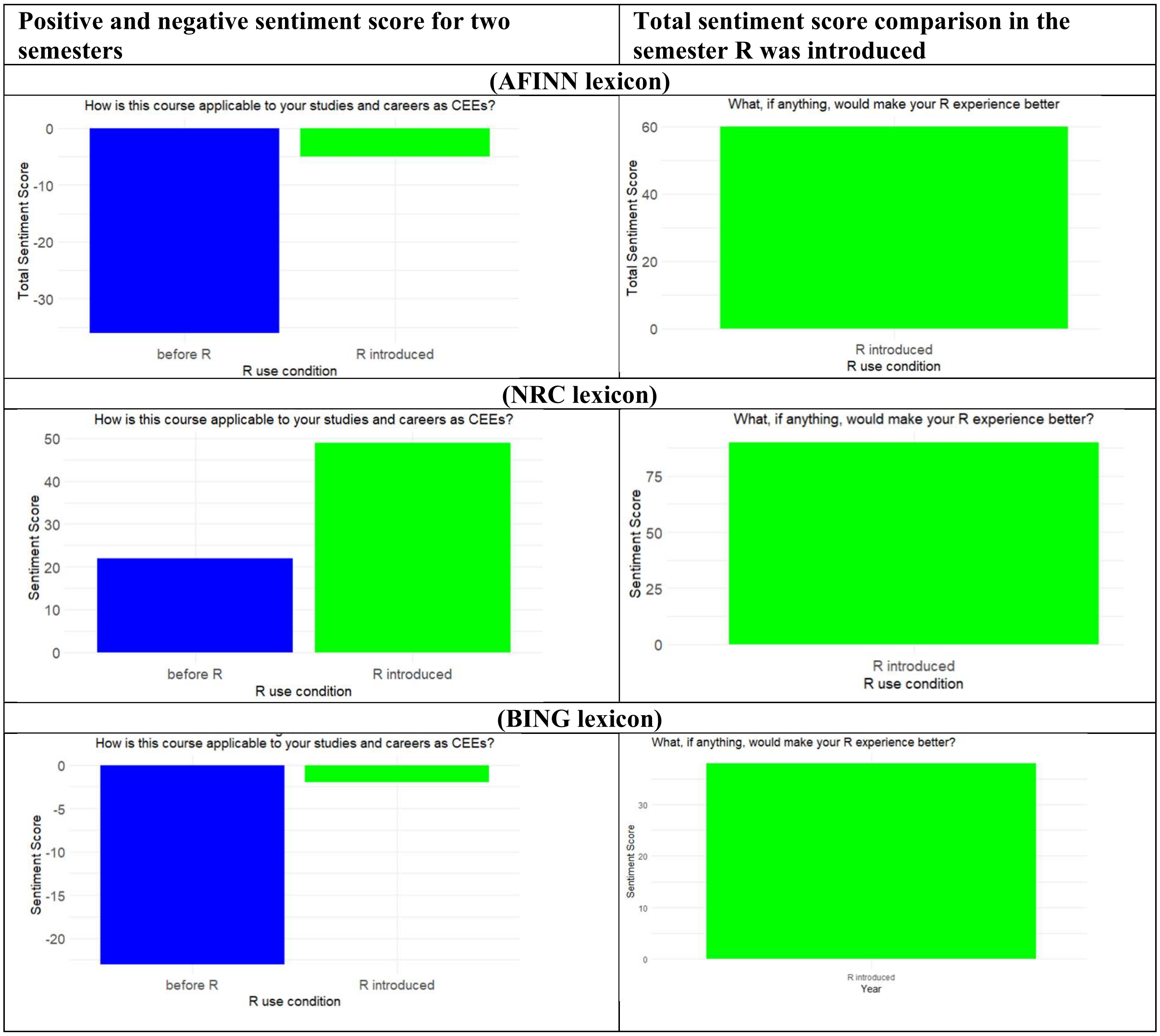


Figure 1. Sentiment analysis of student responses to midsemester surveys relevant to computation and computational tools. Data collected between the two semesters before and during transition to R. Positive and negative total score values indicate positive and negative student sentiments, respectively (RU course).

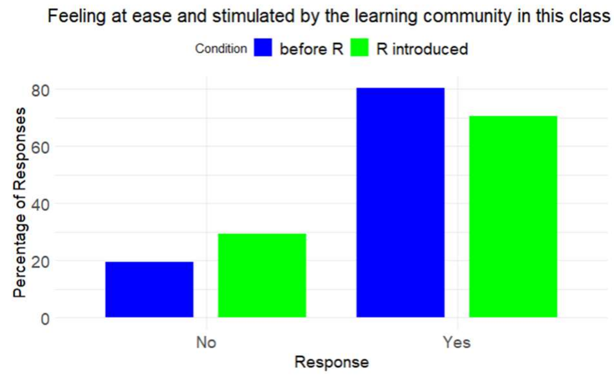


Figure 2. Student responses (YES/NO) to the question 'Are you feeling at ease and stimulated by the learning community in this class?' asked in midsemester surveys during the two semesters before and during transition to R (RU course).

Continuing coding integration in subsequent semesters

As we continued to develop and integrate coding with core course materials, there is an indication of a changing culture. For example, in Figure 3, for a semester sequence, the left panel shows that the vast majority of students had no previous experience with the coding language (R in this case), at the beginning of the course. While this remained true across subsequent semesters, at the end of the semesters (right panel) the expressed level of confidence increases as use of the coding language becomes the norm for the class. As there was no drastic change in the materials or teaching methods in these semesters (by comparison to the first transition semester), we expect the right panel depicts a change in the acceptance of coding as part of the CEE learning experience.

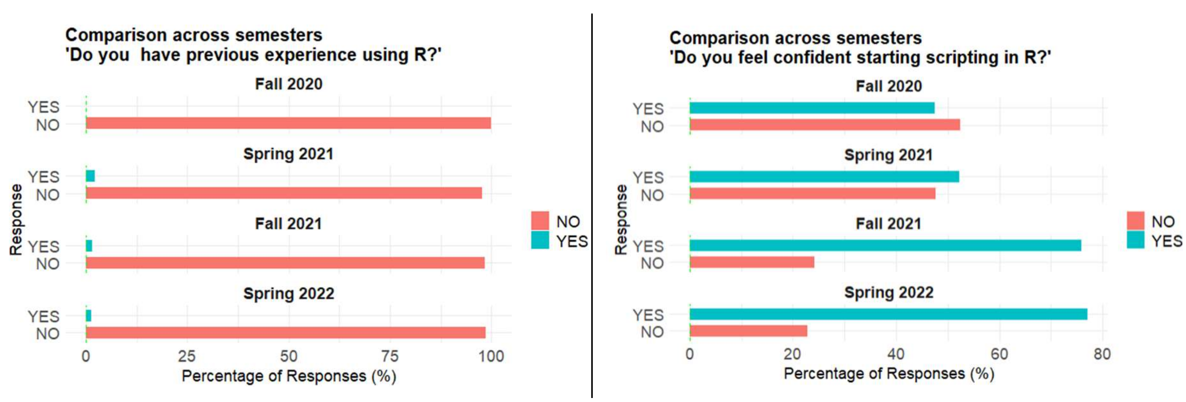


Figure 3. Responses to the questions 'Do you have previous experience with R?' (left panel, early semester surveys) and 'Do you feel confident starting scripting in R?' (right panel, end-of-semester surveys).

Expanding use of coding in the 3rd year courses

Next, we present results from student responses across 4 courses. We use the 1st year course as a reference. This is a project based (PB) course introducing incoming students to the different areas of CEE. The 2nd year course is the RU course mentioned earlier, which is offered twice a year. The other two courses are the earliest adopters of coding for problem solving at the 3rd year level. The topic of the first 3rd year course is on the behavior of materials (referred to as MB course, henceforth). The second 3rd year course (referred to as the GE course) is on the global environment and energy. Two case studies also in review for this conference provide further details on the two 3rd yr courses that have been included for discussion here [25], [26]. Table 2 and Figure 4 summarize the number of responses per course and semester and the coding background for prerequisite courses, respectively. Figure 5 displays coding experience of responding students before they took one of the courses included in this presentation.

Figures 6a to 6e display students' responses to questions asked to help us understand the change in CEE student attitudes toward coding and computational tools. Over time, qualitatively we see that responses shift to a recognition of the value of coding skills and computational tools, which is a significant change from the past, when most CEE students were expressing distress about having to learn and use coding.

Table 2. Survey response rates by course included in this paper in recent semesters (F: Fall, S: Spring). 'Early' and 'End' indicate early and end-of semester surveys, respectively.

Course	1st yr PB F23 Early	1st yr RU F23 End	2nd yr RU S23 Early	2nd yr RU S23 End	2nd yr RU F23 Early	2nd yr RU F23 End		3rd yr MB S24 Early	3rd yr MB S24 End	3rd yr GE F23 Early	3rd yr GE F23 End
Comp. tool used	Excel		R					Python		R	
Nenrolled	207		105		84			89		61	
Nrespond.	186	151	79	51	66	44		80	31	38	31
(%) response	90	73	75	49	79	52		90	35	62	51

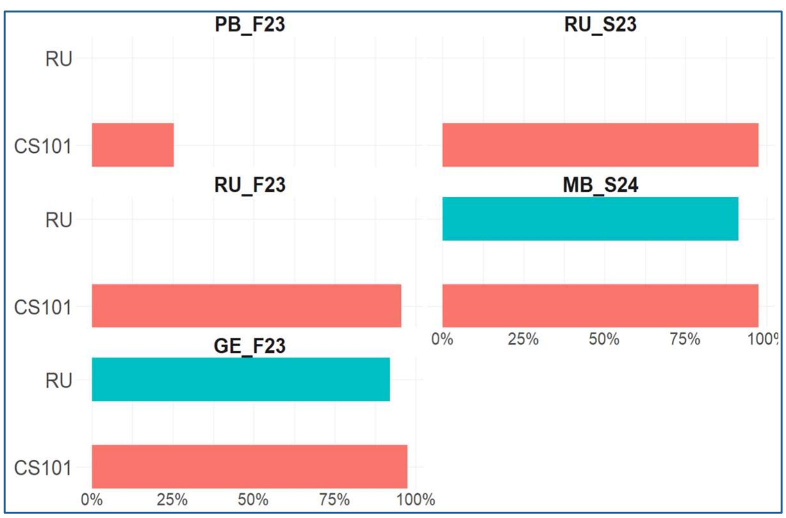


Figure 4. Student percentages who had taken a prerequisite course where Python (CS 101-orange) or R (RU-green) are introduced. RU is applicable to 3rd-year courses only. The prerequisite courses are shown in the vertical axis. (Recent semesters, F: Fall, S: Spring). Almost all students in the 3rd year courses MB and GE had taken at least one of the coding prerequisites.

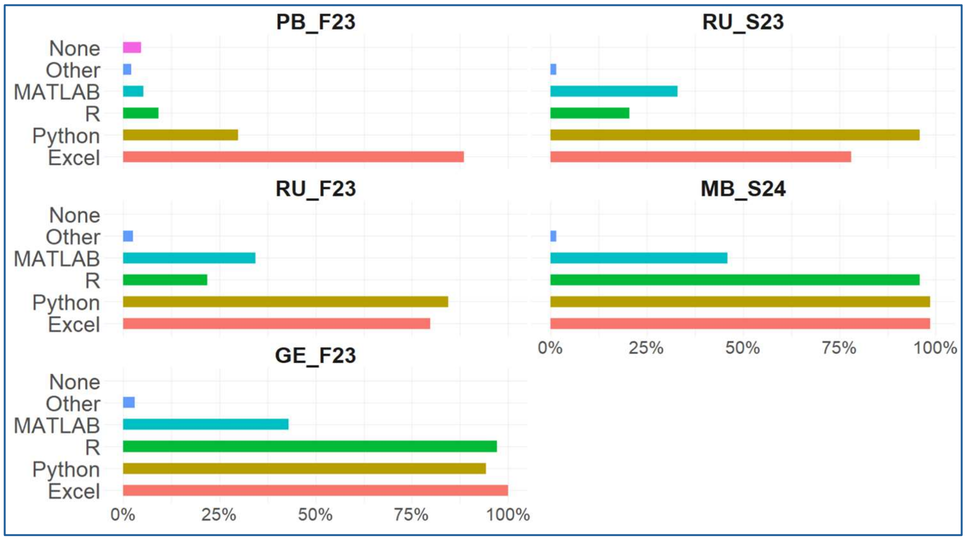
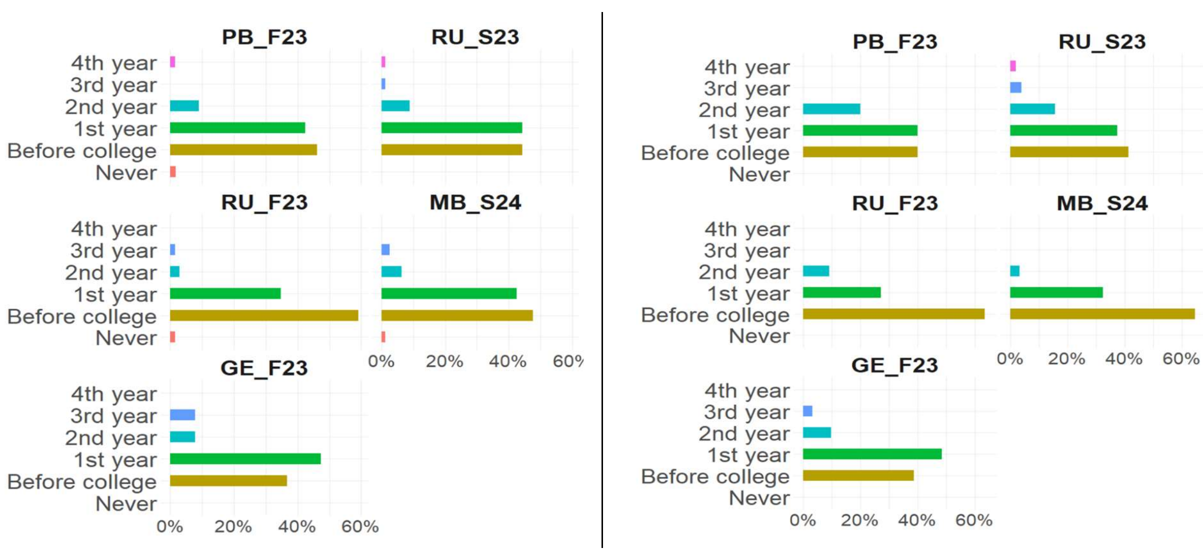
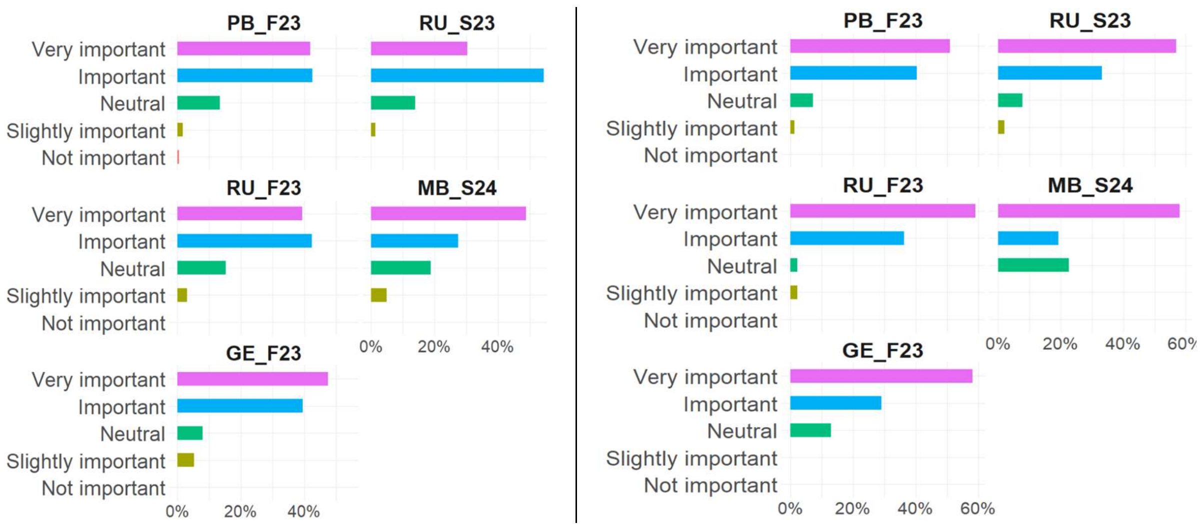


Figure 5. Computational tools students have used before taking one of the listed courses. In the depicted semesters students in 3rd year courses have already taken one or more courses where R or Python are required (Recent semesters, F: Fall, S: Spring).

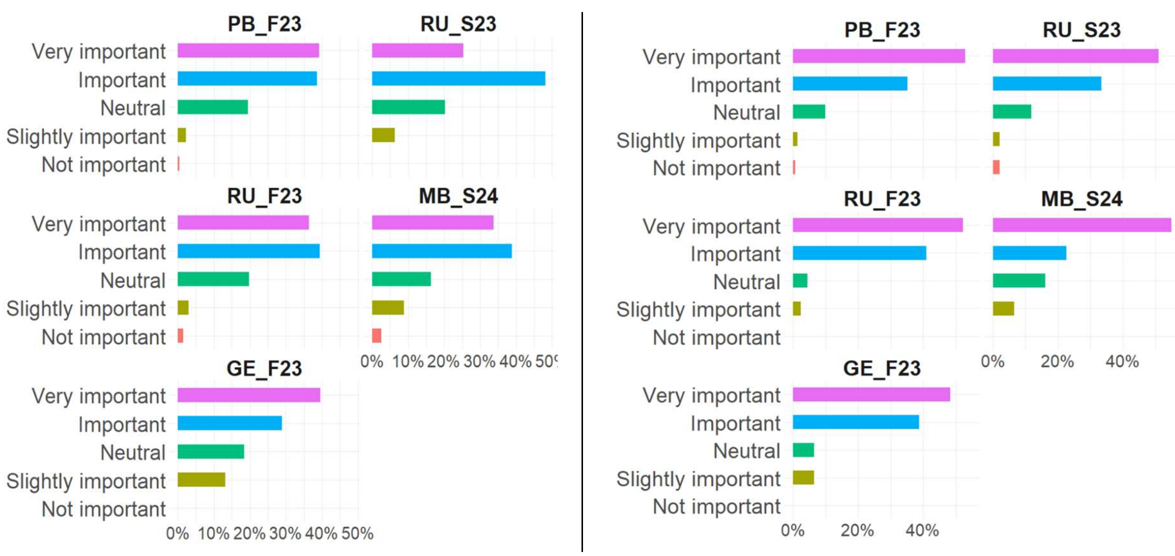
a) ‘What in the best time to start learning coding?’



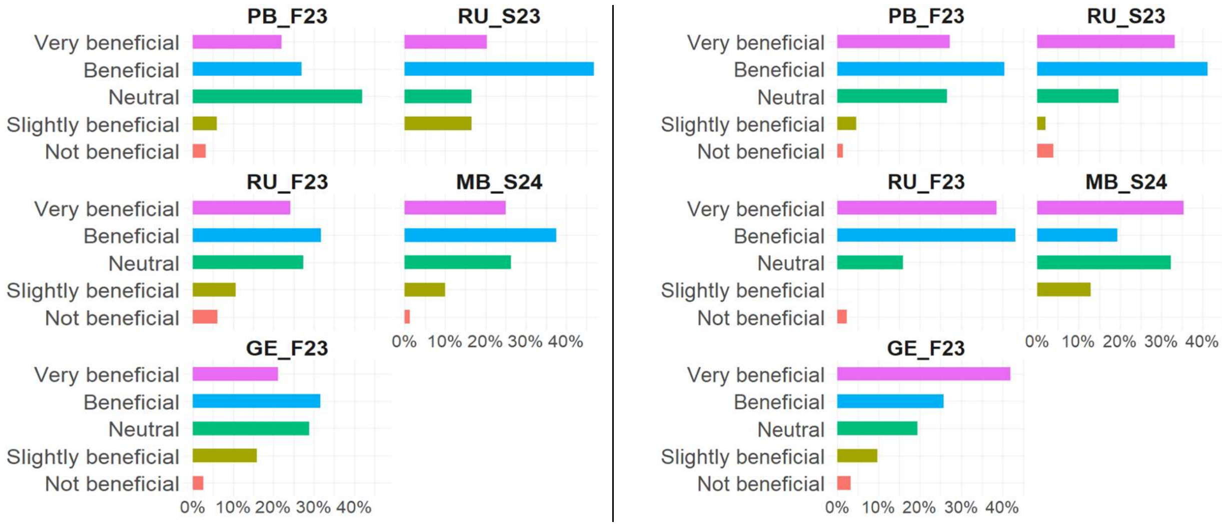
b) ‘Computational tools are important for CEE’.



c) ‘Computational tools are important to help you understand better CEE course materials through simulation and visualization’



d) ‘Computational experiences you had in previous CEE courses were beneficial to you’.



e) ‘Current amount of computation/coding in CEE courses is…’

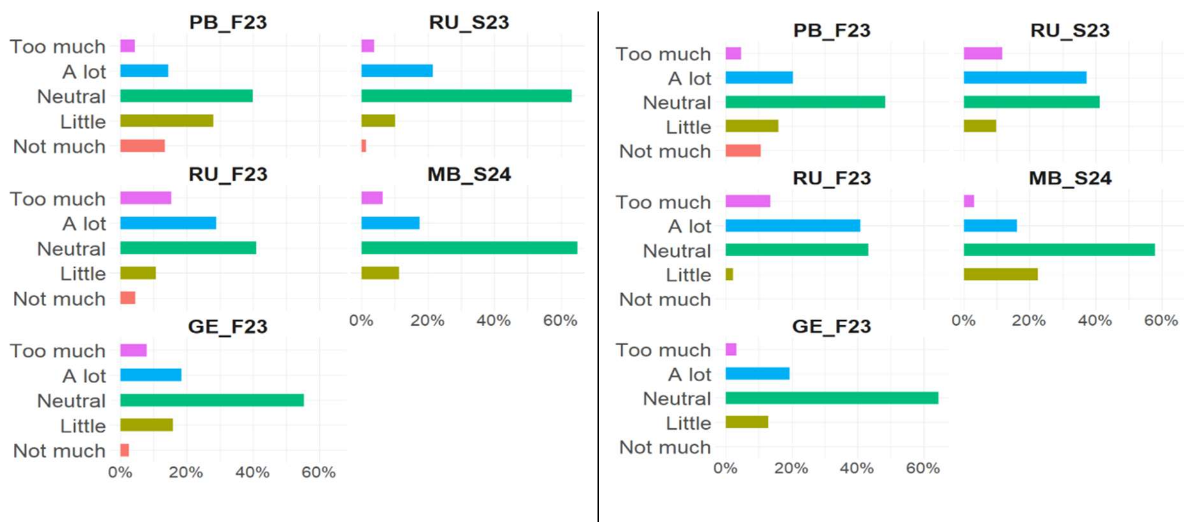


Figure 6. Student responses to questions about coding in recent semesters (F: Fall, S: Spring). Left: early semester responses. Right: end-of-semester responses.

Despite an indicated increased acceptance of coding integration in their CEE studies, the word clouds in Figure 7 display that coding is a dominant word in answers to the question: ‘What challenges do you anticipate in this course?’, for courses where coding is required for most of the assignments. Even though wordclouds provide rough qualitative information, it is still interesting to see that ‘coding’ is the dominant word (expressed as challenge) in courses where coding is required.

Figure 7. Word clouds summarizing responses to the question 'What challenges do you anticipate in this course?" Question is asked early in the semester. (Recent semesters for each course between Spring 2023 to Spring 2024. Wordclouds produced by the *wordcloud* package in R, skipping 'stop-words' such as 'and', 'the', 'to','is', 'are').

In the end of the semester surveys students expressed confidence in using a coding language for specific tasks (Figure 8). Although there is variation across semesters, the overall responses weigh toward higher confidence at the end of a semester, as indicated by the extent of the color-coded sections of the graphs.

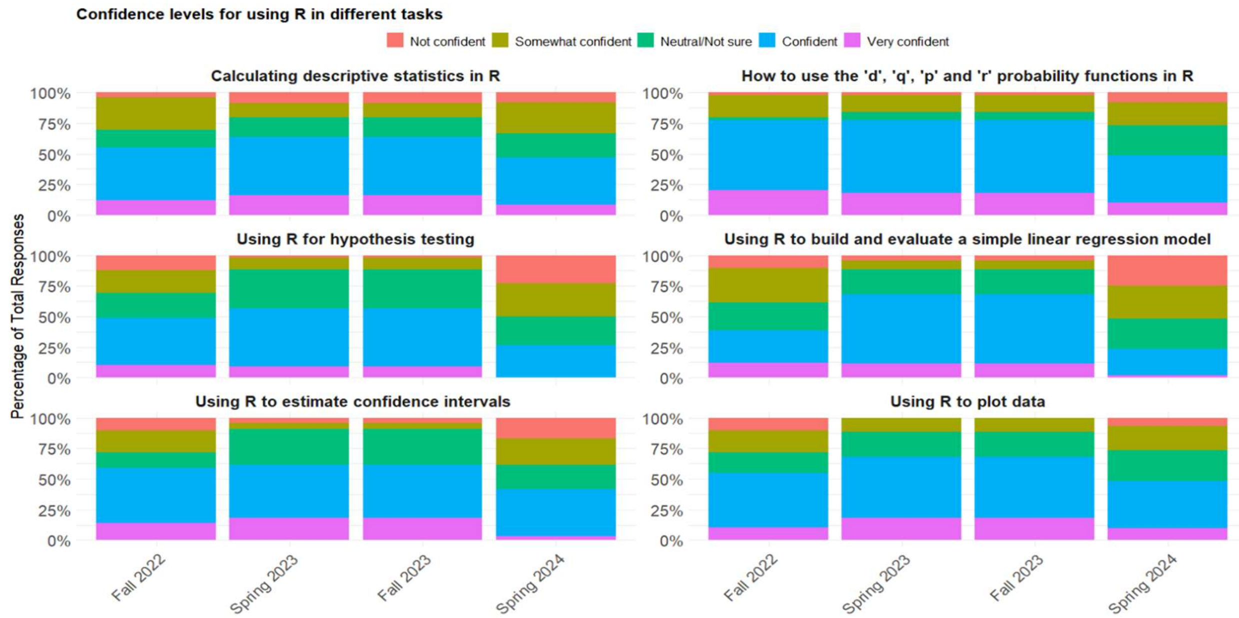
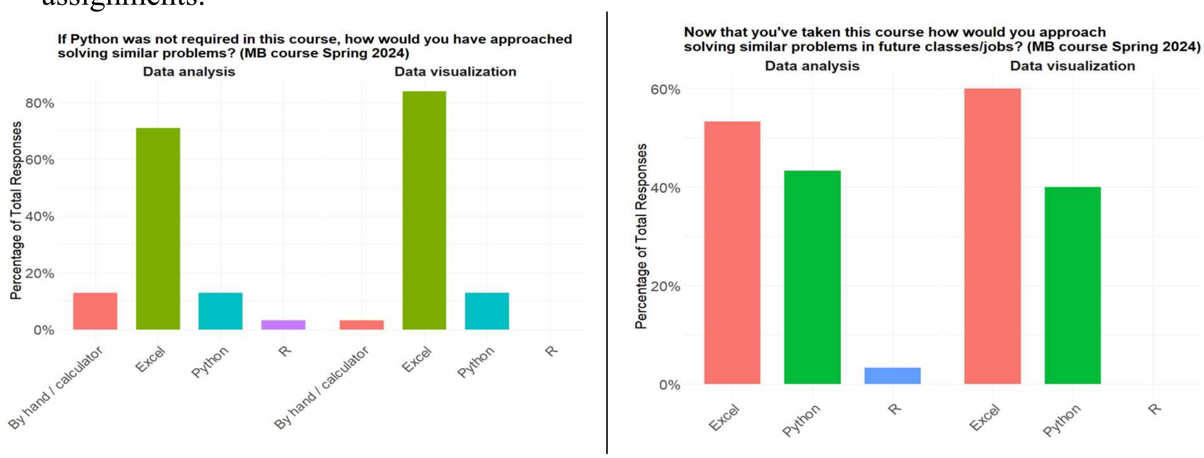


Figure 8. Student responses to the question 'How confident do you feel …'. Responses correspond to the 2nd year RU course where R is required.

Further, in Figure 9, we display % responses to questions asked about the level of comfort students have for using different computational tools at the end of a semester. We observe differentiation between different courses, part of which relates to prerequisite requirements of different courses. In the 2nd year RU course, the first course students take where R is introduced and required (Fig. 9a) Python would have been a choice for most students, which is consistent with CS 101 being a prerequisite for the RU course. By the end of the semester, 50% of the students would use R for data analysis and visualization. For the 3rd yr MB course where Python is required for some assignments (Fig. 9b), Excel is shown as the preferred tool while Python seems to gain acceptance by the end of the semester. For the 3rd yr GE course where R is required (Fig. 9c), there is initially a high preference for Excel, with Excel, Python and R showing equal preference shares by the end of the course.

a) Responses correspond to the 2nd year RU course where R is required.

b) Responses correspond to the 3rd year MB course where Python is required in some assignments.

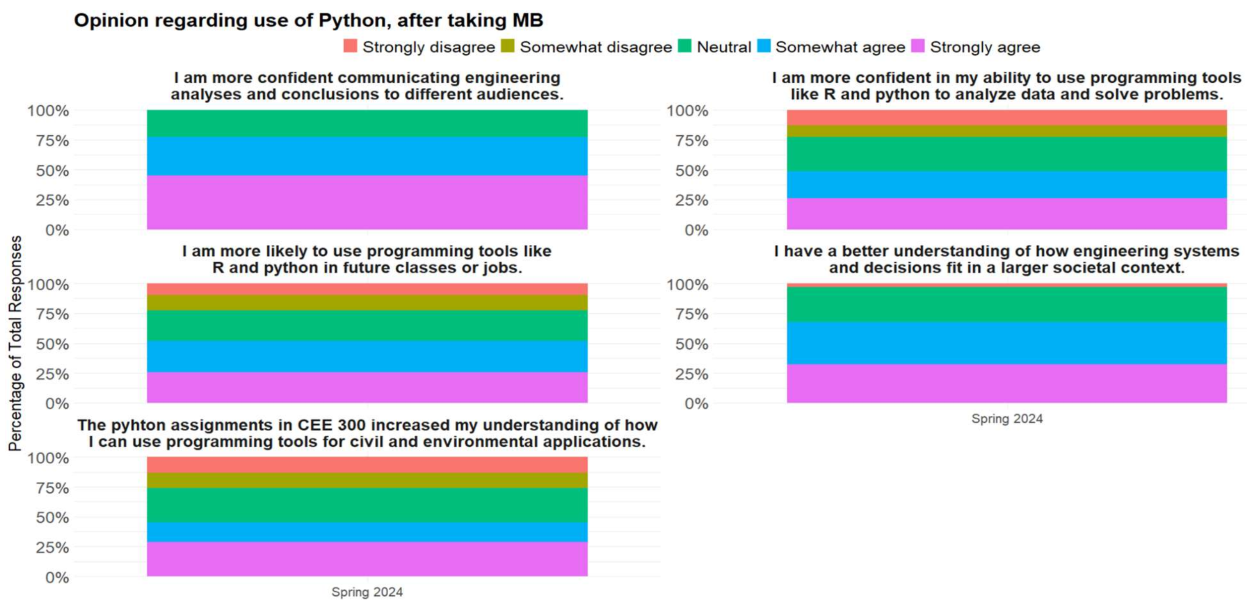


c) Responses correspond to the 3rd year GE course where R is required.

Figure 9. Responses to the questions 1) (left panel) 'If R/Python was not required in this course, how would you have approached solving similar problems?' 2) (right panel) 'Now, that you have taken this course how would you approach solving similar problems in future classes/jobs?'

Figure 10 summarizes student opinions for different aspects of the 3rd yr courses. The sub-questions are modified to better match instructor priorities in their courses. Both these courses emphasize written communication in addition to coding integration. Figures 10 a), b) support increased student confidence in using Python and R in the CEE discipline.

a) Responses correspond to the 3rd year MB course (Spring 2024) where Python is required in some assignments.



b)

b) Responses correspond to the 3rd year GE course (Fall 2023) where R is required.

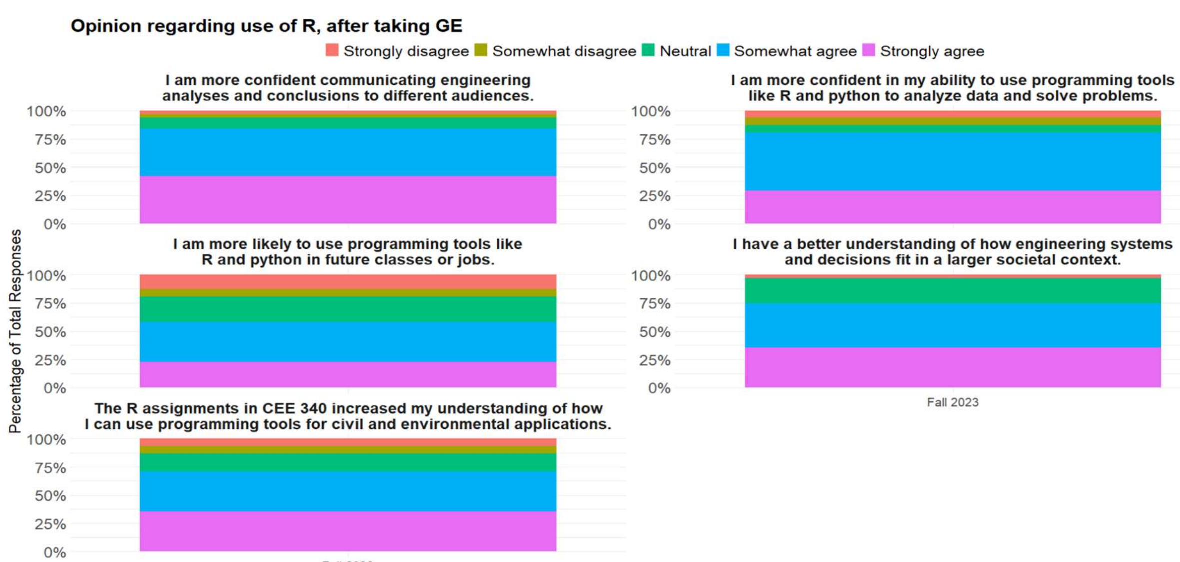


Figure 10. Responses to the question 'What best describes your opinion of … after taking this class?' for several sub-questions.

5. Conclusion

Integration of interpreted high-level object-oriented programming languages in non-CS courses has historically faced CEE student reluctance and negative feelings of fear and concern. In this paper, we presented a snapshot of student responses following the early steps of a curriculum-wide effort to integrate Python and R in the problem-solving process within CEE courses. In agreement with [2], calling for encouraging the development of technological adaptability and agility, our approach fosters computational literacy in multiple platforms as a necessity for which students need to adopt flexible mindset in the current technological environment. Our evidence from the student surveys suggests a change in the student culture as the use of R and Python is becoming the norm in CEE courses. At this stage, responses present a mixed picture shifting between acceptance and resistance (such as the high preference for Excel or a few students expressing preference for calculators) but the trend we see is in the direction of acceptance of coding in the discipline. There are increased numbers of students catching up, as survey responses hint and as we see from the graded assessments (not in the scope of this presentation). In the most recent two semesters the department has offered weekly tutoring sessions on R and Python in addition to the teaching materials used in each course. Response analysis of future surveys will help us document the changes in student attitudes, perceptions and learning outcomes over time. It will also allow us a better quantification and understanding of student attitudes and outcomes as they move from lower to upper-level courses.

The integration of coding into undergraduate civil engineering education requires continuing effort. The rapid development of AI is changing how we use and interact with more traditional familiar computational tools. The impact of AI on how we teach coding is an open issue to be included in our future research and in-class efforts. In such a department-wide effort, faculty engagement is of paramount importance. To this end, we have described our approach for promoting sustainability of the efforts through the organization of the faculty into course specific CoPs as subgroups of the faculty-wide CoP. As the effort continues, fostering of the CoP culture of cooperation is of paramount importance.

Acknowledgments

This work is supported by a 3-year (2022-2025) grant from the Strategic Instructional Initiatives Program (SIIP) of The Grainger College of Engineering and matching funds by the Department of Civil and Environmental Engineering, at the University of Illinois Urbana-Champaign. We thank the Grainger College of Engineering Academy for Excellence in Engineering Education (AE3) Education Innovation Fellows who mentored us in different phases of this project: Professors A. Schleife, A.A.M. Alawini, and C. Radhakrishnan.

References

[1] NASEM, National Academies of Sciences, Engineering, and Medicine, “Imagining the Future of Undergraduate STEM Education: Proceedings of a Virtual Symposium”. Washington, DC, USA: The National Academies Press. 2022. https://doi.org/10.17226/26314.

[2] K. H. McCord et al., "Computing in AEC Education: Hindsight, Insight, and Foresight," Journal of Computing in Civil Engineering, vol. 38, no. 3, 2024, doi: 10.1061/jccee5.Cpeng-5646.

[3] G. Van Rossum and F. L. Drake, Python 3 Reference Manual. Scotts Valley, CA, USA: CreateSpace, 2009.

[4] R Core Team, R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing, 2021. [Online]. Available: https://www.R-project.org/.

[5] OpenAI, "ChatGPT conversation on citing ChatGPT in IEEE format," ChatGPT, Sep. 1, 2024. [Online]. Available: https://chat.openai.com/

[6] A. Karabulut-Ilgu, N. Jaramillo Cherrez, and C. T. Jahren, "A systematic review of research on the flipped learning method in engineering education," British Journal of Educational Technology, vol. 49, no. 3, p. 13, 2018.

[7] CAST (2021). The UDL Guidelines. https://udlguidelines.cast.org/. Last accessed 5/27/2024.

[8] G. B. Wright, "Student-Centered Learning in Higher Education," International Journal of Teaching and Learning in Higher Education, vol. 23, no. 3, pp. 92–97, 2011. [Online]. Available: http://www.isetl.org/ijtlhe/.

[9] J. Palazón-Herrera and A. Soria-Vílchez, "Students' perception and academic performance in a flipped classroom model within Early Childhood Education Degree," Heliyon, vol. 7, no. 4, p. e06702, 2021.

[10] L. Deslauriers, L. S. McCarty, K. Miller, K. Callaghan, and G. Kestin, "Measuring actual learning versus feeling of learning in response to being actively engaged in the classroom," Proc. Natl. Acad. Sci. USA, vol. 116, no. 39, pp. 19251–19257, Sep. 24, 2019.

[11] I. Pablo-Lerchundi, C. Núñez-Del-Río, A. Jiménez-Rivero, S. Sastre-Merino, A. Míguez-Souto, and J.L. Martín-Núñez, "Factors affecting students' perception of flipped learning over time in a teacher training program," Heliyon, vol. 9, no. 11, p. e21318, Oct. 29, 2023.

[12] J. Wing, "Computational thinking," Commun. Assoc. Computing Machinery, vol. 49, no. 3, pp. 33–35, 2006.

[13] J. Wing, "Computational Thinking: What and Why?" 2010. [Online]. Available: www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf.

[14] D. Barr, J. Harrison, and L. Conery, "Computational thinking: A digital age skill for everyone," Learning & Leading with Technology, vol. 38, no. 6, pp. 20-23, 2011.

[15] National Academies of Sciences, Engineering, and Medicine, "Cultivating Interest and Competencies in Computing: Authentic Experiences and Design Factors". Washington, DC, USA: The National Academies Press, 2021. [Online]. Available: https://doi.org/10.17226/25912.

[16] C. Varela, C. Rebollar, O. García, E. Bravo, and J. Bilbao, "Skills in computational thinking of engineering students of the first school year," Heliyon, vol. 5, pp. e02820, 2019. [Online]. Available: https://doi.org/10.1016/j.heliyon.2019.e02820

[17] J. A. Lyon and A. J. Magana, "Computational thinking in higher education: A review of the literature," Computer Applications in Engineering Education, vol. 28, no. 5, pp. 1174-1189, 2020.

[18] A. J. Magana, H. W. Fennell, C. Vieira, and M. L. Falk, "Characterizing the interplay of cognitive and metacognitive knowledge in computational modeling and simulation practices," Journal of Engineering Education, vol. 108, no. 2, pp. 276-303, 2019.

[19] S. Koloutsou-Vakakis, E. Kontou, C. W. Tessum, L. Zhao, and H. Meidani, "Educational technology platforms and shift in pedagogical approach to support computing integration into two sophomore Civil and Environmental Engineering courses," 2021 ASEE Virtual Annual Conference Content Access, Virtual Conference, 2021. [Online]. Available: https://peer.asee.org/37005.

[20] S. Koloutsou-Vakakis, E. Kontou, C. W. Tessum, L. Zhao, and H. Meidani. “Cloud technologies for scalable engagement and learning in flipped classrooms”. Civil Engineering Division -The New Normal: Enduring Technology Improvements in the Classroom (W7211), 2022 ASEE Annual Conference, Minneapolis, MN. https://peer.asee.org/40811.

[21] F. Ä. Nielsen, “A new evaluation of a word list for sentiment analysis in microblogs,” Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages, 93-98, 2011.

[22] M. Hu and B. Liu, “Mining and summarizing customer reviews,” Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD-2004), 2004.

[23] S. Mohammad and P. Turney, “Crowdsourcing a Word-Emotion Association Lexicon,” Computational Intelligence, 29(3): 436-465, 2013.

[24] M. Misuraca, A. Forciniti, G, Scepi, M. Spano, “Sentiment Analysis for Education with R: packages, methods and practical applications”, 2020. https://doi.org/10.48550/arXiv.2005.12840

[25] J. S. Popovics, Y. Niu, S. Koloutsou-Vakakis, K. Pattaje. CASE STUDY: Integration of Python programming in a civil engineering laboratory course. Civil Engineering Division., 2025 ASEE Annual Conference & Exposition, Montreal, Canada [Accepted].

[26] M. L. Matthews, S. Koloutsou-Vakakis, A. S. Stillwell. CASE STUDY: Project-based integration of societal context with engineering communication and computational thinking in an upper-level civil and environmental engineering course. Civil Engineering Division., 2025 ASEE Annual Conference & Exposition, Montreal, Canada [Accepted].