Bringing Computational Science to Collegiate Education

Mitchell Eithun
Department of Computational Mathematics, Science and Engineering
Michigan State University
Lansing, MI 48824
Email: eithunmi@msu.edu

Introduction

Motivated by increased demand for computational skills in science and industry, universities have started offering programs in data science but are faced with the challenge of developing courses in hard-to-define areas such as "computational modeling" and "data analytics." Loosely defined, data science is the science of collecting and analyzing data and is closely related to computational science, which is broadly interested in applications of computing. Sometimes data science is cast as an enhanced version of statistics or computer science; but the predominate academic view envisions computational and data sciences as a unified, yet multidisciplinary field. Practices such as data analysis, mathematical modeling and programming are common in mathematics, statistics and computer science; but computational science adds applications to these practices to foster a particular worldview about modern uses of computing.

Defining "data science" and "computational thinking" provides a framework for talking about computational science in the context of science and mathematics education. New academic programs at the collegiate level represent different approaches to infuse computational science into the curricular landscape. The high cost of creating new programs leads many universities to create programs using resources from existing departments, often computer science and statistics, while other universities have invested in entirety new departments and coursework. A lack of research about computational science education makes it difficult to establish curriculum or choose teaching methods, although progress has been made defining terms, developing broad standards, and adopting practices used in industry. Recent perspectives on computational science education from various fields are surveyed here to summarize the state of the field, compare definitions of terms and provide examples of relevant academic programs for those developing curriculum and coursework.

Evidence-based teaching practices in STEM provide a starting point for new collegiate programs in computational science interested in promoting effective teaching. In particular, the literature strongly supports using active learning, problem-based learning and peer instruction to improve student engagement. However, these methods may be difficult to transfer directly from computer science and mathematics because of cultural and pedagogical differences, necessitating new research on computational science education.

Defining Data Science

"Computational science", "computational modeling", "computational thinking" and "data science" refer to an emergent scientific paradigm focused on using computers to solve science

and business problems. The phrase "data science" has been given the most attention in news articles and job reports, including this description of a data scientist in the Harvard Business Review: "At ease in the digital realm, they are able to bring structure to large quantities of formless data and make analysis possible" [1, Para. 9]. Although this description avoids technical language it may be misleading by not discussing specific practices of data scientists. Cao, in an extensive review of the state of the field, assesses: "while data science as a term has been increasingly used in the titles of publications, it seems that a great many authors have done this to make the work look 'sexier'. The abuse, misuse and over-use of the term 'data science' is ubiquitous, and essentially contribute to the buzz and hype" [2, p. 7]. Hence, a discussion of data science should be brought closer to its actual practitioners such as industry leaders and academics.

Some academics define "data science" with a focus on data itself, while others emphasize methods and disciplinary content. The National Science Foundation writes that data science is the "science of planning for, acquisition, management, analysis of, and inference from data" [3]. De Veaux et al. [4] adopt this definition as a basis for suggestions for undergraduate coursework in data science and append details about each data process. On the other hand, data science for business ("data analytics") focuses on statistical methods for prediction [5]. In the context of supply chain management, Walker and Fawcett assert that "data science is the application of quantitative and qualitative methods to solve relevant problems and predict outcomes. One of the salient revelations of today, with the vast and growing amount of data, is that domain knowledge and analysis cannot be separated" [6, p. 78]. Highlighting the actual activities of data scientists, West cites this definition and adds that "data science methods usually touch on statistics, computer programming (coding) and strategies for understanding and explaining phenomena in various domains of interest" [7, p. 140]. A key distinction between data-focused definitions and Walker and Fawcett's definition is the weight given to methods, rather than content. Blogger Conway writes that "the difficulty in defining these skills is that the split between substance and methodology is ambiguous, and as such it is unclear how to distinguish among hackers, statisticians, subject matter experts, their overlaps and where data science fits" [8, Para. 2]. Conway proposes that data science lie at the intersection of "hacking skills", mathematics and statistics and knowledge of a subject area (see Figure 1), a model that has been used by the Society for Industrial and Applied Mathematics [9]. Therefore, alongside simply using data, data science is related to several existing academic disciplines, including mathematics, statistics, and computer science; but there are disagreements about exactly how these subject areas interact.

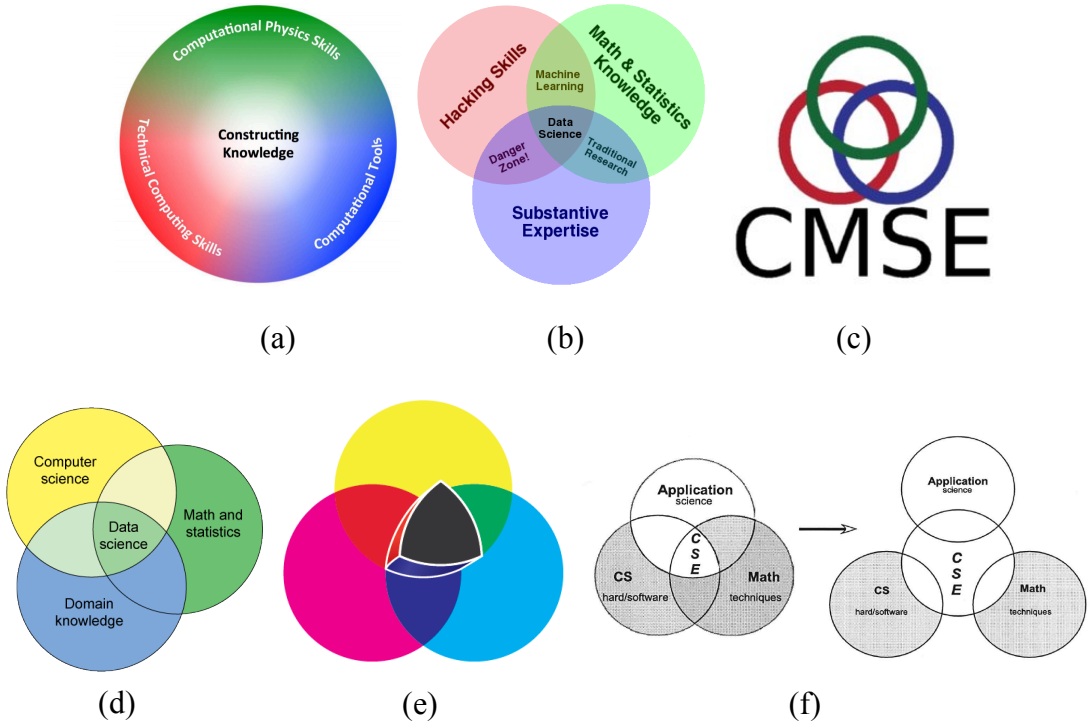


Figure 1. Venn Diagram-type views of computational science. Computational science and data science are often understood as including elements from mathematics, computer science, statistics and domain science.

a) The logo for "Recommendations for Computational Physics in the Undergraduate Physics Curriculum" from the American Association of Physics Teachers [10].
b) The "Data Science Venn Diagram" developed by Conway [8] and referenced by SIAM [9]. The "danger zone" is a lack of mathematics knowledge.
c) The logo for Michigan State University's department of Computational Mathematics, Science and Engineering (CMSE) representing the intersection of mathematics, scientific computing and applications [11].
d) A vision of data science from a talk by Bin Yu, from the Institute of Mathematical Statistics [12].
e) Figure from Denning and Freeman [13] arguing that computing is distinctly different from mathematics, science and engineering.
f) Elements of Computational Science and Engineering Education (CSE) evolving from an intersection of mathematics, computer science and applications to having "common concerns with these disciplines, as well as having content of its own" [14, p. 788].

Determining the substance of data science and its relationship to other fields is a crucial step for designing degree programs and job descriptions. In academia, this discussion is about the ways in which a program is "interdisciplinary" [5], [15]. De Veaux et al. argue that Data Science is a "blend" of statistics, computer science and mathematics; "but it is neither the simple intersection, nor the superset of the three" and that "at the undergraduate level, we conceive of Data Science as an applied field akin to engineering" [4, pp. 3–4]. Similarly, West compares data science to other interdisciplinary fields such as medicine, informatics business and engineering, arguing that students of data science should be trained in different subject areas as well as processes such as "problem definition, data analysis, critical thinking, evidence-based problem solving, communication and ethics" [7, p. 138]. Cao argues that the thinking practices by processes such as data-driven discovery, intelligence-driven exploration, data analytical thinking and hypothesis-free exploration set data science apart from other fields [2, p. 334]. Sometimes data science is defined more narrowly as an enhanced version of statistics; but this perspective lacks high-level aspects of computation and future scientific goals. Donoho writes that

> the now-contemplated field of data science amounts to a superset of the fields of statistics and machine learning, which adds some technology for 'scaling up' to 'big data.' This chosen superset is motivated by commercial rather than intellectual developments. Choosing in this way is likely to miss out on the really important intellectual event of the next 50 years [16, p. 745].

In summary, the emergent field of data science has new ways of thinking about creating, organizing and analysis data that transcend the union or intersection of different fields.

Computational Thinking Practices

The terms "computational thinking" and "computational science" usually focus on how people think about applications of computing. In the last decade there has been a strong push to incorporate computational thinking in K12 education [17]–[22]. This thrust has been motivated by educational standards such as "using mathematics and computational thinking" in the Next Generation Science Standards and using "technological tools to explore and deepen their understanding of concepts" in the Common Core [17]. In a highly influential article, Wing [19] highlights the importance of computational thinking in society by providing a litany of computational thinking practices [18], [23]. Computational thinking, Wing argues, is a fundamental twentieth century skill and "to reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability" [19, p. 33]. According to Barr & Stephenson [24], collaboration between K12 educators and college faculty is necessary to achieve this goal.

Like "data science", "computational thinking" is difficult to define and there is a "lack of agreement on whether [computational thinking] should ultimately be incorporated into education as a general subject, a discipline-specific topic, or a multidisciplinary topic" [18, p. 40]. Barr & Stephenson organized educator workgroups to create a table of "Core Computational Thinking Concepts and Capabilities" and discuss the complexities of defining computational thinking using a "practical approach, grounded in an operational definition" [23, p. 50]. The table contains a list of computational thinking practices and examples of how they could be incorporated into

Computer Science, Mathematics, Science, Social Studies and Language Arts classes [18, p. 52]. To further clarify the nature of "computational thinking," Weintrop et al. analyzed the content of high school programs aimed at fostering computational thinking and consulted with experts in computational fields to devise a taxonomy of practices in computational science [17]. Their *computational thinking taxonomy* provides a way of organizing practices exercised by scientists and educators, which can be used to develop pedagogy and learning goals (see Figure 2). Although the study focuses on demystifying computational thinking in the high school science curriculum, the framework was designed by consulting with academics and researchers and is also useful to discuss computational science in collegiate education.

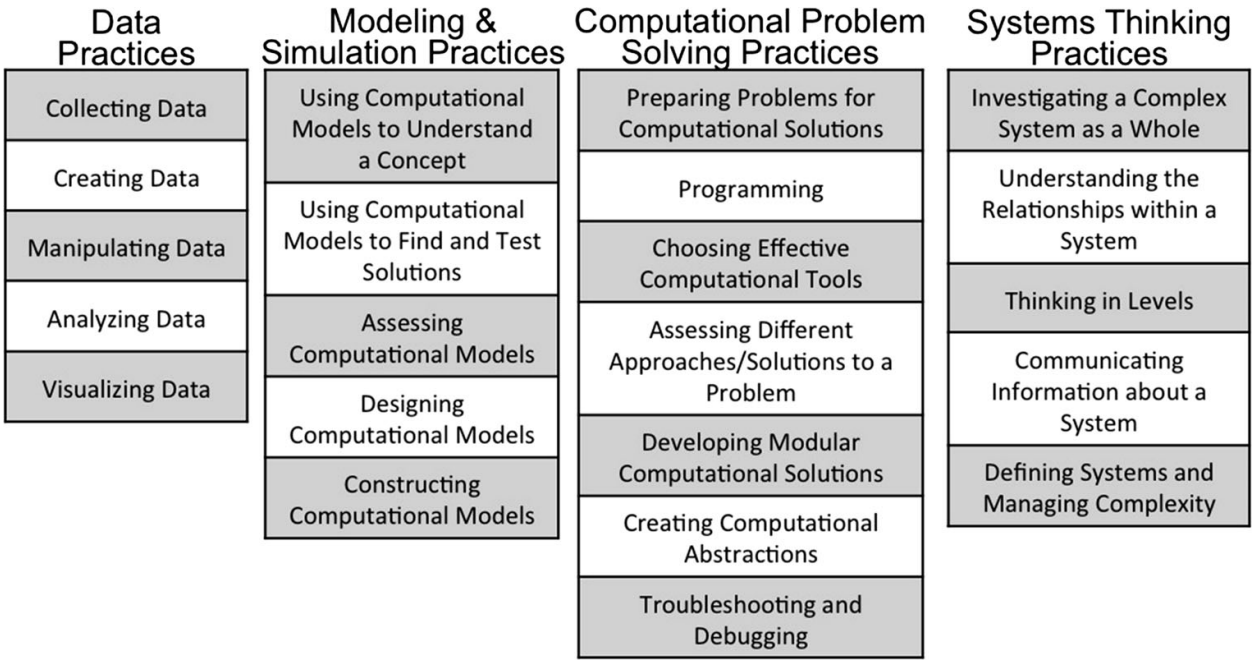


Figure 2. The computational thinking taxonomy. Collated practices in computational thinking based on K12 educational programs and the work of computational researchers [17].

While "computational thinking" refers to computing practices, "computational science" focuses on using and developing computational tools, which may or may not include data analysis. According to Yasar and Landau, computational science in the university denotes both a "multidisciplinary combination of computational techniques, tools, and knowledge needed to solve modern scientific and engineering problems" and "science or engineering that uses computer simulations as its basis, and sometimes it denotes the research and development of computational skills and tools needed for applications" [14, pp. 787–788]. Similarly, Grover and Pea propose nine practices that should form the basis for computational thinking in K12 education, but aside from "systematic processing of information", none of these practices make any explicit references to data [18, pp. 39–40]. Hence, this view of computational thinking is nearly disjoint from data science practices. Others see a more unified view of computational science and data science. For example, Michigan State University "treats computational and data science as the junction of algorithm development and analysis, high-performance computing, and applications to a wide range of important problems in science, engineering, and other fields" [11, p. 2]. This view is implicitly held by Weintrop et al. who include data science

practices in their computational thinking taxonomy, asserting that computational science subsumes data science [17]. Adopting this integrated view leads to implications for coursework in computational science.

Computational Science Curriculum

New programs in computational science are growing rapidly, with about 150-200 US universities offering courses in data science, big data and data analytics [2], [4]. These new programs are usually motivated by the fact that mathematics and statistics are essential for understanding computational science [5], but existing academic departments do not adequately address computational thinking practices or do not sufficiently prepare students for industrial careers. For example, computer science courses at Michigan State University are often "heavily oversubscribed and typically do not encompass the range of skills that employers desire, such as fluency in computational and data science methods" [11, p. 2]. Additionally, while computational science includes many aspects of mathematical thinking [20], the mathematics pipeline is too long for students to be involved in relevant research projects. An advisor at Rensselaer Polytechnic University writes that "a big problem with our current math sequence is that it just takes too long to get where you want to go . . . as a community we should be thinking about if we should be doing things in a different order" [9, Para. 10]. Current pathways to learning computational science are often slow, and so a challenge in collegiate education is deciding how to establish programs dedicated to teaching computational methods.

The similarities between practices in computational science and existing disciplines can be used to build new programs in computational and data sciences. One option is to combine courses offerings from different departments to approximate the breadth of practices in computational science. In 2003 the College of Charleston created an interdisciplinary degree program for undergraduate students to major in Data Science [25]. A few of the courses required for this degree were created for the program, but most were drawn from the Departments of Computer Science and Mathematics and focus on data analysis and statistics. The program successfully brought Data Science to a liberal arts environment, boosted enrollment in computer science and attracted high-achieving students [25]. While this degree is innovative, jointly-administered programs may not be able address all of the computational thinking practices in each course, since other departments have long-standing curricular expectations. For instance, the data science task force at the University of Delaware has acknowledged that "without an established major or minor in data science, structuring a program with multiple units that support students' diverse passions is challenging . . . perhaps the solution is to take such a program out of a department and run it as an interdisciplinary project" [9]. In response to the emergence of as computational science as a distinct entity, some universities have created entirely new programs in computational science and data science.

In 2016, the Park City Math Institute (PMI) devised a set of curriculum guidelines for undergraduate programs in data science, with an interdisciplinary focus on computational and statistical thinking [4]. The result is a collection of syllabi outlining a data science major (see Figure 3). Recognizing that data science is "inherently interdisciplinary", but not a "simple intersection" (cf. Figure 1), the faculty at PMI advocate that "new courses should be developed to take advantage of the efficiencies and synergies that an integrated approach to Data Science

would provide" [4, p. 4]. The proposal suggests nine new courses in data science, but the authors realize that most universities will simply draw from existing courses (the approach taken by the College of Charleston) and hope that this will eventually lead to "more fully integrated courses" [4]. While easier and more cost-effective, this approach might not provide adequate training for students interested in data science careers. Cao calls this approach "old wine in new bottles" since "the re-combination and shuffling of existing subjects, the logical relationships between subjects are often very weakly drawn and do not form a logical, reasonable concept map and knowledge base for training a data scientist" [2, pp. 334–335]. On the other hand, Weintrop et al. argue that adding computational components to existing classes will address minority representation in STEM fields by "directly [addressing] the issue of students self-selecting into (or out of) computer science classes, which has been a challenge long plaguing the effort to reach underserved youth" [17, p. 129]. They also discuss a "reciprocal relationship" between established disciplines and computational science: while computational methods lead to scientific advancements, it is also true that "science and mathematics provide a meaningful context (and set of problems) within which computational thinking can be applied" [17, p. 129]. Hence, introducing computational thinking practices in STEM classrooms has the potential to strengthen traditional subject areas and boost representation, but may not be sufficient to introduce students to all of the interdisciplinary content of data science.

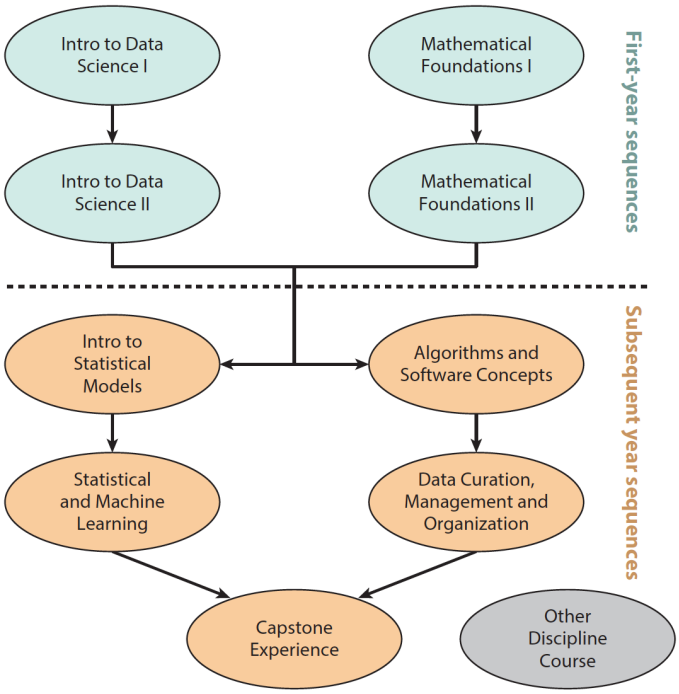


Figure 3. Requirements for an undergraduate degree in Data Science. A workshop at The Park City Math Institute produced guidelines for undergraduate coursework Data Science [4].

The Department of Computational Mathematics, Science and Engineering (CMSE) at Michigan State University (MSU) is an example of an entirely new academic department created to address a need for training in computational and data science practices [11]. Department faculty explain that "a wide range of industries now use computational modeling and data analytics to inform

many aspects of their day-to-day practices, encompassing a range of activities that include product design, optimizing manufacturing, hiring decisions, and choosing advertising targets" [11, pp. 1–2]. The first educational initiative by CMSE was developing CMSE 201: "Introduction to Computational Modeling." CMSE 201 aims to make a broad palette of computational practices available to as wide an audience as possible using domain content from physics, chemistry and biology to teach skills common to both data science and modeling [11, p. 4]. The only prerequisite is one semester of calculus. The programming skills taught in the class are motivated entirely by applications within the course and no prior programming experience is expected. "Students certainly learn many of the same computer science topics (i.e., variables and arrays, loops, simple data structures, functions, etc.), but they do so in service of building and running simulations and analyzing data" [11, p. 4]. Class work is done is completed in groups and students use vertical whiteboards to share and discuss ideas, strategies Liljedahl [26] identifies with a "thinking classroom" in mathematics education.

The course content and teaching style of CMSE 201 are rooted in scientific computational skills rather than traditional syllabi in computer science or mathematics. The designers of CMSE 201 list six learning goals:

(1) Gain insight into physical, biological, and social system through the use of computational algorithms and tools.
(2) Write programs to solve common problems in a variety of disciplines.
(3) Identify salient features of a system that can be codified into a model.
(4) Manipulate, analyze, and visualize datasets and use these data to evaluate models.
(5) Understand basic numerical methods (e.g. numerical integration, differential equations, Monte Carlo) and be able to use them to solve problems.
(6) Synthesize results from a scientific computing problem and present it both verbally and in writing [11].

These goals are closely aligned with the computational thinking taxonomy designed by Weintrop et al. [17]. "Data Practices" are found in goal 4, "Computational Problem Solving Practices" are found in goal 2, "Modeling and Simulation Practices" are found in goals 1, 3 and 5 and "Systems Thinking Practices" are found in goals 1 and 3. Hence, CMSE 201 captures the breadth of computational practices envisioned by Weintrop et al. along with scientific writing (goal 6). Furthermore, each day of class has three components: "modeling/data analysis concept", "context/application", and "programming practices." These different activities expose students to a mix of mathematics, domain science and technical skills [11], which are important aspects of computational and data sciences [4], [12], [14]. Courses similar to CMSE 201 have been developed by Asamoah, Doran and Schiller [5] at Wright State University with a focus on business analytics and Baumer [27] at Smith College with a focus data science in statistics. CMSE 201 is particularly notable for fusing computational and data sciences skills with mathematics and science in a way that is accessible to undergraduates and is aligned with computational thinking practices.

Teaching Computational Science

One primary difficulty in designing and teaching a computational science course is the lack of research about computational science education. While designing CMSE 201, O'Shea et al.

identified the following gap in the literature: "while a set of measurable concepts in programming has been defined in computer science education research community, a similar inventory of concepts and a related assessment tool does not exist with regards to computational modeling and data analysis" [11, p. 21]. West (2018) echoes this sentiment, writing that "data science degree development would benefit from the objective translation of these activities into competencies for the development of a 'baseline' data science curricula" [7, p. 141]. While the curriculum designed by De Veaux et al. [4] arguably fills this role, West [7] argues for an "objective approach" to validate curriculum, such as natural language processing, to remove human intuition from the process. Aside from limited competencies for computational and data sciences, there is little empirical research about how to encourage computational thinking [18, p. 42] and so it is difficult to design courses around evidence-based teaching practices.

On the other hand, the overlap between closely-related areas such as mathematics, computer science and engineering suggests several practices for teaching computational science. One approach to teaching college science courses that has garnered a lot of attention from faculty is "active learning." There are many realizations of active learning, which Prince (2014) defines as "any instructional method that engages students in the learning process . . . [and] requires students to do meaningful learning activities and think about what they are doing. Active learning is often contrasted to the traditional lecture where students passively receive information from the instructor" [28, p. 1]. By analyzing 225 studies about the effects of active learning in collegiate STEM classes, Freeman et al. [30] find that using active learning increases student performance on examinations and concept inventories by 0.47 standard deviations on average when compared to a lecture format. The results of this meta-study suggest that active learning may be well-suited to computational science, which has elements of science, technology, engineering and mathematics—the different areas in STEM. However, the success of active learning depends on the implementation. According to Prince, "claiming that faculty who adopt a specific method will see similar results in their own classrooms is simply not possible. Even if faculty master the new instructional method, they cannot control all other variables that affect learning" [29, p. 3]. The similarity of the course goals and content in STEM disciplines suggest that successful practices may transfer to computational science, but there is little research targeting this question.

A universally accepted definition of active learning does not exist, there are several generally accepted practices in which students actively participating in the learning process. For example, Caceffo, Gama and Azeved [31] define active learning as "a student-centered paradigm in which students should not only passively listen, but also read, write, discuss, be engaged in solving problems and also actively think and reflect throughout this process" [31, p. 922]. A similar concept to active learning is "collaborative learning" which refers to "any instructional method in which students work together in small groups toward a common goal" [29, p. 1]. One type of collaborative learning is "cooperative learning", which is defined as a "structured form of group work where students pursue common goals while being assessed individually" [29, p. 1]. As an example, the MSU course described earlier (CMSE 201) uses cooperative learning since students collaborate in class to solve computational problems and then each student turns in their own digital notebook [11]. A related category of active learning methods is *problem-based learning* (PBL) in which "problems are introduced at the beginning of the instruction cycle and used to provide the context and motivation for the learning that follows" [29, p. 1]. After examining

several studies on active learning, Prince [29] finds evidence to support collaborative, cooperative and problem-based learning in STEM. The aspects of learning that are boosted the most by these methods are "instruction in problem solving" and "cooperative." The former of these fits with the computational taxonomy's category of "computational problem solving" [18] and so active learning may be well suited to computational science.

Another teaching method meant to encourage active student participation is *peer instruction* (PI). Vickrey, Rosploch, Rahmanian, Pilarz and Stains define PI as "an evidence-based instructional practice that consists of asking students conceptual questions during class time and collecting their answers via clickers or response card" [32, p. 1]. The purpose of PI is to "engage students through activities (conceptual questions) during the class, allowing the instructor to identify any misunderstandings or learning issues among the students regarding the core concepts taught" [31, p. 923]. Peer instruction involves a cycle of specific steps that allow students to reflect and collaborate on questions posed by the teacher. According to Vickrey et al. [32], students in STEM classes taught using peer instruction have better conceptual understandings, higher confidence in the material and higher self-efficacy, compared to students in a lecture-based class. Like active learning, the success of peer instruction in STEM suggests that it would be well-suited for computational science classrooms; but it depends on the actual implementation.

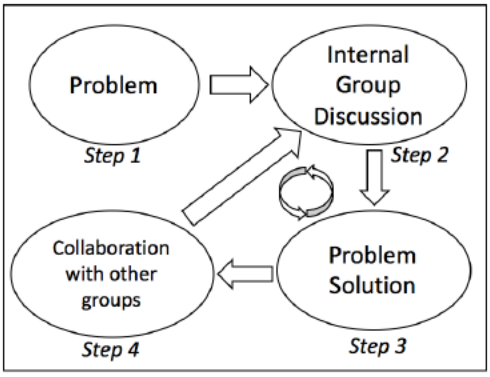
Active learning has been studied in computer science in various settings [31], [33]–[36]. Nuutila et al. [36] apply PBL to an introductory programming class in a computer science department, with a more specific formulation than the definition given by Prince [29]. Based on a traditional PBL scheme proposed by Schmidt [37] for use in medicine (see Figure 4), students cycle through a process of reading a case, brainstorming, establishing goals and discussing solutions. Students examine three types of cases: *knowledge-orientated* cases present a real-world problem meant to create cognitive dissonance; *design* cases present a real-world design problem that is too difficult for one student to complete; and *analytic* cases are meant to identify and confront students' misunderstandings. As compared to a lecture-based course, Nuutila et al. [36] find that their PBL implementation lowered dropout rates, improved motivation and provided emotional support for students.

In a later study, Caceffo et al. (2018) conducted a small experiment ($n < 20$) to compare the effectiveness of PBL and PI with a traditional lecture format in an introductory computer science course. A distinction is made between "active" strategies such as PBL and PI and lecture, "is an instructor-centered approach in which students passively listen to and absorb any data presented to them." [31, p. 922]. The approach to PBL was simpler than the one used by Nuutila, Törmä and Malmi [36] by using short in-class activity that allowed and intra-group collaboration (see Figure 4). The four-step method proceed as follows: divide class into groups and discuss problems (15 minutes), group starts to discuss (15 minutes), develop initial solution (20 minutes), groups exchange solutions and then repeat (30 minutes). For the PI course, students were instructed students to watch video and completed exercises outside of class. In class the website Poll Everywhere was used to evaluate students' understanding. Comparing the three formats (lecture, PI and PBL), students' self-reported learning gains and motivation levels were highest for PBL and preparation time for students was lowest [31]. While this study is small it provides promising anecdotal evidence to support using peer-instruction in an introductory programming class. Learning to program is a critical skill for computational science, which

suggests that peer instruction may be an appropriate technique to adopt in computational science courses.

Opening session – half an hour, in the group
Step 1: Examination of the case. The group gets familiar with the case material. Step 2: Identification of the problem. An initial title for the case is specified. Step 3: Brainstorming. The students present their associations and ideas about the problem to find out what is already known and how does the case relate to the previous knowledge. The ideas are said aloud and written on self-stick notes, which are organized on a white board. Step 4: Sketching of an explanatory model. An initial version of the explanation for the problem is constructed and most important concepts and their relations are identified. Step 5: Establishing the learning goals. Those parts of the explanatory model that are mysterious, fuzzy, or simply unknown are identified and the central ones are chosen as learning goals for the group.
Study period -- one week, each student working independently
Step 6: Independent studying. Each student independently studies to accomplish *all* learning goals. This phase includes information gathering and usually a substantial amount of reading (e.g., 50--150 pages).
Closing session -- one to two hours, in the group
Step 7: Discussion about learned material. Equipped with the newly acquired knowledge, the group reconvenes to discuss the case. The discussion includes *explanation* of central concepts and mechanisms, *analysis* of the material, and *evaluation* of its validity and importance.

(a)



(b)

Figure 4. Schemes for problem-based learning.

a) Nuutila et al. [36] used a seven-step PBL scheme to engage students in solving problems in an introductory programming class.
b) Caceffo et al. [31] used an abbreviated four-step version of PBL and allowed for more collaboration between groups.

Different views developing technological skills may make it difficult to transfer effective practices to computational science. One important consideration is that computer science education is focused heavily on developing good programming practice, whereas computational thinking focuses on solving problems [18]. Nuutila, et. al define "computer programming" as "a skill of designing, implementing and analyzing computer programs of various scales" [36, p. 136]. Even though the PBL course described by Nuutila et al. [36] uses problem-based learning, problems are a means to learn programming skills rather than explicitly honing problem-solving skills. In an effort to introduce students to computational thinking, a professor might suggest that a student take an introductory programming course; but the course may not promote "computational thinking" directly or completely. For example, faculty at MSU created CMSE 201 as an opportunity to learn programing through the lens of computational applications [11]. Like computer science, the mathematics education community views technology in ways that might conflict with computational science goals. To review studies focusing on the effectiveness of incorporating technology in mathematics classrooms, Drijvers asks "does digital technology really work, why does it work, which factors are decisive in making it work or preventing it from working?" [38, p. 2]. This discussion is focused on "does technology work" rather than "how can we teach mathematics relevant to technology". In other words, technology is seen as a tool to improve current ways of learning mathematics, rather the object of the lesson itself, which is the

case in a computational science classroom. Furthermore, implicit misconceptions about different academic fields directly influence the educational goals of different departments. Nuutila et al. write that "in other academic areas [aside from computer science], e.g. mathematics and physics, the student tries to solve small problems defined by the teacher. There is typically one correct solution and the student does not use the solution anywhere" [36, p. 137]. In light of the computational thinking taxonomy, this notion is misguided. An important aspect of modern mathematics and physics – and now computational science – is to construct models and manipulate data [4, Sec. 3.2], [10]. Furthermore, the category of "systems thinking" in the computational thinking taxonomy supports the application and interconnectedness of different computational systems, meaning that results are used in different contexts [18]. Misperceptions like these may make it difficult to borrow pedagogical methods from other disciplines to teach computational science in a way that truly encompasses all of its practices, especially since practices are not fully standardized.

Conclusion

Computational science is a new frontier with the potential to reshape science, industry and society. The core elements of computational science are data, modeling and computation, but the multidisciplinary field is difficult to describe since it closely related to other fields. Furthermore, relevant applications are usually in the context of a scientific or business domain. Computational science arguably subsumes the field of data science since it provides new routes to scientific discovery by collecting, organizing and analyzing data. By identifying important computational thinking practices, K12 teachers can better prepare students for computationally intensive careers and universities can start to develop new courses aimed at developing prerequisite mathematical, statistical and computational practices. Many universities have created interdisciplinary programs out of mathematics and computer science departments; but this approach may limit the ability to fully capture computational science practice, which also includes unique ways of thinking about technology and computation. Entirely new departments in computational science have also been created in hopes of synthesizing technical skills and applications. Relying on existing teaching methods in closely related STEM fields is helpful since there is little available research about effective practice in computational science education, but expectations around curriculum and skill development in computer science may make it difficult to transfer effective practices. Hence, methods of organizing important practices in computational science, such as the computational thinking taxonomy [18] and the curriculum guidelines for data science [4], are valuable tools for developing educational programs with concrete goals. Building on these frameworks and adopting effective teaching methods such as active learning may be a good place to start addressing the lack of empirical research in computational science education.

Bibliography

[1] T. H. Davenport and D. J. Patil, "Data Scientist: The Sexiest Job of the 21st Century," *Harvard Business Review*, 2017.

[2] L. Cao, "Data Science Education," in *Data Science Thinking: The Next Scientific, Technological and Economic Revolution*, L. Cao, Ed. Cham: Springer International Publishing, 2018, pp. 329–348.

[3] I. Johnstone and F. Roberts, "Data Science at NSF." National Science Foundation, 2014.

[4] R. De Veaux *et al.*, "Curriculum Guidelines for Undergraduate Programs in Data Science," *Annu. Rev. Stat. Its Appl.*, vol. 4, no. 1, pp. 15–30, Mar. 2017.
[5] D. Asamoah, D. Doran, and S. Schiller, "Teaching the Foundations of Data Science: An Interdisciplinary Approach," *ArXiv151204456 Cs*, 2015.
[6] M. A. Waller and S. E. Fawcett, "Data Science, Predictive Analytics, and Big Data: A Revolution That Will Transform Supply Chain Design and Management," *J. Bus. Logist.*, vol. 34, no. 2, pp. 77–84, Jun. 2013.
[7] J. West, "Teaching data science: an objective approach to curriculum validation," *Comput. Sci. Educ.*, vol. 28, no. 2, pp. 136–157, Apr. 2018.
[8] D. Conway, "The Data Science Venn Diagram," *Drew Conway Data Consulting*, 2010. .
[9] L. Sorg, "Integrating Data Science in Applied Mathematics Curricula," *SIAM News*, 2017. [Online]. Available: https://sinews.siam.org/Details-Page/integrating-data-science-in-applied-mathematics-curricula. [Accessed: 12-Oct-2018].
[10] E. Behringer and L. Engelhardt, "Guest Editorial: AAPT Recommendations for computational physics in the undergraduate physics curriculum, and the Partnership for Integrating Computation into Undergraduate Physics," *Am. J. Phys.*, vol. 85, no. 5, pp. 325–326, Apr. 2017.
[11] B. O'Shea, D. Silvia, and B. Danielak, "A Learner-Centered Approach to Teaching Computational Modeling, Data Analysis, and Programming," 2019.
[12] B. Yu, "IMS Presidential Address: Let us own Data Science," *IMS Bulletin*, 2014. [Online]. Available: http://bulletin.imstat.org/2014/10/ims-presidential-address-let-us-own-data-science/. [Accessed: 27-Jan-2019].
[13] P. J. Denning and P. A. Freeman, "The profession of IT: Computing's paradigm," *Commun. ACM*, vol. 52, no. 12, p. 28, Dec. 2009.
[14] O. Yasar and R. H. Landau, "Elements of Computational Science and Engineering Education," *SIAM Rev.*, vol. 45, no. 4, pp. 787–805, Jan. 2003.
[15] L. R. Lattuca, L. J. Voight, and K. Q. Fath, "Does interdisciplinarity promote learning? Theoretical support and researchable questions. Rev. Higher Educ," 2004.
[16] D. Donoho, "50 Years of Data Science," *J. Comput. Graph. Stat.*, vol. 26, no. 4, pp. 745–766, Oct. 2017.
[17] J. M. Wing, "Computational thinking and thinking about computing," *Philos. Transact. A Math. Phys. Eng. Sci.*, vol. 366, no. 1881, pp. 3717–3725, Oct. 2008.
[18] D. Weintrop *et al.*, "Defining Computational Thinking for Mathematics and Science Classrooms," *J. Sci. Educ. Technol.*, vol. 25, no. 1, pp. 127–147, Feb. 2016.
[19] S. Grover and R. Pea, "Computational Thinking in K–12: A Review of the State of the Field," *Educ. Res.*, vol. 42, no. 1, pp. 38–43, Jan. 2013.
[20] J. M. Wing, "Computational Thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.
[21] J. Lockwood and A. Mooney, "Computational Thinking in Education: Where does it Fit? A systematic literary review," *ArXiv170307659 Phys.*, Mar. 2017.
[22] M. Bower *et al.*, "Improving the Computational Thinking Pedagogical Capabilities of School Teachers," *Aust. J. Teach. Educ.*, vol. 42, no. 3, Jan. 2017.
[23] V. Barr and C. Stephenson, "Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community?," *ACM Inroads*, vol. 2, no. 1, pp. 48–54, Mar. 2011.
[24] V. Barr and C. Stephenson, "Bringing Computational Thinking to K-12: and What is the Role of the Computer Science Education Community?," *ACM Inroads*, vol. 2, no. 1, 2011.
[25] National Institute of Standards and Technology, "NIST big data interoperability framework: Volume 1, definitions." NIST Special Publication, 2017.
[26] P. Anderson, J. Bowring, R. McCauley, G. Pothering, and C. Starr, "An Undergraduate Degree in Data Science: Curriculum and a Decade of Implementation Experience," *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, pp. 145–150, 2014.
[27] P. Liljedahl, "Building Thinking Classrooms: Conditions for Problem-Solving," in *Posing and Solving Mathematical Problems*, P. Felmer, E. Pehkonen, and J. Kilpatrick, Eds. Cham: Springer International Publishing, 2016, pp. 361–386.
[28] B. Baumer, "A Data Science Course for Undergraduates: Thinking With Data," *Am. Stat.*, vol. 69, no. 4, pp. 334–342, Oct. 2015.
[29] M. Prince, "Does active learning work? A review of the research," *J Engr Educ.*, pp. 223–231.
[30] S. Freeman *et al.*, "Active learning increases student performance in science, engineering, and mathematics," *Proc. Natl. Acad. Sci.*, p. 201319030, May 2014.

[31] R. Caceffo, G. Gama, and R. Azevedo, “Exploring Active Learning Approaches to Computer Science Classes,” in *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, New York, NY, USA, 2018, pp. 922–927.
[32] T. Vickrey, K. Rosploch, R. Rahmanian, M. Pilarz, and M. Stains, “Research-based implementation of peer instruction: a literature review,” *CBE Life Sci. Educ.*, vol. 14, no. 1, p. es3, Mar. 2015.
[33] J. Eickholt, P. Seeling, and M. Johnson, “Supporting Active Learning in Computer Science Through Technology and Community,” *J Comput Sci Coll*, vol. 33, no. 1, pp. 39–40, Oct. 2017.
[34] J. Pirker, M. Riffnaller-Schiefer, and C. Gütl, “Motivational Active Learning: Engaging University Students in Computer Science Education,” in *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, New York, NY, USA, 2014, pp. 297–302.
[35] J. Kay *et al.*, “Problem-Based Learning for Foundation Computer Science Courses,” *Comput. Sci. Educ.*, vol. 10, no. 2, pp. 109–128, Aug. 2000.
[36] E. Nuutila, S. Törmä, and L. Malmi, “PBL and Computer Programming — The Seven Steps Method with Adaptations,” *Comput. Sci. Educ.*, vol. 15, no. 2, pp. 123–142, Jun. 2005.
[37] H. G. Schmidt, “Problem-based learning: rationale and description,” *Med. Educ.*, vol. 17, no. 1, pp. 11–16, Jan. 1983.
[38] P. Drijvers, “Digital Technology in Mathematics Education: Why It Works (Or Doesn’t),” in *Selected Regular Lectures from the 12th International Congress on Mathematical Education*, S. J. Cho, Ed. Cham: Springer International Publishing, 2015, pp. 135–151.

Biographical Information

Mitchell Eithun
Website: egr.msu.edu/~eithunmi
Email: eithunmi@msu.edu