2021 ASEE ANNUAL CONFERENCE
Virtual Meeting | July 26–29, 2021 | Pacific Daylight Time
Paper ID #32325

Towards Identifying Core Computational Literacy Concepts for Inclusion in a First-year General Engineering Course

Dr. Darren K. Maczka, University of Tennessee at Knoxville

Darren Maczka is a Lecturer and Research Assistant Professor in the Engineering Fundamentals program at the University of Tennessee, Knoxville. He received his PhD in Engineering Education from Virginia Tech.

Mr. Rehan Shah, University College London

Rehan Shah is a third year doctoral student pursuing a PhD in Applied Mathematics at University College London (UCL). He has an MSc in Applied Mathematics from the University of Oxford (St. Anne's College) and a BEng in Mechanical Engineering with Business Finance from University College London (UCL) with the London School of Economics (LSE).

Dr. Andrew L. Gillen, University College London

Andrew L. Gillen is a Lecturer in the Department of Civil, Environmental, and Geomatic Engineering at University College London. He has a Ph.D. in Engineering Education from Virginia Tech and B.S. in Civil Engineering from Northeastern University.

Towards Identifying Core Computational Literacy Concepts for Inclusion in a First-Year General Engineering Course

Abstract

In this paper, we describe an exploratory study to support efforts in revising first-year courses required for engineering students. It is common to include some form of introductory programming or spreadsheet computation topics in first-year courses. The inclusions of these topics is ostensibly to provide foundational computational skills needed in later courses. However, there are many challenges associated with teaching and learning these skills, the least of which is selecting which skills to include in the finite time allotted for a first-year introductory course that may also be tasked with teaching foundational problem solving and professional skills. This study is the first stage towards identifying a core set of skills for inclusion that would be relevant for most first-year students.

Introduction and Background

It is well established that most first-year programs have an introductory programming course (Reid & Reeping, 2014). Design and successful implementation of these courses is challenging in the context of a general first-year engineering program as students enter with a wide variety of prior knowledge and experience, and will matriculate into disciplinary majors with varying expectations regarding programming and data expertise. While several studies have demonstrated a benefit to teaching these computational literacy concepts grounded in a disciplinary context (Magana, Falk, Viera & Reese, 2016; Forte & Guzdial, 2005), it is difficult to do this in a first-year course intended for all engineering disciplines. What computing concepts are common across all non-computing engineering majors, and which ought to be included in such a course? Towards establishing a taxonomy, we found cross-cutting core computational literacy concepts based on data collected from publicly available course catalog descriptions of disciplinary engineering courses.

Computational Literacy

We use the phrase "computational literacy" to capture the intersection of data literacy and computing skills necessary to work with data on a computer in a problem solving context. Data literacy, in its traditional sense, is defined as a competency measure used to gauge individuals' abilities to access, interpret, critically assess, manage, communicate and ethically use data (Prado & Marzal, 2013). However, with the recognition of the increasing importance of computational literacy as a valuable learning outcome within undergraduate courses, a new framework of

data-informed learning i.e., learning that emphasizes the use of data within a specific disciplinary context, while constructively building on students' past experiences, is being used progressively as a tool to promote lifelong learning in higher education (Maybee & Zilinski, 2015).

An empirically motivated case-study by Magana et al. (2016) discusses a similar notion termed "authentic computational learning" (learning that is meaningful to the learner, contextual to the discipline and relevant to real-world applications) by assessing the effectiveness of a novel computational and programming freshman/sophomore-level course introduced within the Material Science and Engineering department of a large, private US university. The key conclusions of this study suggest that embedding engineering computational literacy i.e., the use of computer software and associated modeling and simulation processes within the curriculum, led to increased student engagement with disciplinary concepts, along with positive effects on their self-beliefs and academic performance.

Preliminary findings from an analogous project conducted by Valenzuela, Smith, Reece and Shannon (2010) to evaluate the effect of incorporating programming skills within junior and senior year Industrial Engineering courses at a large, public US university, also indicated favorable reception of such curricular refinements from both staff and students. To obtain these results, the authors adopted a case-studies approach, using data collected through feedback surveys from students and faculty members, while also analyzing students' academic performance in these modules.

A more recent research study by Shoaib, Cardella, Madamanchi, and Umulis (2019) examined the challenges and aspects of computational thinking (CT) competencies such as data analysis, algorithm design, simulation, testing and debugging within an analytical problem-solving classroom activity of a sophomore-level thermodynamics course for biomedical engineering students at a major public, US university. A combination of classroom observations and artifact-based interviews revealed that students in fact, found the modeling and derivation process of developing analytical solutions to be the most demanding, occurring much before the pseudo-code development stage of the activity.

The wide variety of specific skills and concepts associated with computational literacy motivate the development of concept inventories for this topic.

Concept Inventories

Courses designed in accordance with the principles of cognitive theory involve deep assimilation of concepts in order to link in effectively with learners' pre-existing knowledge. In this regard, concept inventories possess immense value by serving as effective measures to gauge acquired

conceptual knowledge (Streveler, Litzinger, Miller, & Steif, 2008). They may be best described as "research-based distracter driven multiple-choice instruments" (Lindell, Pea & Foster, 2007, p. 14) featuring multiple-choice questions designed to test students' understanding of concepts, while making use of incorrect 'distracter' solutions as options to highlight common student errors and misconceptions.

Touted as a unique tool of assessment finding utility in both the summative evaluation of student learning as well as in the formative planning of instructional design (Reed-Rhoades & Imbrie, 2008), concept inventories are gaining increasing prominence within engineering education. The most well-known of these is the Force-Concept Inventory (FCI) that was developed by Hestenes, Wells, and Swackhamer (1992) focusing on Newtonian mechanics, which has applications not only as a diagnostic tool to identify and clarify student misconceptions, but also as an accurate and reliable indicator to evaluate teaching, in addition to being used as a introductory placement test by colleges and universities.

Following on from this, a variety of concept inventories have been formulated across different engineering subjects such as the Concept Assessment Tool for Statics (CATS) (Steif & Dantzler, 2005), the Statistics Concept Inventory (SCI) (Stone et al., 2003) and the Fluid Mechanics Concept Inventory (FMCI) (Martin, Mitchell & Newell, 2003). Subsequent research has also been undertaken through the administration of each of these inventories, in order to assess the value and effectiveness of each. Steif & Hansen's study (2006), for example, made use of the Statics Concept Inventory to reveal significant correlations between performance scores from the inventory and those from course examinations, with concept-specific inventory sub-scores providing valuable insight into highlighting common misconceptions. Using a "lesson-study" approach, Fraser, Pillay, Tjatindi, and Case (2007) assessed students' difficulties with fluid mechanics concepts through the use of computer simulations and the Fluid Mechanics Concept Inventory, resulting in significant student improvement and positive feedback. In terms of refinements, it was also found that the benefits of concept inventories can be sufficiently enhanced by administering them online to ensure wider access and involvement across universities and colleges (Steif & Hansen, 2007).

Research Question

Towards developing an understanding of the current state of computational literacy learning outcomes in disciplinary engineering programs, we addressed the following research question: What implicit or explicit computational literacy concepts or skills can be gleaned from course descriptions? We focused on a single large public research university to trial our approach and inform future work.

Methods

For this preliminary study, we chose a large, public research university in the Midwestern United States. We collected course descriptions from 8 engineering disciplines within their school of engineering and analyzed these entries based on the explicit or implicit computational content.

Data Collection

To identify a university, we conducted an initial search to determine available public data for syllabi or course descriptions at major universities in the United States. We used the following criteria to conduct this search:

- Large public research university in the United States
- Multiple undergraduate engineering programs
- Common first year engineering program or experience
- Available database of learning objectives, syllabi, or course descriptions

Based on this search, a database of course descriptions from the selected university was identified as reasonably representative of similar programs based on the professional judgement of the cross-disciplinary engineering faculty involved in the project. This institution has about 40,000 undergraduate students and is ranked within the top 100 engineering schools by US News and World Report.

Data Analysis

For this qualitative study, we followed analytical techniques from Miles, Huberman, & Saldaña (2013). Using a spreadsheet, we manually coded over 700 course catalog entries across engineering majors. Through a mix of “in-vivo” and “descriptive” coding (Miles et al., 2013), we flagged any entries about computational or data literacy skills and concepts as “implicit” or “explicit”. This mixed approach was in part to mitigate the lack of detail in online course descriptions and the differentiation in the use of terminology, for example, some course descriptions may list a number of related disciplinary variables that are “measured” and this would collapse to a descriptive code such as “measuring disciplinary-specific variables”. In other cases, we were unable to collapse phrases into anything more descriptive than how they appeared in the text, for example “application of computational methods to problem solutions” appeared in a course description for a chemical engineering course. While this clearly indicates the expectation that computational methods will be included, without additional information, we cannot assume beyond that general claim.

The coding process was collaborative and iterative, often going back over majors we had already discussed to reconsider the course content. We made analytical memorandums to track our progress and decisions (Miles et al., 2013). After completing our coding of the course descriptions in the spreadsheet, we aggregated and organized our codes to help build our interpretations of cross-cutting concepts.

By operationalizing computation in our context, we mean applying mathematical or engineering concepts to data on a computer. We purposely exclude word processing but include the use of spreadsheets in this definition. We also excluded listings from independent studies and special topics from our data, since course descriptions for such courses tend to be generic. Our case site actually had programs focused solely on computation data and computer science. For this study, we did not include these majors, but their existence affirms our motivation to establish a baseline of computational skills and concepts in first year programs, as more of these types of majors come into engineering.

Findings and Discussion

Extracting concepts from course descriptions

Course descriptions are limited by nature in what they can tell us and some were written in a different style than others (e.g. talking about experiences versus listing topics). There is disciplinary jargon that we need help unpacking to identify what the actionable concept or skill might be (e.g. machines and power laboratory). The disciplines of civil, environmental, transportation, and electrical engineering fell within the background knowledge of the researchers on this project. We knew that certain topics would most likely be software-based because of our experiences with these types of courses. Although we did periodically look up unknown discipline-specific terms as we coded, this leads us to believe we may be missing items from other disciplines and motivates our future work to engage with experts across these different majors. There were less explicit codes for biological systems engineering and fewer codes for applied engineering (business), but these programs had a lower number of courses overall. It is also important to note that we only extracted course descriptions from a single university for this study. Although we outline our decision-making process for choosing this exemplar university, additional universities will be formally recruited as this project progresses.

Towards identifying cross-cutting concepts

Acknowledging the limitations of this preliminary investigation, we were still able to identify several cross-cutting computational literacy concepts through the course descriptions. These fell into the overarching theme of mathematical representations of real-world systems (e.g., matrix

analysis, mathematical modeling). Modeling is a foundational engineering concept, and when this happens in the context of problem solving on a computer, there is an additional step of mapping abstract representations to something a computer can represent. It is notable that teaching mathematical modeling and problem solving are both challenging, even without the addition of computing concepts (Wedelin, Adawi, Jahan & Andersson, 2015; Loch & Lamborn, 2016).

Cross-cutting themes:

- Optimization
- Modeling (e.g., mapping disciplinary specific problems to computer representations such as mechanical drafting or circuit diagramming)
- Working with uncertainty (i.e., probability and statistics)
- The use of industry software
- Numerical methods

Each one of these will need to be further unpacked before they can inform specific skills that might be included in an introductory course. For example, while "numerical methods" in one form or another was a common reference, arguably most problems solved on a computer will be done using numerical methods. In fact, this highlights the importance of identifying which numerical methods might be most used in future courses so that we may lay the groundwork in first-year courses.

Other themes could likely be further distilled into those that are embedded in engineering problem solving in general, and those that are unique to problem solving on a computer. For example, "working with uncertainty" is generally understood to be a fundamental requirement of engineering as the real-world is an uncertain place. However, computers are deterministic by design and thus we find ourselves in the predicament of representing imprecise data on a device that is built on assumptions of exactness. This introduces a number of additional considerations to the problem solving process when implemented on a computer that would not otherwise be present, such as the effects of discretizing continuous time values, and increases the salience of other engineering skills such as determining when "close" is "close enough".

Conclusion and Future Work

This was the first step towards establishing a taxonomy. We found that course descriptions yield limited insights into the assumed computational skills required in engineering courses. This motivates the next phase of this project which will include a quantitative survey across institutions as well as interviews with faculty members in the disciplines. We will use public records to contact department heads of engineering departments at public research institutions,

asking that they share an invitation to participate with faculty in their department who teach core disciplinary courses. Our interviews will be anchored in example problems identified by the participants to be representational of problems covered in core disciplinary courses. Example questions include:

- How would you define data literacy and computing literacy (in your discipline...in general…)?
- What are some common mistakes/misconceptions that you see with student's work for these representational problems?
- Based on the process you have just described, can you summarize the computations or data analysis skills that students are expected to perform to be successful in the class?

Both literature and the authors' personal experiences suggest that the close relationships between computational literacy, mathematical modeling, and problem solving skills present significant challenges for teaching these skills, especially in a first-year course. Cognitive load theory (Paas, Renkl & Sweller, 2003) is a useful framework for describing this challenge: As most first-year students are still struggling with problem solving and mathematical modeling, these skills require a significant amount of students' finite cognitive energy. If for a given problem, students are just managing to understand the context and relationships described by relevant mathematical models, adding any additional load such as working in an unfamiliar computing environment, can overwhelm students leading to frustration. One strategy for creating more supportive learning environments would involve separating out problem solving and mathematical modeling that is linked to the problem in general with those concepts specifically linked to the computing context. For example, most traditional engineering problems are described in terms of continuous time equations, but in transitioning these problems to use measured data, they become discrete-time problems. This transformation is not trivial, and the resulting equations, while similar, are not equivalent to the continuous time versions students may be familiar with.

With only limited time in first-year courses to prepare students for success, it will be important to identify a minimum set of computing skills that will best support most engineering disciplines. The theory of threshold concepts can be useful to help decide which skills are included in a first-year course. Male & Baillie (2014) characterize threshold concepts as those that are both transformative and troublesome. Transformative in that mastery of these concepts results in a significant reshaping of student viewpoints and abilities, and troublesome because these students tend to struggle with these concepts. For example, mastering time and frequency domain transformations leads to students recognizing that a given problem can have multiple representations. Solutions that are complex in one domain may be straightforward in another. Once identified, threshold concepts specific to computational problem solving become strong candidates for inclusion in a first-year course.

References

Forte, A., & Guzdial, M. (2005). Motivation and nonmajors in computer science: Identifying discrete audiences for introductory courses. *Education, IEEE Transactions On, 48(2)*, 248–253. https://doi.org/10.1109/TE.2004.842924

Fraser, D. M., Pillay, R., Tjatindi, R. L., & Case, J. M. (2007). Enhancing the learning of fluid mechanics using computer simulations. *Journal of Engineering Education*, *96*(4), 381– 388. https://onlinelibrary.wiley.com/doi/epdf/10.1002/j.2168-9830.2007.tb00946.x

Hestenes, D., Wells, M., & Swackhamer, G. (1992). Force concept inventory. *The Physics Teacher*, *30*, 141–158 https://aapt.scitation.org/doi/pdf/10.1119/1.2343497

Lindell, R. S., Pea, E., & Foster T.M. (2007). Are They All Created Equal? A Comparison of Different Concept Inventory Development Methodologies, *American Institute of Physics Conference Proceedings, 883*(14), 14-17. https://doi.org/10.1063/1.2508680

Loch, B., & Lamborn, J. (2016). How to make mathematics relevant to first-year engineering students: Perceptions of students on student-produced resources. *International Journal of Mathematical Education in Science and Technology, 47(1),* 29–44. https://doi.org/10.1080/0020739X.2015.1044043

Magana, A. J., Falk, M. L., Vieira, C., & Reese, M. J. (2016). A case study of undergraduate engineering students' computational literacy and self-beliefs about computing in the context of authentic practices. *Computers in Human Behaviour 61*, 427-442. http://dx.doi.org/10.1016/j.chb.2016.03.025

Male, S., & Baillie, C. (2014) Engineering Threshold Concepts as an Approach to Curriculum Renewal. In Johri, A., & Olds, B. (Eds.) *Cambridge Handbook of Engineering Education Research*, (pp. 393-408). Cambridge University Press. https://doi.org/10.1017/CBO9781139013451

Martin, J., Mitchell, J., & Newell, T. (2003). Development of a concept inventory for fluid mechanics, *33rd Annual Frontiers in Education (FIE) Conference.* https://doi.org/10.1109/FIE.2003.1263340

Maybee, C., & Zilinski, L. (2015). Data informed learning: A next phase data literacy framework for higher education. *Proc. Assoc. Info. Sci. Tech., 52*: 1-4. https://doi-org.libproxy.ucl.ac.uk/10.1002/pra2.2015.1450520100108

Miles, M.B., Huberman, A.M., & Saldaña, J. (2013). *Qualitative data analysis: A method sourcebook* (3rd ed.) SAGE Publications, Inc. https://us.sagepub.com/en-us/nam/qualitative-data-analysis/book246128

Paas, F., Renkl, A., & Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. *Educational Psychologist, 38(1)*, 1–4. https://doi.org/10.1207/S15326985EP3801_1

Prado, F. J., & Marzal, M. A. (2013). Incorporating data literacy into information literacy programs: core competencies and contents. *Libri*, *63*(2), 123-134. https://core.ac.uk/download/pdf/288499712.pdf

Reed-Rhoads, T., & Imbrie, P. K. (2008). Concept inventories in engineering education. *National Academies Board on Science Education, Evidence on Promising Practices in Undergraduate Science, Technology, Engineering, and Mathematics (STEM) Education Commissioned Papers.* http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.674.4703&rep=rep1&type=pdf

Reid, K., & Reeping, D. (2014), A Classification Scheme for "Introduction to Engineering" Courses: Defining First-Year Courses Based on Descriptions, Outcomes, and Assessment, *ASEE Annual Conference & Exposition, Indianapolis, Indiana*. https://peer.asee.org/19916

Shoaib, H., Cardella, M., Madamanchi, A., & Umulis, D. (2019). An Investigation of Undergraduates' Computational Thinking in a Sophomore-Level Biomedical Engineering Course. *International Journal of Engineering Education (IEEE) Frontiers in Education Conference (FIE), Covington,* 1-4. https://ieeexplore.ieee.org/document/9028503/

Steif, P. S., & Dantzler, J. A. (2005). A statics concept inventory: Development and psychometric analysis. *Journal of Engineering Education*, *94*(4), 363–371. https://onlinelibrary.wiley.com/doi/epdf/10.1002/j.2168-9830.2005.tb00864.x?saml_referrer

Steif, P. S., & Hansen, M. A. (2006). Comparisons between performances in a statics concept inventory and course examinations, *International Journal of Engineering Education*, *22*, 1070–1076. http://www.andrew.cmu.edu/user/steif/papers/Statics_Concept_Inventory_Steif_Hansen_2005.pdf

Steif, P. S., & Hansen, M. A. (2007). New practices for administering and analysing the results of concept inventories. *Journal of Engineering Education*, *96*, 205–212. https://onlinelibrary.wiley.com/doi/epdf/10.1002/j.2168-9830.2007.tb00930.x

Stone, A., Kirk A., Reed-Rhoads, T., Murphy, T. J., Shehab, R. L., & Saha, C. (2003). The Statistics Concept Inventory: A Pilot Study. *ASEE/IEEE Frontiers in Education Conference (FIE), Boulder, Colorado.* https://www.researchgate.net/profile/Kirk-Allen/publication/216771545_The_Statistics_Concept_Inventory_a_pilot_study/links/0912f4fad2cdf291ef000000/The-Statistics-Concept-Inventory-a-pilot-study.pdf

Streveler, R. A., Litzinger, T. A., Miller, R. L., & Steif, P. (2008). Learning conceptual knowledge in the engineering sciences: Overview and future research directions. *Journal of Engineering Education*, *98*(3), 279–294. https://onlinelibrary.wiley.com/doi/epdf/10.1002/j.2168-9830.2008.tb00979.x

Valenzuela, J., Smith, J., Reece, B., & Shannon, D. (2010). Integrating Computer Programming Technologies Into The Industrial Engineering Curriculum. *Annual Conference & Exposition, Louisville, Kentucky*. https://peer.asee.org/15902

Wedelin, D., Adawi, T., Jahan, T., & Andersson, S. (2015). Investigating and developing engineering students' mathematical modelling and problem-solving skills. *European Journal of Engineering Education, 40*(5), 557–572. https://doi.org/10.1080/03043797.2014.987648

Appendix

Major	Explicit	Implicit
Biosystems Engineering	• "geographic information systems"	• "steady-state and stability analysis" • "sensors and instrumentation" • "linear programming" • "optimization techniques" • "global position systems" • "water budget analysis" • "engineering economics" • "design project related to food and agricultural industries" • "watershed modeling" • "Geospatial data collection" • modeling • design
Civil and Environ. Engineering	• "structural analysis computer software" • "CAD software" • "matrix methods of structural analysis...application software" • "Geometric design of highways...CAD systems" • "solving systems of differential equations", matrix? • "programming methods" • "Physical modeling" • "numerical techniques" • "visualizing the results"	• "engineering economics" • "probability, statistics" • transportation engineering computation • "constructability, cost, and schedule" • "statistics used in laboratory analysis" • design project • "environmental measurements laboratory" (Environmental)
Chemical Engineering	• "Application of computational methods to problem solutions • transport phenomena • "computer-aided design methods" • mass transfer and separations • "process simulation"	• engineering statistics • model building • statistical quality control • "integration of control theory with modern practice" • design of control systems • "mathematical programming methods for optimization" • "flowsheet layout and

		optimization" ● "instrumentation and control systems" ● "application of statistics"
Electrical and Computer Engineering	● embedded coding ● "computer-aided design" ● circuit layout ● CAD ● logic design tools ● "using standard software packages" ● electrical modeling ● "SPICE software" ● modeling ● "hardware and software verification" ● "SPICE macro-modeling" ● VLSI Design ● HDL design ● "Control laboratory" ● VLSI ● "simulation of nonlinear control systems" ● "performance analysis and testing of data converters" ● "simulation" ● power converters ● "cyber-physical systems" ● "discrete event simulation projects" ● "computer projects" ● "analysis of real physiologic signals" ● "projects laboratory in communication systems" ● "Simulations" ● "microelectronic" ● "contemporary design tools"	● applications of linear algebra ● "applications and design" ● "Measurement of the properties of antennas and microwave networks" ● "design of switched capacitor circuits" ● robotics ● "machines and power laboratory" ● "biomedical imaging" ● "design"
Mechanical Engineering	● "computer-aided three-dimensional design" ● "Computer-based analysis in support of design"	● "numerical methods" ● "heat transfer laboratory" ● "modeling of thermal equipment" ● "predictive models"

	• machine elements and mechanical systems • "Advanced 3-D solid modeling" • "Computer based design projects" • thermal systems • "Application of commercial software to computational fluid dynamics problems" • "simulation and optimization of thermal systems" • "finite element models" • computer aided design • "algorithms for constrained and unconstrained optimization" • "Computational methods for analysis, design, and optimization of structural component" • "geometric modeling" • "finite element analysis" • "structural optimization"	• alternative energy systems • "measurement of stress, strain, vibration, and motion" • "capstone design"
Applied Engineering (Business)	• "computational analysis tools for large data sets" (E)	• "applied engineering sciences curricular elements" (I)