Received: 8 March 2022 | Revised: 10 August 2022 | Accepted: 24 September 2022

DOI: 10.1002/cae.22580

RESEARCH ARTICLE

WILEY

Data science knowledge integration: Affordances of a computational cognitive apprenticeship on student conceptual understanding

Matilde Sánchez-Peña[1] | **Camilo Vieira**[2] | **Alejandra J. Magana**[3]

[1]Department of Engineering Education, University at Buffalo, New York, Buffalo, USA

[2]Department of Education, Universidad del Norte, Barranquilla, Colombia

[3]Department of Computer and Information Technology and School of Engineering Education, Purdue University, Indiana, West Lafayette, USA

Correspondence
Alejandra J. Magana, Department of Computer and Information Technology and School of Engineering Education, Purdue University, West Lafayette, IN 47906, USA.
Email: admagana@purdue.edu

Funding information
National Science Foundation, Grant/Award Numbers: CMMI # 2134667, DBI # 2120200

Abstract

This study implements a computational cognitive apprenticeship framework for knowledge integration of Data Science (DS) concepts delivered via computational notebooks. This study also explores students' conceptual understanding of the unsupervised Machine Learning algorithm of K-means after being exposed to this method. The learning of DS methods and techniques has become paramount for the new generations of undergraduate engineering students. However, little is known about effective strategies to support student learning of DS and machine learning (ML) algorithms. The research questions are: How do students conceptualize their understanding of an unsupervised ML method after engaging with interactive visualizations designed using the computational cognitive apprenticeship approach? How do the affordances of the interactive visualizations support or hinder student knowledge integration of an unsupervised machine learning method? Design-based research allowed for the iterative design, implementation, and validation of the pedagogy in the context of a working classroom. For this, data collection methods often take the form of student artifacts. We performed a qualitative content analysis of students' written responses and reflections elicited during the learning process. Results suggest that the computational cognitive apprenticeship promoted knowledge integration. After interacting with the computational notebooks, most students had accurate conceptions of the goal and the nature of the method and identified factors affecting the output of the algorithm. Students found it useful to have a concrete representation of the method, which supported its conceptual understanding and showcased the acquisition of strategic knowledge for its appropriate execution. However, we also identified important misconceptions students held about the algorithm.

KEYWORDS
cognitive apprenticeship, data science, engineering education

1 | INTRODUCTION

The ever-increasing capacity to obtain and store data [73] has derived from the pressing need to train the next generation of citizens that can create and use valuable data-generated knowledge [30]. Data Science (DS) knowledge is now endorsed as the main tool to transform the wealth of available data into informed decisions [23]. A report from the National Science Foundation titled Realizing the Potential of DS [3] defined DS as a new field "that focuses on the process and systems that enable the extraction of knowledge or insights from data in various forms, either structured or unstructured" (p.2). Therefore, the goal of infusing the new field of DS within higher education has been argued as an endeavor to be prioritized as the demand for DS professionals keeps increasing [32].

Consequently, DS has become a crucial tool for the research and educational community [3]. Educational research has explored diverse topics using DS tools, including student admission [72] and retention [71], as well as student approaches to problem-solving [65]. However, less is known about the processes involved in DS education itself, with the persistent question still being "How do we train [a] workforce of professionals who can use data to its best advantage?" (p.2) [3]. Most of the existing discussions around DS education have been devoted to framing a definition [22], pertinent content of DS and ethics of conducting DS [19], or evaluation of programs [33, 41]. Furthermore, only recently the computing competencies for undergraduate curricula in DS have been formally discussed and agreed upon [18]. However, less attention has been paid to supporting the cognitive process involved in learning DS concepts and how such process could be supported. To strengthen the training of the next generation of data scientists, we need to develop a better understanding of such processes. Accumulated knowledge from the learning sciences [70] would benefit DS education, making it more likely to democratize DS knowledge [34].

Students in engineering and other Science, Technology, Engineering and Mathematics (STEM) areas are uniquely poised to achieve the fastest gains in learning DS due to the overlap of their regular training in analytical, problem-solving, and computational skills [38]. Therefore, focusing on the difficulties faced by engineering students when learning DS techniques could advance the understanding of the key challenges of learning the topic for the general population.

Machine learning (ML) lies at the intersection of computing and statistics and constitutes an essential component of DS [51]. However, little is known about how students learn ML techniques [18, 17]. While using ML methods may enable engineering and other STEM professionals to make important decisions and solve complex problems, it may also be misused if they do not understand the underlying mechanisms of these methods [56]. These methods use datasets to create models that represent phenomena, but many of them will always offer an output whether it correctly represents a phenomenon or not. The user is responsible for evaluating and interpreting the outputs. The user needs to be aware of the underlying assumptions of the method to be able to interpret the outputs. Also, the data set may include some outliers or may have missing data, which usually have an impact on the model output [28].

Furthermore, the training data itself may include some bias that we as a society might not want to reproduce. For instance, a study exploring biases in word embeddings showed that an algorithm might complete the analogy: "man is to Computer Programmer as Woman is to... Homemaker" [4]. Having a biased input data set will lead to a biased model. Therefore, any teaching practices to integrate ML methods into engineering and STEM education should not just use them as a black box but also discuss the assumptions, possible biases, and implications.

Computing skills are essential to DS. Together with math and statistics knowledge and domain knowledge, they are recognized as the building blocks of DS [51]. Therefore, knowledge of computing learning might contribute significantly to the understanding of how DS is learned. To take advantage of such overlap, this study contributes to the body of knowledge by exploring how the computational cognitive apprenticeship framework [21] supports knowledge integration in DS when implemented into interactive visualizations through computational notebooks. Computational notebooks have been used extensively with different learning goals [5, 44, 53]. Preliminary results suggest that students find them useful in integrating theoretical concepts and practice activities. However, there has been limited exploration of their pedagogical design using learning theories. This study addresses such a gap and identifies persisting misconceptions that hinder students' learning of DS concepts, as misconceptions have been documented to limit student learning [24]. For our goals, we explored students' conceptual understanding of an ML algorithm after engaging with the interactive visualizations and evaluated their perceived affordances of learning DS under the lens of knowledge integration. The research questions for this study are:

- How do students conceptualize their understanding of an unsupervised ML method after engaging with interactive visualizations designed using the computational cognitive apprenticeship approach?

- How do the affordances of the interactive visualizations support or hinder student knowledge integration of an unsupervised ML method?

2 | BACKGROUND

2.1 | Status of DS education

DS is a young field that combines knowledge of computing, statistics, and particular expertise domains into the analysis of data to create value [51]. Educational institutions have benefited significantly from the use of DS techniques as a tool to facilitate administrative tasks, improve instruction, and identify barriers to student learning [11]. Nevertheless, little attention has been paid to how students learn DS concepts. In fact, it has been recognized that the variety of nature and quality of available resources to learn DS can become a barrier for learners [21]. Furthermore, a lot has been argued about the relevance of learning DS in the current data-rich society, not only by those trained in computing and statistics but also by the general population [34]. This need requires understanding the processes involved in DS education.

Most of the accumulated evidence on ML education is focused on teaching ML to young students through a variety of interventions. A recent review identified 30 instructional units that focused on teaching ML basics at the K-12 level [42]. Most of them were developed as extracurricular activities, workshops, or summer camps and considered topics including the definition of learning, specific supervised algorithms such as linear regression, specific unsupervised algorithms such as clustering, as well as the limitations and social implications of ML.

The wide spectrum of interventions with respect to the topics, learning objectives, and pedagogies implemented reflects the growing interest in taking ML techniques to the younger generations. These initiatives often used block-based programming languages, as was the case for the work presented by Estevez et al., [25], who used Scratch for their pedagogical intervention teaching clustering and artificial neural networks following a design-based research (DBR) approach grounded in experiential learning theory. Their intervention proved effective in changing students' perceptions of artificial intelligence systems [25], which are derived from the use of ML techniques. While extensive documentation exists of such K-12 interventions, scalability to reach all students is still a distant goal, making it also relevant to address ML education at the undergraduate level.

At the undergraduate level, some attention has been paid to how to teach ML to majors outside of computing and statistics. In their recent work, Sulmont et al. [55] identified elements that instructors found easy or difficult to teach ML. Their results, presented under the Structure of Observed Learning Outcomes (SOLO) taxonomy, showed that elements in the lower scale of the taxonomy, such as those related to algorithm learning and execution, were reported as easy to teach. On the other hand, those elements mapped to higher levels of the taxonomy, such as making design decisions and comparing/contrasting models, were harder to teach. To identify students' preconceptions and barriers to their learning as well as the tactics used by instructors to address such difficulties, the authors also reported on the analysis of the same data using the Pedagogical Content Knowledge (PCK) framework. Faculty-cited students acknowledged that ML was important, but their preconceptions of the nature of ML were limited to what they heard on social media. In addition, students usually considered ML implementation outside of their reach as the main barriers they identified were their own limited math and programming knowledge [54]. Finally, the instructors identified a variety of strategies to support students' advancement through their learning, such as working on simple problems by hand, simulating algorithms, strategically choosing data sets, and the use of visualizations [54]. Other work has used embodied systems, such as Lego platforms, as viable alternatives to effectively teach ML [58]. While these works provide valuable evidence on ML education for noncomputing and statistics majors, their cumulative evidence was built from the faculty perspective and lacked students' actual learning perceptions and experiences.

Current efforts are on their way to generating a deeper understanding of the misconceptions related to DS learning and acknowledging the experiences of students, faculty, and industry professionals. The project *Investigations of Student Difficulties in DS Instruction* is identifying student misconceptions and prior knowledge in DS courses; it will also confirm such foundational knowledge from experts and develop a concept inventory that would support the training of the next generations of DS professionals [69]. Attending to the limited understanding of the challenges involved in learning DS in general and ML techniques in specific, we introduce an approach that merges the theory of knowledge integration within the pedagogical framework of computational cognitive apprenticeship [26]. We argue that this approach has the potential to support the learning of students outside of computing and statistics majors. Our research design is proposed to capture the student perspective while learning unsupervised ML techniques

with the goal of informing effective pedagogies to support student learning of ML.

2.2 | Computational notebooks

Computational notebooks have gained traction in recent years as a popular format for instruction in higher education. Despite the original goals for their use, which were focused on the expansion of computational literacy and research reproducibility [49, 52], the potential for their use in educational spaces has been extensively documented [5, 44, 53], [47, 74]. Examples of their use now include topics such as artificial intelligence [44], chemical engineering [5,20], optimization [53], digital signal processing [74], computational modeling [47], and remote lab experimentation [7].

However, most of the existing literature on the use of computational notebooks for teaching challenging concepts is limited to reporting the technical implementation of the notebooks [e.g., [53, 74]]. Few of them include the learning goals of the summarized courses [e.g., [7]] and even less report on any output related to the student experience, such as student engagement [53], or student satisfaction [47, 74]. In addition, some criticism exists that computational notebooks can also propagate the dissemination of negative learning outcomes, such as poor coding practices [67]. As a result, a set of recommendations has been developed for the use of computational notebooks in science reproducibility [48]; it is important to make a clear distinction between the use of computational notebooks for the development of programming skills and work organization, which is their usual goal in computer science spaces [49, 52], and their use for achieving specific learning goals in other areas of knowledge [5, 44, 53], [47, 74].

This study focuses on the latter, with a specific goal of supporting students in learning DS and ML algorithms. While critics of computational notebooks exist based on (a) their goal of enhancing productivity and reproducibility of coding [10, 29], and (b) suggestions for improving their design in such context [50], only recently, research studies have focused on introducing guidelines to create computational notebooks for educational purposes. One notable example is the work in physics education, where computational notebooks have been used as a pedagogical approach to introduce writing and argumentation practices [45] and computational literacy, both in the form of computational essays [46].

At the time of this writing, no publications were identified showcasing the design of computational notebooks from a pedagogical perspective for the development of DS skills. Nor do any other identified publications consider students' conceptual understanding as a performance measure of their use. This study fills this gap by building computational notebooks using research-based pedagogies and aims to gauge student learning derived from their use. This small-scale study supports building evidence of the potential for the theory-driven design of computational notebooks that can support the scalability for the creation of tools that effectively support student learning of DS and ML concepts.

3 | THEORETICAL FRAMEWORK

The theoretical framework for this study is knowledge integration [36]. Knowledge integration refers to the process of incorporating and synthesizing multiple representations into a common body of existing knowledge, thus resulting in the development of an integrated understanding of a complex domain [35]. Knowledge integration can be achieved via the design of curricula that integrates complex topics enabled by technological tools [35]. To create learning experiences that promote knowledge integration, students must be exposed to opportunities to compare and contrast multiple ideas and representations of those ideas as part of the instruction [12]. These ideas may refer to concepts, principles, or practices across disciplines. For example, learning interventions that combine physics concepts with mathematics practices may better promote knowledge integration [12].

To guide the design of learning interventions that promote knowledge integration, learning scientists have identified a collection of instructional patterns that can help learners make connections among ideas and in the process develop a coherent understanding [37]. These patterns include: (a) *eliciting ideas* by helping students bring their prior experiences to the learning context; (b) *adding new ideas* by providing meaning-making mechanisms that enable learners to make connections between what they already know and what will be delivered in the instruction; (c) *distinguishing ideas*, by helping learners see how their existing ideas relate to or conflict with normative ideas during instruction; and (d) *sorting out ideas*, by providing opportunities to learners to refine their knowledge through analysis and reflection [12].

The implications for this study relate to the intended learning outcomes of the course, where students need to apply DS practices to make meaning in a specific field or application area. To do this successfully, knowledge integration processes are needed because students need to bring together their disciplinary knowledge, apply their computing skills, and make meaning of information resulting from those two bodies of knowledge.

4 | PEDAGOGICAL FRAMEWORK

We used the computational cognitive apprenticeship framework [26] as the pedagogical framework to embody the patterns for eliciting ideas, adding new ideas, distinguishing ideas, and sorting out ideas as prescribed by knowledge integration. The computational cognitive apprenticeship framework builds upon decades of research on cognitive apprenticeships [1]. Cognitive apprenticeship emerges from comparisons between traditional classroom instruction and the cultural tradition of apprenticeship, which more prominently features observation, coaching, and successive approximation [15]. Collins et al. [15] posited that "apprenticeship embeds the learning of skills and knowledge in the social and functional context of their use" (p. 1). This focuses not just on "how to do," that is, the mechanical steps involved in completing a task, but "how to think" through and develop an understanding of tasks within a complex social environment [6, 14, 15].

Cognitive apprenticeships provide guidelines to design learning environments that merge disciplinary content and practices being taught, successful deployment of pedagogical methods, sequencing of learning activities, and social characteristics of the learning environment [16]. These teaching methods entail: (1) *modeling*, where the instructor demonstrates how to perform a task; (2) *coaching*, including observation and facilitation at the moment students perform a task; (3) *scaffolding*, regarding supporting methods to help students perform a task; (4) *articulation*, consisting of instructors encouraging students to state their knowledge and thinking; (5) *reflection*, where instructors enable students to compare their performance with experts' approaches; and (6) *exploration*, consisting of instructors prompting students to solve problems on their own [16].

Informed by more than 15 years of DBR in teaching computation and DS at the undergraduate level, we have adapted the cognitive apprenticeship model to train the next generation of the computational and data-enabled workforce [26]. Such research has resulted in a set of guidelines, as described in Table 1.

Our prior work identified an effective way to implement "content," "method," "sequencing," and "sociology" through anchored instruction that integrated computation within disciplinary engineering practices [40] coupled with scaffolding approaches [62]. Anchored instruction [57] involved the integration of computational and DS practices within disciplinary problems in science and engineering [39]. Such opportunities enable students to integrate knowledge in science and engineering domains along with practices in computing and mathematics. However, this integration of knowledge was challenging for students, so scaffolding approaches need to be considered.

Scaffolding approaches included worked-out examples, which helped students of various backgrounds to succeed in accomplishing the challenging task of integrating newly acquired disciplinary expertise with newly acquired programming skills [63]. In the most successful deployment, students were provided with access to step-by-step examples of sample solutions to programming problems [64, 66]. These solutions included the conceptualization of the problem, the algorithm development, and the programming, resulting in a final working but uncommented code. We identified that a very effective strategy to engage students in understanding worked-out examples was having them write in-code comments to self-explain the examples, resulting in students' meaning-making [61].

5 | LEARNING DESIGN

To deliver our curricular approach, we took advantage of interactive computational documents [27]. Computational documents (see Figure 1) permit exploring,

TABLE 1 Computational cognitive apprenticeship dimensions

Dimension	Definition
Content	Teaching disciplinary domain knowledge in combination with data science techniques and tools. The content also includes instruction on expert modes of thinking relevant to how scientists and engineers use data.
Method	The ways in which a material is delivered to learners, including scaffolding approaches such as code-snippets, test cases, and worked examples. Instructional methods can be made more adaptive and student-directed in computational settings through software-realized scaffolding, online resources, and networks (GitHub, NanoHub, EdX, zoomi. ai).
Sequencing	The order in which activities are presented to learners to make the learning process as natural as possible (i.e., increasing complexity, increasing diversity). Online tools and resources are used to make the activities more student-directed.
Sociology	Utilizing anchored instruction by contextualizing the learning experiences within authentic tasks that occur in real-life contexts and establishing meaningful associations between learning experiences and a discipline's knowledge, skills, and practices.

K-Means

Usage-Rigts

These materials were created for educational purposes. They are licensed under creative commons.

Learning outcomes

By the end of this activity, you will be able to:

- Use R to identify clusters in a data set using the kmeans method
- Create simple plots to depict the clusters in a data set

K-means

K-means is an unsupervised machine learning algorithm that takes a N-dimensional data set and mimizes a distance measure for partitioning the data set into *k* groups or clusters. The distance measure is the *sum of squared Euclidean distances* between the data points and the.

Procedure

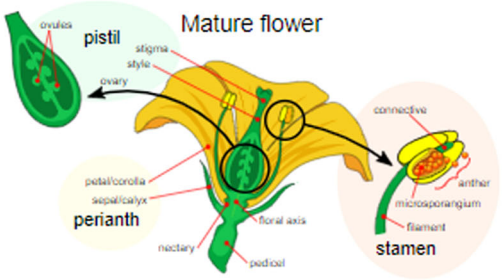
1. Initially, you randomly pick *k* centroids (or points that will be the center of your clusters) in *d-space*. Try to make them near the data but different from one another.
2. Then assign each data point to the closest centroid through a distance measurement.
3. Move the centroids to the average location of the data points (which correspond to users in this example) assigned to it.
4. Repeat the preceding two steps until the assignments don't change, or change very little.

Example

For this example, we are going to use a pre-loaded dataset called *iris*. This dataset includes the length and width of the sepal and the petal for 150 iris flowers belonging to three species of plants (setosa, virginica, versicolor).

Here is you can see the difference between a sepal and a petal:

```
knitr::include_graphics("https://upload.wikimedia.org/wikipedia/commons/7/7f/Mature_flower_diagram.svg")
```



Let's start by exploring the columns in the dataset. Execute the following instruction by pressing *Run*

```
Code  Start Over                                   Run Code
1 iris
2
3
```

FIGURE 1 Sample of a computational notebook with embedded guidance

authoring, and executing code within a single document. Computational documents also give their users opportunities to share work, keep track of details, and collaborate in the process [68]. Although such technological advances make possible the delivery of quality programming content more accessible for teaching and learning by supporting (a) visualization of code execution, (b) feedback generation to identify mistakes, and (c) code grading automation, there is still a lack of learning materials that integrate all these affordances into a single solution [31].

We took advantage of computational documents to deliver content as well as pedagogy in the form of a computational cognitive apprenticeship. We have

already identified and pilot-tested its affordances for learning in the context of programming education [60]. K-means was the ML technique we chose to explore the ability of the proposed implementation of the computational cognitive apprenticeship in facilitating students learning of DS concepts. Our rationale was based on (1) its simplicity, which deems the algorithm accessible to novices and experts alike as it offers a clear, intuitive interpretation of its results [25]; (2) its popularity, K-means is deemed one of the most popular unsupervised ML algorithms mainly because of its efficiency [8]; finally (3) its dependency on the parameters used in its performance [8], which offers unique opportunities from a pedagogical perspective since simple misunderstanding or misconceptions can lead to poor execution and interpretation of the algorithm. Therefore, we deemed it a suitable first ML algorithm to implement the proposed computational cognitive apprenticeship approach under the framework of knowledge integration. We used R markdowns with the package LearnR to create the computational notebook. R is a free statistical programming language that already has a built-in implementation of K-means and provides useful visualizations. LearnR is an R package for creating interactive tutorials.

We adopted the computational cognitive apprenticeship approach to inform our learning design as described below. In general, the learning design included (1) a modeling component led by the course instructor to introduce the theoretical foundations of the method; (2) a set of interactive activities with the computational document where the students explored ideas and representations, answered prompts, and completed practice challenges to articulate their learning; and (3) a reflection sheet to connect all the activities and promote knowledge integration.

5.1 | Modeling

The course instructor introduces the concept of DS/ML/AI and the theory behind a specific method (e.g., K-Means) through a short video lecture. The video lecture was 10 min long, with the intended learning outcomes for students to (1) describe the ML algorithm of K-means and its uses and (2) implement the ML algorithm of K-means. It included information on why the algorithm is convenient for managing multiple dimensions of data (i.e., variables) and showcased an example using two dimensions, highlighting the results' differences when using different k. In such recorded lectures, the R commands to execute the algorithm were listed, but its execution was not modeled. The interactive computational document also provided a brief explanation of the algorithm and both a black box and a detailed implementation of the algorithm.

5.2 | Coaching

Students explore the interactive computational document to visualize a sample data set and invoke the method to analyze the results (see Figure 2).

5.3 | Scaffolding

The interactive computational document also demonstrates a step-by-step implementation of the algorithm as well as its iterative visual execution (see Figure 3).

5.4 | Articulation

The students write in-code comments to describe the purpose of specific sections of the code implementing the algorithm. Figure 4 includes a sample code preceded by the practice activity statement: "What is the algorithm doing?" Write comments within the sample code to describe what each line does.

5.5 | Reflection

Students complete a reflection sheet after working on the interactive computational document. This sheet included the following reflection questions: 1. Please describe the goal of K-means and how it works; 2. What are the factors that might affect the results from K-means?; and 3. Are the identified clusters by K-means always meaningful? Why?

This learning environment also addressed the instructional patterns to promote knowledge integration (KI), namely:

(a) Eliciting ideas – for that purpose, students were first asked the following open-ended questions:

(1) Do you think ML or AI is something you will have to use in your career? Do you feel confident in your ability to use these approaches?
(2) Do you think that ML/AI is a positive/negative/or mixed influence on the world? Describe why you feel that way.

Question: Do you see any patterns in the plot?

Code Start Over Run Code

```
ggplot(iris, aes(x=Petal.Length,y=Sepal.Length))+
  geom_point()
```

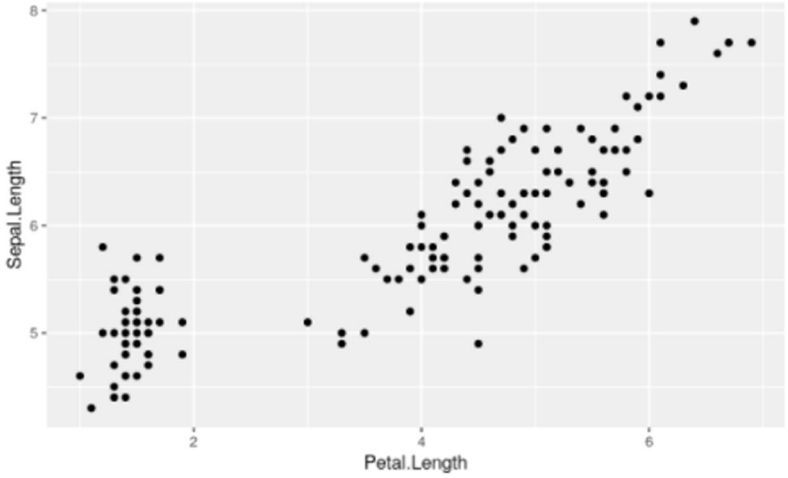


FIGURE 2 Sample of interactive visualization to provide coaching

Code Start Over Run Code

```
numIter=4
dataToPlot<-iris[,c('Sepal.Length','Petal.Length')]
randomCentroids$group<--1
randomCentroids$iteration<-0
dataToPlot$group<-1
dataToPlot$iteration<-0
dataToPlot<-rbind(dataToPlot,randomCentroids)
for(i in 0:numIter) {
  # Compute the new centroids
  randomCentroids[1,]<-c(colMeans(iris[iris$group==1,c('Sepal.Length','Petal.Length')]),-1,i)
  randomCentroids[2,]<-c(colMeans(iris[iris$group==2,c('Sepal.Length','Petal.Length')]),-1,i)
  randomCentroids[3,]<-c(colMeans(iris[iris$group==3,c('Sepal.Length','Petal.Length')]),-1,i)

  # Compute the distances to the new centroids, and identify what cluster each datapoint belongs to
  cluster<- apply(iris, 1, FUN=computeDistance, randomCentroids=randomCentroids[,1:2])
```

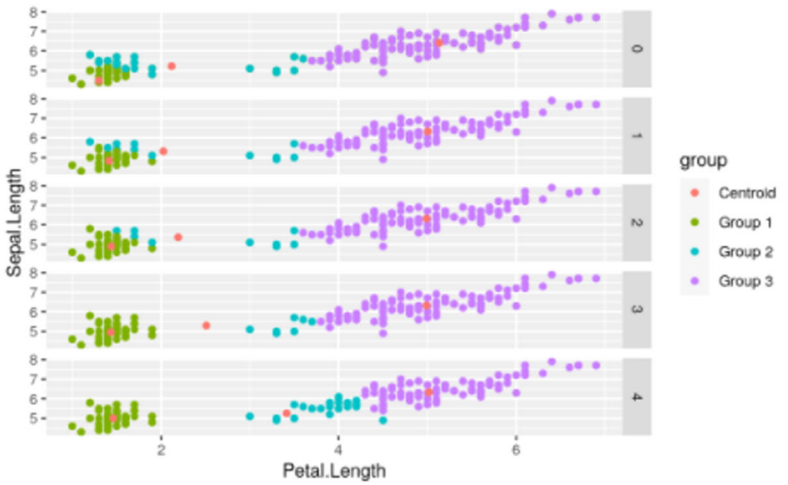


FIGURE 3 Sample of a step-by-step demonstration as scaffolding

After such a reflective opportunity, the ML algorithm is formally introduced through a flipped lecture format, in which students first watch a video of the method, followed by an opportunity to practice it in an active learning setting in class.

(b) Adding new ideas—the interactive computational documents are implemented after introducing the topic, and these interactive documents help to connect the theoretical ideas to concrete representations;

Practice Activity:

What is this algorithm doing? Write comments within the sample code to describe what each line does.

Code | Start Over | Run Code

```
set.seed(1234)

groups<- kmeans(iris[,c('Sepal.Length','Petal.Length')], 3)
groups

iris$group<-groups$cluster

ggplot(iris, aes(x=Petal.Length,y=Sepal.Length, color=factor(group)))+
  geom_point()
```

FIGURE 4 Sample of the code with in-code comments to elicit articulation.

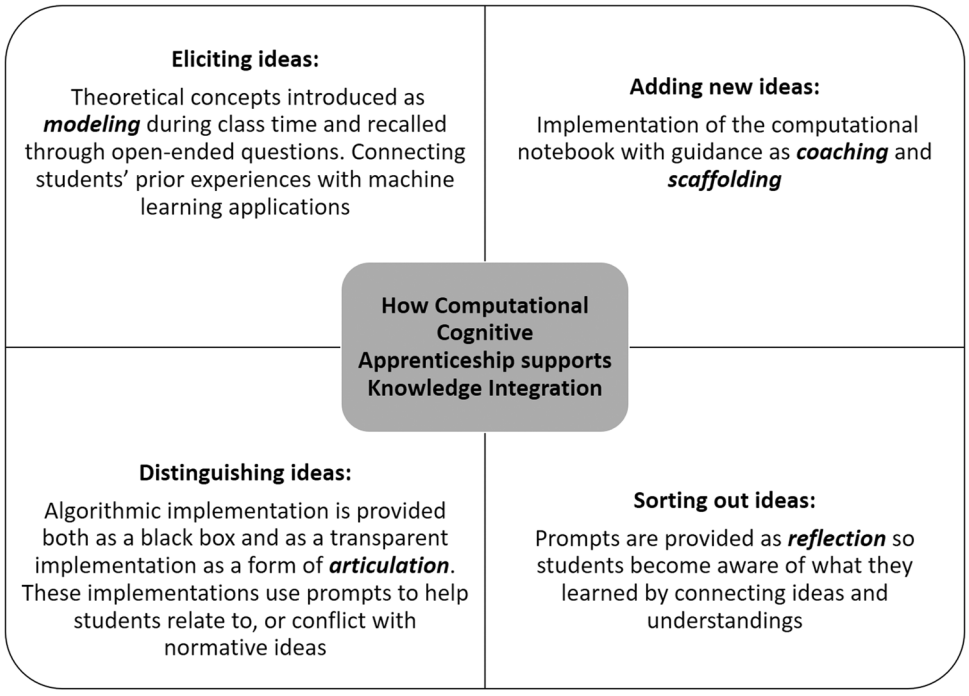


FIGURE 5 Alignment between knowledge integration and the computational cognitive apprenticeship dimensions

(c) Distinguishing ideas—we provide both the black box version of the method as well as the transparent step-by-step model, so the learners see how their existing ideas relate to or conflict with the actual functioning of the algorithm–; and

(d) Sorting out ideas—students complete a reflection sheet that helps students refine their knowledge about the algorithm and how the visualizations supported or hindered their understanding.

A summary of how the computational cognitive apprenticeship framework is envisioned to support knowledge integration is illustrated in Figure 5. The elements in bold and italics represent the dimensions of the computational cognitive apprenticeship framework aligning with each of the stages of knowledge integration.

6 | METHODS

6.1 | Research design

This study implemented the first iteration of a DBR study [2]. DBR is a methodological approach that simultaneously permits addressing a learning need and performing education research in working classrooms. DBR is similar to the engineering design process as it starts with a learning need. Based on the learning need, an educational innovation is designed and grounded in evidence-based practices. The educational innovation is continuously revised and iterated through classroom research.

For this study, we identified the learning need for teaching DS concepts to engineering undergraduate

students sooner and often within the engineering curriculum. We then grounded our learning design as explained in Sections III to V. The last step in the process is to evaluate such innovation in a classroom setting. As the learning innovation is continuously and iteratively evaluated, important contributions of DBR are the validated learning materials, the pedagogical approaches that guided their design, and new knowledge about students' understanding of DS.

6.2 | Context and participants

The presented computational documents in K-means were used for two editions of a senior undergraduate Statistics class. The class was a requirement for a specific engineering program at a Large Public Midwestern institution in the United States. The implementation took place in the Spring 2020 ($n = 10$) and Fall 2019 ($n = 19$) semesters. The class was conducted in a flipped classroom format, in which students watched recordings of the content previous to the class session, in which they engaged in activities to practice the content. The R programming language was used for the course. Based on classroom surveys, all students were novices in DS concepts and methods. We were granted approval from our institutional review board (IRB-2020-1074) to conduct this study, and an exemption was granted for informed consent.

6.3 | Procedures

Students first submitted their responses to the activity of the eliciting idea aimed at unpacking their current knowledge of the topic. Then, students were provided with the flipped lesson on the topic, followed by the active learning session corresponding to the K-Means topic, in which students worked with the computational document in R Markdown (see Figures 1–4 and 6), which covered the different stages of the computational cognitive apprenticeship framework. This activity was part of what was deemed as in-class activities (ICAs) throughout the course. To capture students' experiences working through the computational document, an additional postactivity reflection sheet was provided to answer the following questions:

1. Please describe what the goal of K-means is and how it works.
2. What are the factors that might affect the results from K-means?
3. Are the clusters identified by K-means always meaningful? Why?
4. What do the explored visualizations tell you about the K-means algorithm and its results?
5. How did the visualization help/hinder your understanding of the ML algorithm? How did it change with respect to your understanding based purely on the DS—Intro materials?

The employed R Markdown interactive computational document can be found in Vieira [59].

6.4 | Data collection method

The learning materials were used as a formative assessment for the class; therefore, they were graded in completion. Our data were composed of students' responses to their postactivity reflection sheet; we did not collect data through the computational document,

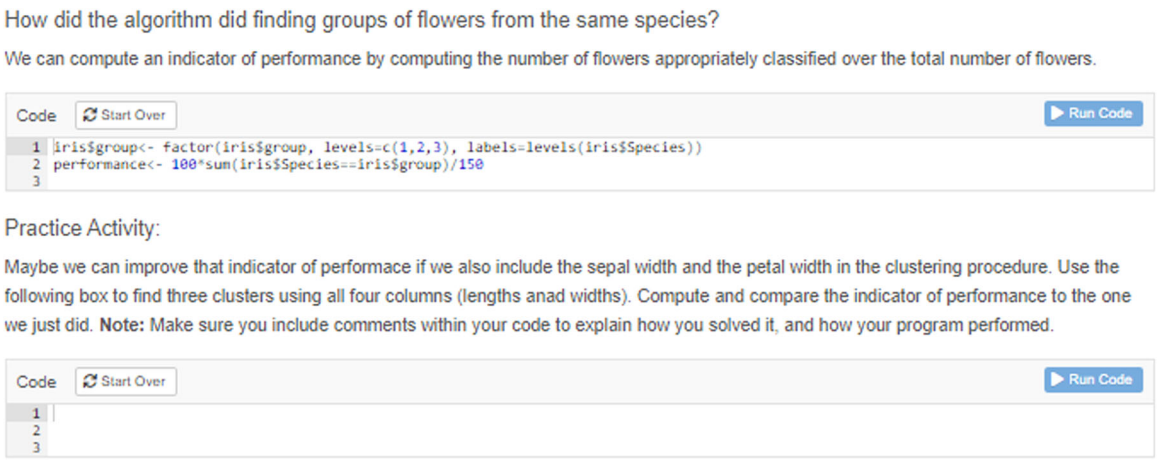


FIGURE 6 Sample of embedded prompts to elicit reflection

such as student code comments. Not all students submitted their responses to both reflection sheets. We collected a total of $n = 25$ responses across both semesters for analysis. This is a common number of participants for qualitative studies, which may range from a single individual to 20–30 participants [17]. Similarly, not all students answered all questions, altering the sample size per question. As data collection took place as part of the educational activities taking place, the study was approved as exempt by the Institutional Review Board.

6.5 | Data analysis method

Since this study inquiry focuses on students' conceptualization of their learning, we have selected a qualitative research approach that would allow capturing the nuances of student understanding rather than quantifying it. We used qualitative content analysis to characterize student responses to these five open-ended questions. The content analysis identifies a set of categories or codes in qualitative data (e.g., open-ended responses) and counts the number of instances in which each category

TABLE 2 Coding scheme for students' conceptualization of K-means

Category	Description and Sample Quote
Please describe what is the goal of K-means and how it works	
Creating groups as a goal	The creation of subgroups (clusters) was explicitly defined as a goal for the method. *To find subgroups of observations within a data set.*
Plotting as a goal	Identifies the creation of plot as a goal of the method rather than a support tool. *Its goal is to graph certain characteristics of the data*
Minimizing distance to centroids	Described the method in relation to the use of centroids. *To identify a certain number of centroids (k) and then allocate each and every data point to its nearest cluster.*
Iterative nature of the process	Recognizes that the algorithm iterate until a condition is met. *Repeats the process until there is little to no change between passes of the algorithm*
Deterministic nature of the process	Expects the algorithm to produce a "right answer" *To determine the definition of the right answer by finding clusters of data*
What are the factors that might affect the results from K-means?	
Input Selection	Initial Selection of Centroids Number of clusters (K) Distance calculation Number of Iterations
Data Quality	Presence of outliers Missing Data Data Spread Group overlapping
Are the clusters identified by K-means always meaningful? Why?	
Relationship between clusters and identified factors	Tied the mentioned factors affecting the results to the clusters output *It all comes down to how closely the clusters follow the factors affecting k-means* *Outliers and other variances could skew cluster limits*
Interpretation	Identified the critical need for interpretation of the results *it is up to the human being to assign meaning to the clusters*

Note: Shaded rows denote identified students' misconceptions.

occurs [21]. This approach allows us to identify how students conceptualize DS concepts and what misconceptions persist after working on the learning activity. Since there are few studies exploring these constructs, using qualitative approaches enabled the research team to identify emerging categories from the data. These categories may be used in the future for designing a quantitative instrument (e.g., a concept inventory).

The coding scheme emerged from the data through an iterative, open coding process for each question. Tables 2 and 3 present the coding scheme for the questions associated with student conceptualization of the method (RQ1) and the visualizations' affordances they identified (RQ2), respectively, each including a sample response for each category. Table 2 shows students' misconceptions highlighted in gray. The sample size of 25 respondents was appropriate for our qualitative approach. The analyzed responses allowed us to reach saturation, suggesting that we thoroughly described this specific case. The idea of

TABLE 3 Coding scheme for the affordances of the visualizations

Category	Description and sample quote
What do the explored visualizations tell you about the K-means algorithm and its results?	
Concrete representation	The visualizations provide a concrete representation of how the method works. *The plots helped show how the observations related to their assigned cluster/The explored visualizations give a helpful example as to how K-means work.*
Understanding the accuracy	The visualizations help to identify how accurate the method is given a specific data set *How well the cluster and the algorithm work and how closely the points are toward the centroid.*
Underlying assumptions	The visualizations help to identify the underlying assumptions for the algorithm *From the visualizations, I can see that the k-means algorithm heavily relies on the centroids chosen for each cluster.*
Strategic knowledge (Under what conditions it works)	The visualizations allow the students to understand under what conditions the algorithm works *The explored visualization tells me that a K-means result is more meaningful when the subsets are separate from each other*
Provides Examples of how they interacted with the visualizations	The student provides examples of how changing specific values or inputs in the tutorial would result in different outcomes *Changing the values of the seed will change the randomly generated centroid. It is necessary to keep the seed consistent if you do not want the centroids to change.*
How did the visualization help/hinder your understanding of the ML algorithm? How did it change with respect to your understanding based purely on the Data Science—Intro materials?	
Conceptual understanding of the method and the field	The visualizations contribute to a better understanding of the algorithm and data science *The visualization helped me a lot in understanding data science and helped broaden my understanding of the topic.*
Iterative nature of Data Science	The visualizations show how the methods iteratively refine their models up to certain accuracy measures. *These algorithms operate on a generation-by-generation basis. That is, these algorithms begin by completely guessing/randomly picking a start point. Their next step is to evaluate their performance, make improvements, and pick new points*
Consolidation through the tutorial activities	The students find useful, engaging activities such as annotating the code (self-explaining) or interacting with visualizations *I also feel like annotating the code enforced the concepts well. Whatever format and website was used for this ICA were awesome. The format was really helpful in understanding what was happening in the code*
Interpreting the accuracy	The visualizations help to interpret the accuracy *However, in terms of interpreting the accuracy of the results, it does seem to help with that as it is easy to see, when color-coded, which points are right and wrong.*
No significant change	The student did not experience a change in their understanding after interacting with the visualizations *It didn't really change much.*

saturation comes from the tradition of grounded theory but is now often used in several qualitative studies [17]. Saturation involves collecting data until the categories or themes start to repeat, and no additional categories emerge from additional data collection [9].

6.6 | Validity and reliability

The analysis was conducted by two of the research team members in two separate stages. During the first stage, only 20% of the data was used to develop our coding structure separately. While there is no agreement on the proportion of the data that needs to be coded by multiple researchers in qualitative research, a range between 10% and 25% is pretty common [43]. With such a first coding structure, the two researchers met to compare and discuss their categories; when we found discrepancies, we negotiated the modified codes as necessary until achieving complete consensus. For example, for the questions related to the affordances of the visualizations, we identified that the category *Strategic Knowledge (Under what conditions it works)* should only be used when students mention under what conditions the method may work better (e.g., *The explored visualization tells me that a K-means result is more meaningful when the subsets are separate from each other)*. Likewise, for the category *Iterative Nature of DS*, one of the researchers had missed some responses that discussed the iterations in the method. This stage contributed to enhancing the reliability of our results as we refined the coding scheme by clarifying codes and identifying keywords that would help us in the rest of the process. In the second stage, we used our agreed coding structure to complete the analysis of all the available data, which is what we report in the following section.

7 | RESULTS

This section presents the results of students' conceptualizations of K-means, followed by their perceived affordances of the visualizations for their learning of the ML method. A quantitative summary of our results is presented in Figures 7 and 8.

7.1 | Students' conceptualizations

Twenty students answered the three questions related to the conceptualization of K-means. These questions captured students' understanding based only on the recorded lecture that they watched previous to the class session. From the analysis of student responses to the question, *please describe the goal of K-means and how it works; 15 students referenced* the creation of groups as the goal of K-means. However, eight framed it within the actual number of groups to create (k). In contrast, seven stated it in more general terms (e.g., "is a way to categorize datasets into clusters and observations") or made a reference to unspecified characteristics to discover from the data (e.g., "the goal of K-means is to be able to take data and determine without the use of

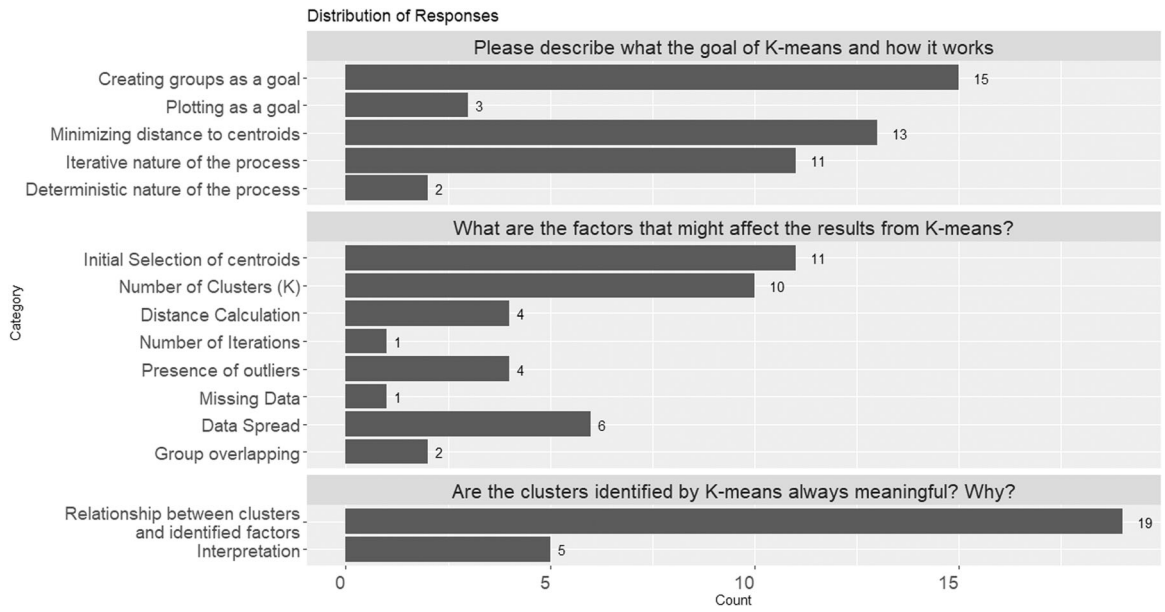


FIGURE 7 Distribution of codes for questions related to students' conceptualizations of the K-means method

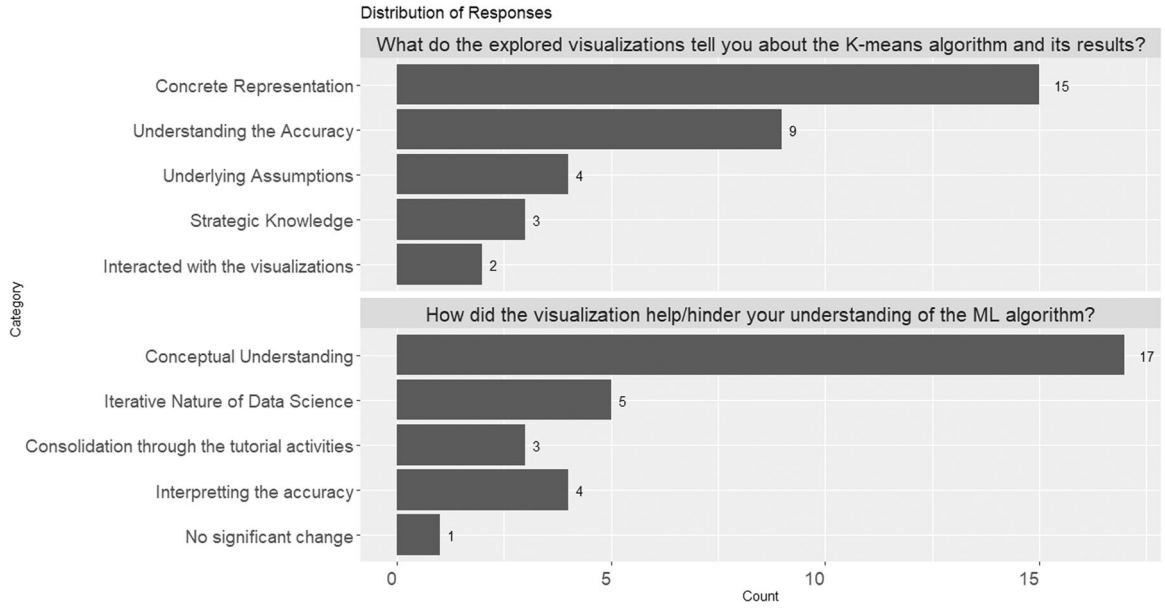


FIGURE 8 Distribution of codes for questions related to students' perceived affordances of the computational notebooks

labels if there are groups of data and what those groups mean").

Minimizing the distance to centroids was mentioned as an essential part of the algorithm. Thirteen students mentioned the use of centroids or means as a central element of the process (e.g., "it works by pairing data points to the closest centroid"). The iterative nature of the algorithm was cited by 11 students in our sample (e.g., "repeat the process until they [the datapoints] are not re-assigned anymore").

Three students cited K-means in the context of its larger category of an unsupervised ML algorithm and/or artificial intelligence algorithm, which was information provided in their online lecture. Some misconceptions about the goal and process of K-means were also evident in students' answers. Three students held the misconception that plotting was the goal of the algorithm. Two students also held a misconception of the deterministic nature of the algorithm, from which they expected to obtain a "right answer."

There were two larger categories in the responses to the question: *What are the factors that might affect the results from K-means?* The first category is related to the selection of the input for the algorithm; 11 students mentioned the initial number of centroids, highlighting their random selection "first shuffling the data set and randomly selecting K data points for the centroids...." Some students showcased an advanced understanding by clarifying that "not all initial 'random' placements will result in the same k-means outcome." Ten students also cited the number of clusters (k) as a factor affecting the result. Only four students mentioned the selection of the distance calculation method as a relevant factor. Finally, one student mentioned the number of iterations as a factor affecting the results of the algorithm, which did not tie the stopping of the algorithm to certain similitude criteria, therefore considered a misconception. The second category of factors was those related to data quality: students mentioned having ill-defined, too spread, or missing data (six students) and the presence of outliers (four students) as factors affecting the results of the algorithm. Two students cited group overlapping as an influencing factor, which might denote a misconception of the algorithm since its resulting groups cannot overlap.

Finally, when analyzing the responses to the question: *Are the clusters identified by K-means always meaningful? Why?* The large majority of students (19) were explicitly answering that the results of the K-means algorithm were not always meaningful. The reasons provided for this varied, but they mostly connected their responses to the factors listed in the previous question. Therefore, these are summarized as a class in its corresponding section of Table 2. Only five students also identified interpretation to be a critical element of the meaningfulness of the resulting clusters. Students who showcased misconceptions related to plotting as a goal and a deterministic nature of the process in the first question showcased answers still aligning with such misconceptions. For example, a student who declared that graphing was the main goal of the method stated: "No, because they are grouped based on distance from a certain point. A graph might not always indicate a relationship, hence the issue with interpretability."

7.2 | Students' perceived affordances

We received responses from 19 students for the question: *what do the explored visualizations tell you about the K-means algorithm and its results?* More than one category from the ones described in the coding scheme (Table 3) emerged in each response. The most common affordance described by the participating students is that the visualizations provided a Concrete Representation of the method (15 responses), which included answers such as *"*The explored visualizations give a helpful example as to how K-means work*"* or *"*The explored visualizations tell you whether the K-means algorithm worked or not. The visualizations will have groups labeled or colored differently, so it is easy to identify which clusters were found.*"* Nine students mentioned that the visualizations allowed them to identify how accurate the method is given a specific data set, including statements such as: "I think it is very useful when trying to group data, but you have to be careful not to rely entirely on it because it is not always 100% accurate. With the iris problem, adding more variables made it less accurate than with fewer."

Four students found the visualizations helpful in understanding the underlying assumptions of the methods (e.g., "From the visualizations, I can see that the k-means algorithm heavily relies on the centroids chosen for each cluster.*"*), three students talked about giving them the strategic knowledge to understand the conditions under which the method works (e.g., *"*There are flaws within the K-means algorithm, especially when the groups you are trying to define are closely related, but it is a useful tool at the base level.*"*), and two students provided concrete examples of how they interacted with the visualizations.

Five categories emerged from student responses to the question: *How did the visualization help/hinder your understanding of the ML algorithm? How did it change with respect to your understanding based purely on the Data Science—Intro materials?* Most students (17) discussed how the visualizations helped them to have a better understanding of the algorithm and data science, with responses such as:

> The visualization helps me understand the ML algorithm better because I can physically see the clusters and where the centroids lie. It is more difficult to just look at different numbers to compare the amount or size of the clusters. It changed with respect to my understanding of the Data Science-Intro materials by actually seeing a process of how machine learning is used and can be interpreted.

Five students found that one of the main affordances of these visualizations was to understand the *Iterative Nature of Data Science*, something that is not as easy to understand with static examples on a whiteboard. As one of the students stated: "I think it helped me understand it better. I'm more visual so seeing the clusters change with the centroids helped me understand it a bit better and gain respect for Data Science because the repetition really does help.*"*

Four students also mentioned in this question that the visualizations help to interpret the accuracy of the method: "I was able to see that it was slightly flawed, but it would get better with more tests." Three students explicitly praised the computational documents, with responses such as:

> "My understanding of machine learning and data science intro materials has definitely been helped thanks to these visualizations. Whatever format and website that was used for this ICA was awesome. The format was really helpful in understanding what was happening in the code."

Finally, one of the students felt that it did not provide any particular help, and their understanding of DS did not change.

8 | DISCUSSION AND IMPLICATIONS

This article used a DBR approach to explore how computational notebooks as a way to deliver a computational cognitive apprenticeship promote knowledge integration of DS concepts. We explored the following research questions: (1) How do students conceptualize their understanding of an unsupervised ML method after engaging with interactive visualizations designed using the computational cognitive apprenticeship approach? And (2) How do the affordances of interactive visualizations support or hinder student knowledge integration of an unsupervised ML method? DBR studies may start from a learning need (e.g., teaching DS concepts to engineering undergraduate students) and use both existing literature and empirical evidence to design, implement, assess, and refine instructional implementations, providing both theoretical and local contributions. Specifically, this study offers theoretical contributions by describing how the computational cognitive apprenticeship model supports knowledge integration and what students' misconceptions about DS persist after completing the

learning activities. This study also provides local contributions related to the learning materials (e.g., the computational notebooks and the reflection sheets) informed by existing learning theories and local empirical data.

The first research question allowed us to understand student conceptualizations of K-means and the misconceptions they showed after watching a video lecture about this topic and interacting with the computational notebooks. Our results showed that most students developed a basic understanding of the goal and nature of the K-means algorithm, correctly identifying the goal of K-means to be the creation of groups, the minimization of distance to centroids, and the iterative nature of the process. Similarly, multiple factors affecting the execution of K-means were correctly identified by students, including (1) inputs, like the selection of centroids, the number of clusters, and the distance calculation, and (2) data quality, like the presence of outliers, missing data, and data spread. However, we also identified some important misconceptions students held after the video lecture. In particular, some students identified plotting as the goal of the method, while others had an expectation for the method to be deterministic in nature; therefore, it would provide a "right answer." The inability of students to identify that the plot is only a representation of the results rather than the result itself and that the algorithm can provide multiple answers can be argued to be a limiting factor in understanding the relationship between algorithm inputs and algorithm outputs.

In general, misconceptions about ML algorithms have been poorly explored [54, 69]. In their research about faculty perceptions of barriers to student learning of ML, Sulmont and colleagues [54] found that faculty teaching ML to students with backgrounds outside of CS and statistics deemed that students "were generally unaffected by misconceptions, but rather a lack of conception to begin with" (p. 948). For the engineering students in our sample, the misconceptions were identified after they were exposed to the video lecture on the topic and interacting with the computational notebooks, which resemble traditional teaching methods that most students experience. Furthermore, when verifying the affordances of the computational documents identified by students that started with misconceptions, we found that they recognized the value of having a concrete representation of how the method works and later showcased elements of conceptual understanding. Therefore, beyond identifying previously unknown misconceptions while learning ML algorithms, we are also advancing the consideration of potential solutions to address them.

The second research question showed the affordances of interactive visualizations to promote knowledge integration. We argue that the pedagogical framework, computational cognitive apprenticeship, helped students with knowledge integration by incorporating and synthesizing multiple representations into a common body of existing knowledge [26, 53]. Specifically, our learning design helped students (according to the listing presented in Section III) to [5, 44]: (a) eliciting ideas by connecting the concepts of video lectures into an interactive computational notebook; (b) adding new ideas and distinguishing ideas, by providing meaning-making mechanisms such as the step-by-step visualizations of the algorithm, that enabled learners to make connections between what they already know and how the method actually worked; and (d) sorting out ideas, by providing opportunities to learners to refine their knowledge through analysis and reflection. The participating students explicitly stated how the computational notebooks helped them identify how the algorithm works iteratively, how the centroids change when to stop, and how accurate the method is. This is a promising result, so we prepare students to recognize that the data sets we use in ML algorithms may include bias that affects the output and that the user is responsible for evaluating and interpreting the outputs [51, 67]. This approach also helped them understand ML algorithms in general and how they work iteratively.

We will further emphasize the reflection and exploration aspects of the computational cognitive apprenticeship for future educational implementations. Specifically, we will have students execute the algorithm through a larger number of cases to provide opportunities to identify the nuanced differences in the output. Along with that, we will implement reflection and discussion processes that will further challenge the identified students' misconceptions. For instance, we may use different data sets, including a specific bias (e.g., only representing some part of the population), so students reflect on the "right answer."

The implications of this study for education research include the identified potential of theoretically based pedagogies for the creation of computational notebooks. To the best of our knowledge, this is one of the initial efforts to use learning theories for the pedagogical design of computational notebooks to (a) scaffold learning processes and (b) evaluate their impact on students' learning. Similar theory-based approaches could be compared using other learning theories and pedagogical strategies in the near future. Furthermore, the pedagogical design and learning principles presented in this study can be used to support the learning of other DS and ML

topics. In addition, our identification of student misconceptions in DS may lead to the development of valid and reliable assessment instruments (e.g., concept inventories), important tools for education research. Moreover, the use of computational notebooks in research can be leveraged for the identification of students' learning shortcomings for timely attention to their mastery of other DS topics and other knowledge areas. Student interactions with the computational documents may be collected as learning analytics to understand how students are learning the computing skills in DS.

Our research also has positive implications for teaching and learning, as these and future results can provide targeted strategies for the design and implementation of computational notebooks by more faculty in and out of DS. Understanding persisting student misconceptions in DS may support the making of informed instructional decisions. A focus on the components of modeling, coaching, scaffolding, and articulation of the computational cognitive apprenticeship framework provides useful guidelines for instructional designers.

9 | CONCLUSION, LIMITATIONS, AND FUTURE WORK

We have presented the results on the potential of using computational notebooks to learn an unsupervised ML algorithm. The computational documents were developed using the computational cognitive apprenticeship framework to facilitate knowledge integration to learn K-means. Our results on students' conceptualization of their understanding of the unsupervised ML method of K-means showed that after watching a video lecture on the topic, students had mostly accurate (a) conceptions of the goals of the methods, (b) the factors affecting its results, and (c) the challenges of getting meaningful results. However, we also identified important misconceptions that students held after the introductory lesson. These misconceptions were mostly related to the goal and nature of the algorithm. Our work is the first to identify such misconceptions, therefore contributing to the literature on ML education. One limitation of such a contribution is that it can be questioned if such misconceptions are directly linked to the understanding of the ML algorithm or can be linked instead to the interaction with the computational notebooks. For example, consider plotting as the goal of the algorithm because plots are prevalent in computational notebooks. The identification of such misconceptions is the first step toward their scrutiny enabling formal questioning of their source and, therefore, their potential eradication.

In addition, when exploring the affordances of interactive visualizations on student knowledge integration of the K-means method, we found that it positively supported students' elicitation of ideas and the addition of new ideas into their conceptions of the method. Furthermore, we found that it supported students with the identified misconceptions to overcome them. Our results encourage future work on more specific inquiries about misconceptions when learning ML algorithms in particular and DS methods in general. A limitation of this study is the use of only one ML topic. Future work will include the implementation of this approach in the teaching of a larger diversity of ML topics beyond K-means.

Although the sample size was adequate for this qualitative study, future work will include the evaluation of the effectiveness of this design strategy for computational notebooks on student learning in larger scale quantitative studies. Such evidence would derive from practical suggestions to educators in the creation of effective computational notebooks. Experimental approaches using control conditions and nontheoretical approaches for the design of computational notebooks are also an important space for future work.

This study exclusively focused on engineering students as the target population, who have a higher quantitative and programming knowledge than students outside of STEM, and less programming expertise than students in computer science. Future studies need to include non-STEM students and CS students to evaluate the potential of the integration of the computational cognitive apprenticeship framework for knowledge integration of DS.

The timing and arrangement of the intervention activities might also be improved. For example, the ability to capture real-time responses to the reflection prompts within the computational notebooks would provide a more nuanced picture of students' learning than the one achieved by separate reflection sheets like those used in this study. Still, the presented evidence is encouraging the potential for the proposed design approach using the computational cognitive apprenticeship pedagogical framework to reach knowledge integration through the use of computational notebooks. Future endeavors will aim to explore the scalability of the strategy as well as the teaching of other unsupervised ML algorithms and other supervised ML algorithms, which would help develop a larger body of evidence propelling best practices for teaching DS topics in the future.

ACKNOWLEDGMENT

This study is in part based upon efforts supported by the EMBRIO Institute, contract #2120200, a National Science Foundation (NSF) Biology Integration Institute, and by National Science Foundation under Award #2134667. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of NSF or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

DATA AVAILABILITY STATEMENT

Data will be available upon request to the corresponding author.

ORCID

Matilde Sánchez-Peña http://orcid.org/0000-0002-3778-3219
Camilo Vieira http://orcid.org/0000-0001-8720-0002
Alejandra J. Magana http://orcid.org/0000-0001-6117-7502

REFERENCES

1. J. L. Ambite, L. Fierro, J. Gordon, G.A. Burns, F. Geigl, K. Lerman, and J.D. Van Horn, *BD2K training coordinating center's ERuDIte: The educational resource discovery index for data science*, IEEE Trans. Emerg. Topics Comput. **9** (2021), no. 1, 316–328. https://doi.org/10.1109/TETC.2019.2903466
2. S. Barab and K. Squire, *Design-Based research: Putting a stake in the ground*, J. Learn. Sci. **13** (2004), no. 1, 1–14. https://doi.org/10.1207/s15327809jls1301_1
3. F. Berman, R. Rutenbar, H. Christensen, S. Davidson, D. Estrin, M. Franklin, B. Hailpern, M. Martonosi, P. Raghavan, V. Stodden, and A. Szalay, *Realizing the potential of data science*, Communications of the ACM. **61** (2018), no. 4, 67–72.
4. T. Bolukbasi, K.-W. Chang, J. Y. Zou, V. Saligrama, and A. T. Kalai, *Man is to computer programmer as woman is to homemaker? debiasing word embeddings*, 30th Conference on Neural Information Processing Systems (2016), https://proceedings.neurips.cc/paper/2016/hash/a486cd07e4ac3d270571622f4f316ec5-Abstract.html
5. F. Boukouvala, A. Dowling, J. Verrett, Z. Ulissi, and V. Zavala, *Computational notebooks in chemical engineering curricula*, Chem. Eng. Educ. **54**. no. 3, 143–150.
6. J. S. Brown, A. Collins, and P. Duguid, *Situated cognition and the culture of learning*, Educ. Res. **18** (1989), no. 1, 32–42. https://doi.org/10.3102/0013189X018001032
7. A. Cardoso, J. Leitão, and C. Teixeira, *Using the Jupyter notebook as a tool to support the teaching and learning processes in engineering courses,* in Challen. Dig. Transform. Educ. ICL 2018. Adv. Int. Sys. & Comp. **917**, 227–236. https://doi.org/10.1007/978-3-030-11935-5_22
8. A. Chadha, and S. Kumar, *An improved K-Means clustering algorithm: A step forward for removal of dependency on K*, in International Conference on Reliability Optimization and Information Technology (ICROIT), (2014), pp. 136–140. https://doi.org/10.1109/ICROIT.2014.6798312
9. K. Charmaz, *Constructing grounded theory: A practical guide through qualitative analysis*, 1st edition. SAGE Publications Ltd, London, 2006.
10. S. Chattopadhyay, I. Prasad, A. Z. Henley, A. Sarma, and T. Barik, *What's wrong with computational notebooks? Pain points, needs, and design opportunities*, CHI Conference on Human Factors in Computing Systems (2020), pp. 1–12. https://doi.org/10.1145/3313831.3376729
11. L. Chen, P. Chen, and Z. Lin, *Artificial intelligence in education: A review*, IEEE Access. **8** (2020), 75264–75278. https://doi.org/10.1109/ACCESS.2020.2988510
12. J. L. Chiu and M. C. Linn, *Knowledge integration and wise engineering*, *J. Pre-Coll. Eng. Educ. Res*. **1** (2011), no. 1, 1–14. https://doi.org/10.7771/2157-9288.1026
13. A. Collins, J. S. Brown, and A. Holum, *Cognitive apprenticeship: Making thinking visible*, Am. Educ. **6**. 38–46.
14. A. Collins, J. S. Brown, and S. E. Newman, *Cognitive apprenticeship: Teaching the craft of Reading, writing and mathematics*, Thinking: J. Philos. Child. **8** (1988), no. 1, 2–10. https://doi.org/10.5840/thinking19888129
15. A. Collins, J. S. Brown, and S. E. Newman, *Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics, in Knowing, learning, and instruction: Essays in honor of Robert Glaser*, Routledge, New York, 1989, pp. 453–494.
16. A. Collins and M. Kapur, *Cognitive apprenticeship*, The Cambridge Handbook of the Learning Sciences (R. K. Sawyer, ed.), 2nd ed., Cambridge University Press, Cambridge, 2014, pp. 109–127. https://doi.org/10.1017/CBO9781139519526.008
17. J. W. Creswell, *Research design: Qualitative quantitative and mixed methods approaches*, 4th edition., SAGE Publications, Inc, Thousand Oaks, 2014.
18. A. Danyluk, and P. Leidig, *Computing competencies for undergraduate data science curricula*, Association for Computing Machinery, New York, 2021. https://dl.acm.org/citation.cfm?id=3453538
19. K. C. Davis. *Ethics in data science education*, ASEE Virtual Annual Conference (2020). https://peer.asee.org/ethics-in-data-science-education
20. J. C. Domínguez, M. V. Alonso, E. J. González, M. I. Guijarro, R. Miranda, M. Oliet, V. Rigual, J. M. Toledo, M. M. Villar-Chavero, and P. Yustos, *Teaching chemical engineering using jupyter notebook: Problem generators and lecturing tools*, Educ. Chem. Eng. **37** (2021), pp. 1–10. https://doi.org/10.1016/j.ece.2021.06.004
21. S. Elo, M. Kääriäinen, O. Kanste, T. Pölkki, K. Utriainen, and H. Kyngäs, *Qualitative content analysis: A focus on trustworthiness*, SAGE Open. **4** (2014), no. 1, 215824401452263. https://doi.org/10.1177/2158244014522633
22. F. Emmert-Streib and M. Dehmer, *Defining data science by a data-driven quantification of the community*, Mach. Learn. Knowl. Extr. **1** (2018), 235–251. https://doi.org/10.3390/make1010015

23. J. Engel, *Statistical literacy for active citizenship: A call for data science education*, Stat. Educ. Res. J. **16** (2021), no. 1, 44–49. https://doi.org/10.52041/serj.v16i1.213
24. C. A. Engelmann and J. E. Huntoon, *Improving student learning by addressing misconceptions*, Eos, Trans. Am. Geophys. Union. **92** (2011), no. 50, 465–466. https://doi.org/10.1029/2011EO500001
25. J. Estevez, G. Garate, and M. Graña, *Gentle introduction to artificial intelligence for high-school students using scratch*, IEEE Access. **7** (2019), 179027–179036. https://doi.org/10.1109/ACCESS.2019.2956136
26. H. Fennell, J. Lyon, A. Madamanchi, and A. Magana, *Toward computational apprenticeship: bringing a constructivist agenda to computational pedagogy*, J. Eng. Educ. **109** (2020), no. 4, 1–7. https://doi.org/10.1002/jee.20316
27. B. E. Granger and F. Pérez, *Jupyter: Thinking and storytelling with code and data*, Comput. Sci. Eng. **23** (2021), no. 2, 7–14. https://doi.org/10.1109/MCSE.2021.3059263
28. V. Gudivada, A. Apon, and J. Ding, *Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations*, Int. J. Adv. Softw. **10**. no. 1, 1–20.
29. A. Head, F. Hohman, T. Barik, S. M. Drucker, and R. DeLine, *Managing messes in computational notebooks*, CHI Conference on Human Factors in Computing Systems (2019), pp. 1–12. https://doi.org/10.1145/3290605.3300500
30. B. M. Hill, D. Dailey, R. T. Guy, B. Lewis, M. Matsuzaki, and Jonathan T. Morgan, *Democratizing data science: The community data science workshops and classes*, Big Data Factories: Collaborative Approaches, Computational Social Sciences (S. Adam Matei, N. Jullien, and S. P. Goggins, eds.), Springer International Publishing, New York, pp. 115–135.
31. P. Ihantola, T. Ahoniemi, V. Karavirta, and O. Seppälä, *Review of recent systems for automatic assessment of programming assignments*, 10th Koli Calling International Conference on Computing Education Research (2010), 86–93. https://doi.org/10.1145/1930464.1930480
32. R. A. Irizarry, *The role of academia in data science education*, Harv. Data Sci. Rev. **2** (2020), no. 1. https://doi.org/10.1162/99608f92.dd363929
33. A. Jaiswal, A. J. Magana, J. Lyon, E. Gundlack, and M. D. Ward, *Student experiences within a data science learning community: A communities of Practice perspective*, Learn. Commun. Res. Pract. **9** no. 1. https://files.eric.ed.gov/fulltext/EJ1320458.pdf
34. S. Kross, R. D. Peng, B. S. Caffo, I. Gooding, and J. T. Leek, *The democratization of data science education*, Am. Stat. **74** (2020), no. 1, 1–7. https://doi.org/10.1080/00031305.2019.1668849
35. M. C. Linn, *Designing computer learning environments for engineering and computer science: The scaffolded knowledge integration framework*, J. Sci. Educ. Technol. **4** (1995), no. 2, 103–126. https://doi.org/10.1007/BF02214052
36. M. C. Linn, *Designing the knowledge integration environment*, Int. J. Sci. Educ. **22** (2010), no. 8, 781–796. https://doi.org/10.1080/095006900412275
37. M. C. Linn, P. Bell, and E. A. Davis, *3 specific design principles: Elaborating the scaffolded knowledge integration framework*, in Internet Environ. Sci. Educ., Routledge, 2004.
38. A. Magana and G. Coutinho, *Modeling and simulation practices for a computational thinking-enabled engineering workforce*, Comput. Appl. Eng. Educ. **25** (2017), 62–78. https://doi.org/10.1002/cae.21779
39. A. J. Magana, M. L. Falk, and M. J. Reese, *Introducing discipline-based computing in undergraduate engineering education*, ACM Trans. Comput. Educ. **13** (2013), no. 4, 16:1–16:22. https://doi.org/10.1145/2534971
40. A. J. Magana, M. L. Falk, C. Vieira, and M. J. Reese, *A case study of undergraduate engineering students' computational literacy and self-beliefs about computing in the context of authentic practices*, Comput. Human. Behav. **61** (2016), 427–442. https://doi.org/10.1016/j.chb.2016.03.025
41. A. Magana, A. Jaiswal, A. Madamanchi, L. Parker, E. Gundlach, and M. Ward, *Characterizing the psychosocial effects of participating in a year-long residential research-oriented learning community*, Curr. Psychol. (2021). https://doi.org/10.1007/s12144-021-01612-y
42. L. S. Marques, C. Gresse von Wangenheim, and J. Hauck, *Teaching machine learning in school: A systematic mapping of the state of the art*, Inform. Educ. **19**. no. 2, 283–321.
43. C. O'Connor and H. Joffe, *Intercoder reliability in qualitative research: Debates and practical guidelines*, Int. J. Qual. Methods. **19** (2020), 160940691989922. https://doi.org/10.1177/1609406919899220
44. K. J. O'Hara, D. Blank, and J. Marshall, *Computational notebooks for AI education*, Twenty-Eight International Florida Artificial Intelligence Research Society Conference (2015). https://repository.brynmawr.edu/cgi/viewcontent.cgi?article=1030%26context=compsci_pubs
45. T. O. B. Odden and J. Burk, *Computational essays in the physics classroom*, Phys. Teach., **58** (2020), no. 4, 252–255. https://doi.org/10.1119/1.5145471
46. T. O. B. Odden, E. Lockwood, and M. D. Caballero, *Physics computational literacy: An exploratory case study using computational essays*, Phys. Rev. Phys. Educ. Res. **15** (2019), no. 2, 020152. https://doi.org/10.1103/PhysRevPhysEducRes.15.020152
47. J. D. Ortega-Alvarez, C. Vieira, N. Guarín-Zapata, and J. Gómez, *Flipping a computational modeling class: Strategies to engage students and foster active learning*, IEEE Frontiers in Education Conference, 2020. https://doi.org/10.1109/FIE44824.2020.9273890
48. J. F. Pimentel, L. Murta, V. Braganholo, and J. Freire, *A large-scale study about quality and reproducibility of jupyter notebooks*, IEEE/ACM 16th International Conference on Mining Software Repositories (MSR), 2019, pp. 507–517. https://doi.org/10.1109/MSR.2019.00077
49. B. M. Randles, I. V. Pasquetto, M. S. Golshan, and C. L. Borgman, *Using the jupyter notebook as a tool for open science: an empirical study*, ACM/IEEE Joint Conference on Digital Libraries (JCDL), 2017, pp. 1–2. https://doi.org/10.1109/JCDL.2017.7991618
50. A. C. Rule, *Design and use of computational notebooks*, ProQuest, University of California, San Diego, 2018. https://www.proquest.com/openview/286f9b19c5438c6d79c96bd203f43daa/1?pq-origsite=gscholar%26cbl=18750
51. R. Schutt, and C. O'Neil, *Doing data science: straight talk from the frontline*, O'Reilly Media, Incorporated, 2013.
52. H. Shen, *Interactive notebooks: Sharing the code*, Nature. **515** (2014 Nov 6), no. 7525, 151–152. https://doi.org/10.1038/515151a

53. A. Suárez, M. A. Alvarez-Feijoo, R. Fernández González, and E. Arce, *Teaching optimization of manufacturing problems via code components of a jupyter notebook*, Comput. Appl. Eng. Educ., **26** (2018), no. 5, 1102–1110. https://doi.org/10.1002/cae.21941
54. E. Sulmont, E. Patitsas, and J. R. Cooperstock, *Can you teach me to machine learn*? 50th ACM Technical Symposium on Computer Science Education (2019), pp. 948–954. https://doi.org/10.1145/3287324.3287392
55. E. Sulmont, E. Patitsas, and J. R. Cooperstock, *What is hard about teaching machine learning to non-majors? insights from classifying instructors learning goals*, ACM Trans. Comput. Educ. **19** no. 4, 33:1–33:16. https://doi.org/10.1145/3336124
56. H. Suresh and J. V. Guttag, *A framework for understanding sources of harm throughout the machine learning life cycle*, EAAMO: Equity and Access in Algorithms, Mechanisms and Optimization (2021). http://arxiv.org/abs/1901.10002
57. The Cognition and Technology Group at Vanderbilt, *Anchored instruction and its relationship to situated cognition*, Educ. Res. **19** (1990), no. 6, pp. 2–10. https://doi.org/10.3102/0013189X019006002
58. B. van der Vlist, R. van de Westelaken, C. Bartneck, J. Hu, R. Ahn, E. Barakova, F. Delbressine, and L. Feijs, *Teaching machine learning to design students*, Technol. E-Learn. Digit. Entertain. **5093**, 206–217. https://doi.org/10.1007/978-3-540-69736-7_23
59. C. Vieira, *cvieiram/learnRResources: Data science and statistics learn R resources*, Zenodo. https://doi.org/10.5281/zenodo.5752397
60. C. Vieira, A. J. Magana, and M. Aasakiran, *Computational methods for qualitative educational research: A hands-on workshop*, 48th Annual Conference of the European Society for Engineering Education (2020).
61. C. Vieira, A. J. Magana, M. L. Falk, and R. E. Garcia, *Writing in-code comments to self-explain in computational science and engineering education*, ACM Trans. Comput. Educ. **17** (2017), no. 4, 17:1–17:21. https://doi.org/10.1145/3058751
62. C. Vieira, A. J. Magana, A. Roy, and M. Falk, *Providing students with agency to self-scaffold in a computational science and engineering course*, J. Comput. High. Educ. **33** (2021), no. 2, 328–366. https://doi.org/10.1007/s12528-020-09267-7
63. C. Vieira, A. Magana, A. Roy, M. L. Falk, and Jr Reese, *Exploring undergraduate students' computational literacy in the context of problem solving*, Comput. Educ. J. **7**. 1, 100–112. https://coed.asee.org/wp-content/uploads/2020/08/11-Exploring-Undergraduate-Students-Computational-Literacy-in-the-Context-of-Problem-Solving.pdf
64. C. Vieira, A. Roy, A. J. Magana, M. Falk, and M. J. Reese, *In-code comments as a self-explanation strategy for computational science education*, ASEE Conference, 2016. https://doi.org/10.18260/p.25642
65. C. Vieira, Y. Y. Seah, and A. J. Magana, *Students' experimentation strategies in design: is process data enough?* Comput. Appl. Eng. Educ. **26** (2018), no. 5, 1903–1914. https://doi.org/10.1002/cae.22025
66. C. Vieira, J. Yan, and A. Magana, *Exploring design characteristics of worked examples to support programming and algorithm design*, J. Comput. Sci. Educ. **6** (2015), 2–15. https://doi.org/10.22369/issn.2153-4136/6/1/1
67. J. Wang, L. Li, and A. Zeller, *Better code, better sharing: On the need of analyzing jupyter notebooks*, ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results (2020), pp. 53–56. https://doi.org/10.1145/3377816.3381724
68. A. Y. Wang, A. Mittal, C. Brooks, and S. Oney, *How data scientists use computational notebooks for real-time collaboration*, Proc. ACM Hum.-Comput. Interact. **3** (2019), no. CSCW, 1–30. https://doi.org/10.1145/3359141
69. R. E. H. Wertz, K. R. B. Schmitt, L. Clark, B. Sandstede, and K. M. Kinnaird, *Crowdsourcing classroom observations to identify misconceptions in data science*, ASEE Virtual Annual Conference (2020). https://peer.asee.org/crowdsourcing-classroom-observations-to-identify-misconceptions-in-data-science
70. M. H. Wilkerson and J. L. Polman, *Situating data science: exploring how relationships to data shape learning*, J. Learn. Sci. **29** (2020), no. 1, 1–10. https://doi.org/10.1080/10508406.2019.1705664
71. C. H. Yu, S. Yu, A. Digangi, C. Jannasch-Pennell, and C. Kaprolet, *A data mining approach for identifying predictors of student retention from sophomore to junior year*, J. Data Sci. **8** (2010), 307–325.
72. A. Zarei, and R. Hutley, *Identifying the best admission criteria for data science using machine learning*. ASEE Annual Conference (2018). https://doi.org/10.18260/1-2–296
73. Y. Zhu and Y. Xiong, *Towards data science*, Data Sci. J. **14** (2015), no. 8, pp. 1–7. https://doi.org/10.5334/dsj-2015-008
74. A. Zúñiga-López and C. Avilés-Cruz, *Digital signal processing course on Jupyter–Python notebook for electronics undergraduates*, Comput. Appl. Eng. Educ. **28** (2020), no. 5, 1045–1057. https://doi.org/10.1002/cae.22277

AUTHOR BIOGRAPHIES

Matilde Sánchez-Peña is an assistant professor at the Department of Engineering Education at the University at Buffalo—The State University of New York, where she leads the Diversity Assessment Research in Engineering to Catalyze the Advancement of Respect and Equity (DAREtoCARE) Lab. Her research focuses on the development of cultures of care and well-being in engineering education spaces, assessing gains in institutional efforts to advance equity an inclusion, and the use of data science for training socially responsible engineers. She holds B.Eng. in Manufacturing Engineering from Universidad Autónoma de Nuevo León (México), MS in Industrial Engineering from the University of Puerto Rico at Mayaguez, MS in Statistics from The Ohio State University, and PhD in Engineering Education from Purdue University. She is the author of one book and more than 30 articles.

Camilo Vieira is an assistant professor in the Department of Education at Universidad del Norte (UniNorte; Barranquilla-Colombia). He holds a BSc in Systems Engineering and a Master's Degree in Engineering from Universidad Eafit (Colombia); and a PhD in Computational Sciences and Engineering from Purdue University (Indiana-US). Dr. Vieira completed a postdoctoral experience at Purdue University in information visualization, and he has been working at UniNorte since 2019, where he coordinates the research group Informática Educativa. In 2022, Dr. Vieira was a Fulbright Visiting Scholar at the University of Virginia. He investigates how to support student complex learning and how instructors teach complex topics, particularly in computing and engineering education. He also explores how to use computational methods to understand educational phenomena.

Alejandra J. Magana is the W.C. Furnas Professor in Enterprise Excellence in Computer and Information Technology and Professor in Engineering Education at Purdue University. She holds a BE in Information Systems and an MS in Technology, both from Tec de Monterrey; and an MS in Educational Technology and a PhD in Engineering Education, both from Purdue University. Her research program investigates how model-based cognition in Science, Technology, Engineering, and Mathematics (STEM) can be better supported by means of expert tools and disciplinary practices such as data science, computation, modeling, and simulation. In 2015 Dr. Magana received the National Science Foundation's Faculty Early Career Development (CAREER) Award to investigate modeling and simulation practices in undergraduate engineering education. In 2016 she was conferred the status of Purdue Faculty Scholar for being on an accelerated path toward academic distinction. Dr. Magana serves as Deputy Editor for the *Journal of Engineering Education* and Associate Editor for the *Computer Applications in Engineering Education* journal.

How to cite this article: M. Sánchez-Peña, C. Vieira, and A. J. Magana, *Data science knowledge integration: Affordances of a computational cognitive apprenticeship on student conceptual understanding*, Comput. Appl. Eng. Educ. 2023;31:239–259. https://doi.org/10.1002/cae.22580