Received: 13 March 2021 | Revised: 31 October 2022 | Accepted: 5 June 2023

DOI: 10.1002/tea.21888

RESEARCH ARTICLE

JRST | WILEY

Designing a framework for teachers' integration of computational thinking into elementary science

Lautaro Cabrera[1] | Diane Jass Ketelhut[1] | Kelly Mills[2] | Heather Killen[1] | Merijke Coenraad[2] | Virginia L. Byrne[3] | Jandelyn Dawn Plane[4]

[1]Department Teaching and Learning, Policy and Leadership, University of Maryland, College Park, College Park, Maryland, USA

[2]Learning Experience Design, Digital Promise, Washington, DC, USA

[3]Department of Advanced Studies and Leadership, School of Education and Urban Studies, Baltimore, Maryland, USA

[4]Department of Computer Science, University of Maryland, College Park, College Park, Maryland, USA

Correspondence
Lautaro Cabrera, Department Teaching and Learning, Policy and Leadership, University of Maryland, College Park, College Park, MD, USA.
Email: cabrera1@terpmail.umd.edu

Funding information
National Science Foundation, Grant/Award Number: 1639891

Abstract

As professional science becomes increasingly computational, researchers and educators are advocating for the integration of computational thinking (CT) into science education. Researchers and policymakers have argued that CT learning opportunities should begin in elementary school and span across the K-12 grades. While researchers and policymakers have specified how students should engage in CT for science learning, the success of CT integration ultimately depends on how elementary teachers implement CT in their science lessons. This new demand for teachers who can integrate CT has created a need for effective conceptual tools that teacher educators and professional development designers can use to develop elementary teachers' understanding and operationalization of CT for their classrooms. However, existing frameworks for CT integration have limitations. Existing frameworks either overlook the elementary grades, conceptualize CT in isolation and not integrated into science, and/or have not been tested in teacher education contexts. After reviewing existing CT integration frameworks and detailing an important gap in the science teacher education literature, we present our framework for the integration of CT into elementary science education, with a special focus on how to use this framework with teachers. Situated within our design-based research study, we (a) explain the decision-making process of designing the framework; (b) describe the pedagogical affordances and challenges it provided as we implemented it with a cohort of pre- and in-service teachers; (c) provide suggestions for its use in teacher education contexts; and (d) theorize possible pathways to continue its refinement.

KEYWORDS

computational thinking, elementary, professional development, science education, teacher education

1 | INTRODUCTION

As computational thinking (CT) continues to become an integral part of scientific investigations (Denning, 2017), researchers, educators, and policymakers are advocating for integrating CT into public K-12 education—particularly into science education (National Science and Technology Council [NSTC], 2018). The focus on science as a fertile ground for CT integration is associated with a proliferation of computational branches of science and a recognition that computers are increasingly a part of the practice of science. And, while these initiatives to integrate CT into science focus on preparing *students* to engage in CT, the preparation of *teachers* is instrumental to successfully provide CT learning opportunities for all science students (Bybee & National Science Teachers Association [NSTA], 2010).

Efforts to prepare teachers to integrate CT have illuminated some effective strategies and some limitations of existing approaches. For example, CT modules in pre-service education and professional development (PD) programs are typically successful at increasing teachers' confidence in their ability to integrate CT, communicating a basic understanding of CT definitions, and encouraging teachers to attempt CT integration in their classrooms (Ketelhut et al., 2019; Mouza et al., 2018; Yadav et al., 2011, 2018). However, the diversity of CT definitions available and the lack of clarity on its pedagogical applications can create confusion and promote diverging operationalizations (Bower & Lister, 2015; Chang & Peterson, 2018; Rich et al., 2020). Teacher educators and PD designers are especially affected by this ambiguity as they have to adapt conceptual tools designed for various grade levels and disciplines to the particular contexts of their teachers (e.g., Hestness et al., 2018).

For example, teacher educators and PD designers who aim to prepare teachers to integrate CT into elementary science instruction are forced to combine a diverse set of guiding resources. Some conceptual tools describe the nature of CT (e.g., Barr & Stephenson, 2011; Computer Science Teachers Association & International Society for Technology in Education, 2011), others define CT integration for middle and high-school levels (e.g., Sengupta et al., 2013; Weintrop et al., 2016), others promote general scientific practices for elementary school (National Science and Technology Council [NSTC], 2018; NGSS Lead States, 2013), and yet others address issues of teacher learning and pedagogical content knowledge in general (e.g., Clarke & Hollingsworth, 2002; Hammerness et al., 2012; Mishra & Koehler, 2006). In this article, we aim to address this need for teacher educators and PD designers.

After establishing the importance of CT integration into elementary science and the literature gap for a conceptual tool to guide this work, we present our framework for the integration of CT into elementary science education, with a special focus on how to use this framework with teachers. Specifically, we (a) explain the decision-making process of designing the framework, (b) describe the pedagogical affordances and challenges it provided as we implemented it with a cohort of teachers, (c) provide suggestions for its use with pre- and in-service teachers, and (d) theorize possible pathways to continue its refinement.

2 | LITERATURE REVIEW AND MOTIVATION

To situate the contribution of this article in science education, we briefly review how CT is positioned within science learning and how opportunities to engage with CT practices within the elementary years may help address equity issues in science and computing participation. Then, we review literature examples of efforts to integrate CT into elementary science and to prepare teachers for that task. We finish this section with a review of existing frameworks that can guide the preparation of teachers to integrate CT into science instruction and the framework gap that motivated this work.

3 | CT FOR SCIENCE LEARNING

CT has multiple and contested definitions (Tedre & Denning, 2016). A review by Grover and Pea (2013) identified consensus in the literature around the main elements of CT such as computational abstractions, systematic processing of information, problem decomposition, iteration, conditional logic, and debugging (p. 39). Wing (2006), who reintroduced the term into educational debates in 2006, advocated for CT as a skill for "everyone, not just computer scientists" (p. 33) because the practices associated with CT could help students solve problems in all disciplines and even their daily lives. And, while some researchers have studied how to integrate CT into compulsory education broadly (Voogt et al., 2015), others have focused on the specific value of engaging students in CT in service of disciplinary learning. Integrating CT into an existing discipline can avoid the challenge of adding a new obligation to an already-packed curriculum where instructional time is limited and contested (Lee et al., 2020).

One discipline that researchers have identified as especially synergistic with CT is science education—as reinforced centrality of computational practices within professional science (e.g., computational biology, computational astronomy, computational neuroscience; Denning, 2017). Researchers and educators have advocated for computational methods and CT practices to be included in science education to prepare students to engage in modern science (Lee et al., 2020).

In other words, the main components of CT are in line with a modern view of science education focused on scientific and engineering *practices*, where we *do* science. This is illustrated by the inclusion of CT as part of NGSS key science and engineering practices (NGSS Lead

States, 2013). Stakeholders promoting the integration of CT into science see it as a valuable group of practices that are "at the core of scientific discovery and innovation in a world driven by technology" (Lee et al., 2020, p. 2).

With this goal, researchers have investigated how middle and high-school students can engage in CT and develop the skills that are necessary in today's professional sciences. For example, researchers have studied how students can develop an understanding of both physics and computing by using a "collaborative, computational STEM learning environment" (Hutchins et al., 2020, p. 83); learn about natural selection by designing algorithms that mimic the selection process (Peel et al., 2019); and use computational modeling tools to advance their understanding of food webs (Rachmatullah & Wiebe, 2021). But, while these studies largely focus on students who are *already enrolled* in science education opportunities, our work focuses on promoting these CT learning opportunities for students *before* middle-school—a fundamental and needed approach as we explain below.

3.1 | CT to expand participation in science and computing

In addition to positioning CT as a tool for science learning, we attend to research on how engagement with CT during elementary school years can foster a competency and confidence with both science and computing. It is well known that people of different races, ethnicities, and genders do not participate equally in science and computing careers (Code.org et al., 2021; Google Inc. & Gallup Inc., 2016). This trend in professional science is preceded by unequal interest in science in K-12 contexts (Anderson, 2017).

Research has indicated that engaging in CT learning has the potential to help elementary students feel like they *can* successfully participate in the computing and science opportunities that arise in middle- and high-school. Specifically, providing these opportunities during formative years may be crucial for the development of interest in science and a scientific identity, which researchers believe can serve as a predictor of participation in future science learning opportunities (Vincent-Ruz & Schunn, 2018). Because students begin to define their academic identities and interests in middle school by participating in elective courses and afterschool programs, intervening *before* they make these decisions can impact the choices they make from an academic menu. Focusing on CT in the service of science learning and to develop competency and confidence with computing in young students, we turn to a review of efforts to integrate CT into elementary science.

3.2 | CT integration in elementary science

One promising way of providing early CT opportunities in science education is by integrating CT practices into K-5 science curricula and instruction. Researchers and educators have investigated how elementary school students can engage with CT within science education. Across these initiatives, CT integration into science can be conceptualized as spanning three different levels (Waterman et al., 2019): Exist, where typical science education *already* engages students in some CT practices; Enhance, where instruction is enriched with activities that engage students in CT; and Extend, where CT is an integral part of science learning.

To clearly illustrate the kinds of integration that we aim to achieve in all science classrooms, we focus below on reviewing cases of integration at the Extend level. In these cases, CT is presented as an integral part of science education—mimicking the increasingly central role that computing plays in professional science. As the highest level of integration, initiatives at this level illustrate the kinds of opportunities best suited to both engage students in CT as a way to learn science and foster their competence and confidence in computing during formative years.

An exemplar of CT integration into elementary science at the highest level—Extend—is provided by Dickes et al. (2019). In this study, the researchers created a unit of fifteen 50-min lessons where students explored an ecosystem within an immersive virtual environment. Students also engaged with a 2D agent-based modeling environment where they use programming to control the behaviors of animals in the environment and see the outcomes in the ecosystem. The researchers created the unit and trained teachers to use these environments by providing PD opportunities. The authors demonstrate different moments of "transformative modeling" where students transform the disciplinary content from one type of representation to another. Overall, this extensive implementation resulted in students advancing their understanding of both the scientific concepts of the curriculum and the purpose and mechanisms of computational models.

Basu et al. (2015) provide another example of how CT can be integrated as a powerful way to learn science. Conducting their study in a "teacher-led, multi-domain classroom" of fifth graders, they demonstrate how students used a computational simulation and modeling environment to construct and test models on the topics of Kinematics and Ecology. The researchers, along with their partner teacher, provided science and CT resources to supplement students' engagement with the modeling environment. They also provided some "front-of-the-class and individual help" to support students in their model development. Pre-post assessments showed positive effects on students' learning of both scientific content and CT skills.

While these studies demonstrate that CT can be a productive tool for learning science, they also reveal that integrating CT at the Extend level requires significant expertise in science and CT; freedom and time to design extensive curricular units; and classroom support for implementation. Logically, a model where researchers dedicate extensive amounts of time to support a targeted teacher is not scalable and unfit to prepare *all* teachers to provide CT learning opportunities for *all* students. Instead, the widespread integration of CT depends on our ability to develop scalable strategies to prepare teachers to integrate CT into their science instruction.

3.3 | Preparing teachers to integrate CT in elementary science

Multiple studies have investigated how best to prepare teachers for the task of integrating CT into elementary science instruction. Some researchers have focused on modifying existing university courses to prepare cohorts of pre-service teachers to integrate CT. Others have created PD opportunities for in-service teachers. The types of learning opportunities provided in both types of interventions have been fairly consistent. For example, across pre- and in-service interventions, researchers have created activities that provide teachers with opportunities to learn about CT as a concept and how to integrate it into disciplines (Bower et al., 2017; Chang & Peterson, 2018; Curzon et al., 2014; Israel et al., 2015; Mouza et al., 2017; Yadav et al., 2014). Others have provided CT activities like programming robots (Gadanidis et al., 2017; Sadik et al., 2017), creating Scratch animations (Adler & Kim, 2018; Jaipal-Jamani & Angeli, 2017), and using data through Citizen Science activities (McGinnis et al., 2020).

Taken together, these studies have shown that interventions in teacher education can increase teachers' self-efficacy in integrating CT, their content knowledge about CT, and their ability to design CT-infused lessons. However, the educators in these studies often adopted different sources to define CT for their own contexts and, in doing so, exposed the difficulties that teacher educators have when selecting appropriate pedagogical and conceptual tools for instruction. The sources of CT definitions used to prepare teachers range from lists or taxonomies of practices that make up CT (e.g., Computer Science Teachers Association & International Society for Technology in Education, 2011; Weintrop et al., 2016) to theoretical frameworks that explain conceptual and pedagogical relationships between CT and science learning (e.g., Sengupta et al., 2013). To discuss these sources as a group of literature, we use the term *framework* to refer to them as it reflects the common goals that these articles have: to provide a definition of CT that aims its integration into education contexts.

The variance in CT frameworks used in teacher education interventions creates disparate experiences for teachers. For example, Mouza et al. (2017) leaned on the TPACK (Mishra & Koehler, 2006) framework to define the kinds of knowledge that their pre-service teachers would need to integrate CT effectively into their classrooms. On the other hand, Jaipal-Jamani and Angeli (2017) focused on the abstraction aspect of CT and how robotics could help develop abstracting skills. Logically, these two approaches resulted in different types of activities that promoted teachers' CT learning. Therefore, to demonstrate the variability and limitations that exist in the CT framework literature, we turn our attention to the most prominent sources that teacher educators have available to present and define CT for teachers.

3.4 | Existing CT frameworks and definitions

When tasked with preparing teachers to learn and teach CT, teacher educators can rely on a multitude of guiding documents and scholarship. As Table 1 summarizes, teacher educators contend with (a) diverse definitions and frameworks about CT at a general level for K-12 grades; (b) science-specific frameworks that guide the integration of CT practices at the 7–16 level grades; and (c) frameworks around teacher education and the types of knowledge that educators should master to integrate CT into their general instruction (Table 1). However, there is no current framework to guide the integration of CT into elementary science designed to be used with teachers.

TABLE 1 CT frameworks and their relationship to K-5, science, and teacher education.

Framework	Elementary	Science	Teacher Ed
Barr and Stephenson (2011)	Yes	No	No
Computer Science Teachers Association and International Society for Technology in Education (2011)	Yes	No	No
Sengupta et al. (2013)	No	Yes	No
Weintrop et al. (2016)	No	Yes	No
Malyn-Smith et al. (2018)	Yes	Yes	No
Angeli et al. (2016)	Yes	No	Yes
Our framework	Yes	Yes	Yes

While this gap in the literature may seem narrow, the lack of a teacher-oriented framework tailored for the integration of CT into science at the elementary level has important implications for teacher education and how CT is enacted in the classroom. In a review of studies focused on the integration of CT into elementary science and engineering instruction, we found that efforts on teacher education often introduce CT concepts by using general CT frameworks or sources aimed at higher grades (Ketelhut & Cabrera, 2020). These conceptual tools can lead elementary teachers to perceive CT as difficult or too complex for their students. These frameworks also do not give teachers developmentally appropriate models to guide their own CT integration, burdening them with the additional task of ideating new activities that engage in CT but are modified for younger students. Moreover, as we present below, our initial efforts to adapt existing frameworks to guide teachers in the integration of CT into elementary science showed significant limitations and convinced us of the need for a specialized conceptual tool.

Therefore, the goal of this article is to present our own conceptual tool to define CT that aims to counter the limitations listed above and provide initial evidence of its affordances and remaining challenges. While we present a list of CT practices that resembles a taxonomy as the core result of our work, we have added additional elements that can serve as a framework for its use in teacher education contexts. Specifically, we suggest future improvements on our own taxonomy based on implementation experiences and make recommendations for teacher educators aiming to use it as a guiding conceptual tool for elementary science teachers. Therefore, and to stay in line with our treatment of varied definitions of CT in the literature above, we refer to our work as the development of a *framework*.

To provide further justification for the need for the framework we present in this work, we turn to a review of the most prominent CT frameworks (Table 1), highlight their potential, and discuss the challenges of adapting them to be used in *teacher* education. Specifically, we review three categories of frameworks guiding the integration of CT: those aimed at CT integration (a) into K-12 contexts broadly, (b) into science as a discipline, and (c) into teacher education as a field of scholarship. In this section, we do not provide a summary of each framework. Instead, we describe the main components of each piece and examine its relationship to our research goal of preparing educators to integrate CT into science at the elementary level.

3.5 | Frameworks of CT for K-12 classroom integration

After an initial workshop exploring the pedagogical aspects of CT was published in 2010, it was clear that *consensus* was a difficult goal to attain (NRC, 2011). While multiple researchers had gathered to discuss and debate the multiple definitions and educational applications of CT, the report left a lack of clear direction for scholars to follow. With the goal of creating a more concise vision to guide CT integration into K-12 education, Barr and Stephenson (2011) wrote a piece outlining the essential components of CT, the necessary changes in K-12 school systems that could make this curriculum-wide integration possible, and even examples of CT integration for multiple disciplines and at various grade levels. As influential as this piece was to begin to define and operationalize CT in classrooms, the discussion around teacher education was limited to a call for preparation of teachers without specific guidelines on what kinds of skills educators should acquire or how teachers should be trained to champion CT integration.

Around the same time, two prominent organizations—the Computer Science Teachers Association and International Society for Technology in Education (2011), led by Stephenson, and the International Society for Technology in Education (ISTE)—released a "leadership kit" on CT to make the case for integrating CT into K-12 education and provide guidance to stakeholders looking to bring this new skill into classrooms. The report directly addressed issues of teacher education in its "implementation strategies guide" (p. 21) and provided multiple points of action that would support teachers to lead this integration. Among those recommendations, CSTA and ISTE specified that teachers should be provided with "resources that define CT and allow teachers to recognize where they already include it in their teaching" (p. 23). Conveniently, the report also provided an operational definition of CT for K-12 education and a "Progression Chart" where key components of CT are operationalized for different grade bands.

A key innovation in this definition of CT was its expansion from a set of skills to include a set of dispositions such as "persistence in working with difficult problems" and "confidence dealing with complexity" (p. 13). In terms of guiding the preparation of teachers for this integration of CT, this report provided useful conceptual tools, examples, and goals for teacher educators to build on. However, the adaptation of these resources to discipline-specific methods courses and PD programs remained less-detailed. This ambiguity was acutely problematic for science educators at the elementary level, where some examples and definitions of CT provided by the report were almost indistinguishable from traditional inquiry-based science instruction.

Therefore, teacher educators and PD designers looking to equip teachers to integrate CT into science could inform their practice with a plethora of general perspectives around CT but were presented with limited examples of CT-infused science instruction at different grade levels. This deficit made it difficult to provide teachers with a unified set of guidelines that could help them understand which instructional changes would meet the goal of CT integration and which changes would not suffice.

3.6 | Frameworks for CT in science and disciplinary content

As general CT frameworks were disseminated, educators turned their focus to the integration of CT into existing disciplines. After all, even the frameworks described above agreed that CT should be integrated into existing structures and aligned with current disciplinary standards. In this context, two exemplary frameworks to guide the integration of CT into science education were developed. First, Sengupta et al. (2013) created a framework based on the concept of abstraction and the productive role of agent-based modeling to develop abstracting skills. The authors argued that abstraction was a foundational skill in both CT and science, and therefore constituted an ideal pedagogical context for integrating the former into the latter.

Sengupta et al. (2013) also provided their own account of this integration in a middle-school science context, where students created programs within an agent-based modeling environment to explore curricular units of physics and biology. While the framework was well positioned to guide educators in their integration of CT into middle-school science, the authors recognize that the tasks described in their paper as key learning opportunities would be difficult to execute with younger children who may not be able to abstract at the same level than middle-schoolers. This framework serves as an important goalpost for elementary science education—delineating the need to equip students with the necessary prerequisites to successfully engage in abstraction and agent-based modeling in middle school.

A few years later, Weintrop et al. (2016) provided additional guidance toward the disciplinary integration of CT. The authors conducted a study to understand how professional scientists and researchers used CT in their work. The researchers used this investigation to propose a taxonomy of CT practices that could be integrated into science and mathematics classrooms at the high-school and college levels. This framework, which has been widely accepted and adopted in educational environments, broadened the perspective of CT to include other practices encompassing the entirety of the scientific process—not just abstraction.

Together, these discipline-specific frameworks provided teacher educators with clear goalposts of what students could encounter in middle school and beyond. However, important questions lingered such as: What are the prerequisites that students should meet to successfully engage in these CT practices after elementary school? And, what types of CT practices are appropriate for students at the elementary level?

A more recent framework by Malyn-smith et al. (2018) aimed to address this question by describing the CT skills that students should develop throughout their K-12 education from a disciplinary perspective. The authors argued that students should develop five CT skills (abstraction, algorithms, programming/development, data collection and analysis, and modeling and simulation) that could be integrated in five ways "consistent with [CT's] use in CT-integrated fields" (p. 184). This framework also included examples from classrooms at different grade-levels and within different disciplines, like math and science. While this document provided an important tool for conceptualizing a CT learning progression in K-12 education, the frameworks above did not address the key issue of preparing *teachers* to make this CT learning progression a reality in all science classrooms. To explore this matter, we turn to frameworks geared toward preparing teachers to integrate CT into their instruction.

3.7 | Frameworks for CT in teacher education

In 2016, an international team of researchers, led by Charoula Angeli, developed a CT K-6 curriculum framework that specified five elements of CT and described the types of knowledge teachers would need to develop to teach this curriculum (Angeli et al., 2016). The authors propose CT as a precursor skill to learning computer science and argue that acquiring the skills in their conceptual framework (abstraction, generalization, decomposition, algorithmic thinking, and debugging) would effectively prepare students for "more advanced theoretical and practical topics of computer science" in the future (p. 50).

This framework provided valuable guidance for science teacher educators at the elementary level by considering the needs of K-6 teachers and the types of knowledge that they will require to teach CT. These considerations were welcome additions to a CT body of literature that had largely focused on students. However, the generality of the framework limited its practicality for teachers in the conceptualization of CT as a tool to engage in *science* learning. For example, the authors describe the element of "Decomposition" as "the skill to break a complex problem into smaller parts that are easier to understand and solve" but that description leaves room for a very broad interpretation of how Decomposition could be applied for science learning. The examples the researchers provide for grade bands K-2, 3-4, and 5-6 are also nonspecific and make it hard for the teachers to envision how those practices fit within disciplinary and standards-based learning. For example, the authors suggest to "break a complex task into a series of simpler subtasks" but the responsibility of connecting how that process can advance content learning is left to the reader. In summary, the framework's specific focus on CT as a precursor of computer science omitted opportunities to include elements of CT that would both prepare students for computing in the future *and* allow them to engage with science (or other disciplinary) content while developing these skills.

As our framework was designed to counter the limitations found in the frameworks, we just reviewed, we use the rest of this article to detail the context of our study, the decision-making process behind our framework design, and provide an initial account of the pedagogical affordances and limitations it provided when implemented in our project. We also discuss remaining challenges and key considerations for its use in teacher education environments.

4 | CONTEXT AND METHODS

The *Integrating CT into Teacher Education Project* (ICTTE; pseudonym blinded for review) project is a design-based research (Brown, 1992; Collins, 1992; The Design-Based Research Collective, 2003) study at a large Mid-Atlantic research university that aims to understand how pre- and in-service teachers learn to integrate CT into their elementary science instruction. The study has two main intervention components: (1) a CT module within an existing science methods undergraduate course for pre-service teachers and (2) a PD experience where pre- and in-service teachers work together to develop CT-integrated science lessons for the elementary level. Each component was designed and implemented for the first time in the academic year of 2017–2018 (referred to as Year 1), redesigned in the summer of 2018, and reimplemented in the academic year of 2018–2019 (referred to as Year 2). In this article, we draw on findings from each design iteration of both components of the study to explain the design of our framework, provide initial evidence of how teachers used it to design CT-infused science lessons, and discuss its potential improvements and use considerations.

4.1 | Methods course CT module

The first component of our study is a CT module designed as a three-session intervention in an existing pre-service elementary science methods course. The goals of the module were to provide pre-service teachers with an understanding of CT, show them examples of integration into science, and give them an opportunity to practice designing their own integration. Instructors designed activities and provided readings for teachers to develop their understanding of CT and experience engaging with CT themselves. For example, teachers completed a series of programming challenges with Lego Mindstorms and mBot robots, created algorithms to order a series of organisms, and interacted with computational simulations about predators and prey populations. While the module took place during the first half of the course, students were expected to demonstrate their knowledge around CT at their capstone project. Specifically, students were required to create a full science lesson plan that integrated CT. The course was primarily taught by two members of the ICTTE research team and other researchers joined CT module classes as invited lecturers.

In the first year of our project, the instructors used a variety of resources to define CT, including some of the frameworks discussed above. In the second year, instructors used the framework we share in this article as the main resource to define CT and students were expected to integrate some practices in our framework into their final lesson plans.

4.2 | Inquiry into science teaching with CT group

The second component of our study is a PD series called the Science Teaching CT Inquiry Group (STIG^{CT}; Coenraad, Mills, et al., 2020; Coenraad, Plane, et al., 2020). The goal of the PD experience was to create a community of practice (Lave & Wenger, 1991) where teachers and researchers co-design ways to integrate CT into elementary science instruction (Killen, Coenraad, Byrne, Cabrera, & Ketelhut, 2020). Participants of the STIG^{CT} were pre-service teachers who were also students in the methods course and in-service teachers—some of them mentors to the participating pre-service teachers. The goals of the STIG^{CT} were to provide a more in-depth treatment of CT that focused on specific groups of practices, to be a forum for discussions of CT implementation in the classroom and curriculum integration, and to allow teachers to co-design CT-infused science lesson plans with peers and researchers. Each year we implemented the STIG^{CT}, the total duration of the PD was between 10 and 12 h and included activities like (a) providing direct instruction on CT practices definitions, (b) engaging the teachers in educational activities that illustrated those practices, (c) discussing how these practices could be integrated in their science teaching, (d) teachers and researchers co-designing CT-infused science lesson drafts, and (e) sharing final lesson plan designs and experiences executing them in the classroom.

For example, a typical STIG^{CT} session started by facilitators presenting the group of CT data practices that would be the focus of the day, providing definitions for the practices. Then, teachers participated in sample activities that engaged them in data practices within a science learning context. For instance, they collected and analyzed weather data in Excel and used a micro:bit and MakeCode to collect and visualize temperature and light data for a plant growth experiment. Teachers and researchers debriefed these activities by highlighting how each activity had engaged them in multiple CT data practices and discussed how those connected to science learning. These activities were followed by an important task: researchers and teachers spent about 30 min co-designing a draft science lesson plan that integrated CT practices, discussing activities and CT practices that could be a part of those lessons. Finally, teachers completed written reflections about the session and their learning process.

Like the instructors in the methods course, facilitators in the STIG^{CT} used a variety of resources to define CT in the first year of the project and switched to the framework we present here in the second year. A more detailed description of the STIG^{CT} design and its evolution throughout the study can be found in our other publications (Killen, Coenraad, Byrne, Cabrera, Ketelhut, Mills, & Plane, in press).

4.3 | Participants

The participants for this study were pre-service teachers enrolled in the elementary science methods course described above and in-service teachers who taught at elementary grade levels in local school districts and participated in the STIG^{CT}. We detail the gender and racial/ethnic composition of participant pools per project year and learning environment in Table 2 to demonstrate that the composition of our sample is roughly equivalent to the larger teacher force which is predominantly composed of women and white teachers. When citing data excerpts associated with a specific participant, we use the letter R for a "resident" or pre-service teacher and the letter M for a mentor or in-service teacher.

4.4 | Data collection and analysis

The development of the framework and assessment of our interventions relied on the analysis of multiple data sources from each context and each year of the project. In this section,

TABLE 2 Study participants.

	Year 1		Year 2	
	Pre-service	In-service	Pre-service	In-service
Environment				
Methods course	55	-	63	-
STIG^{CT}	13	11	21	19
Gender				
Women	45	11	58	17
Men	4	-	5	1
Unknown	6	-	-	1
Race/Ethnicity				
American Indian	-	-	1	-
Asian or Asian-American	6	1	9	3
Black or African-American	3	2	2	1
Hispanic or Latinx	8	1	10	1
White	32	10	42	16
Other	-	-	1	-
Unknown	6	-	-	-

Note: Participants can identify with more than one race/ethnicity.

we describe the data collection and analysis processes that took place during each year of the project. Data collection and analysis in the first year informed the design of the framework we present here while data collection and analysis during the second year inform the affordances and challenges of the framework we describe below. A summary of each data source is in Table 3.

5 | DATA FOR FRAMEWORK DESIGN

The first part of this article focuses on the decisions behind the framework design. The decisions we made when designing the framework were heavily influenced by the combined insights we gathered from analyzing data from the first year of the project (for a summary of data sources across both years, see Table 3). We collected observational field notes, lesson plan designs, written reflections, focus group responses, self-efficacy surveys, and course artifacts from the methods course and the STIG^{CT} in year one. To analyze these data, we employed a combination of inductive and deductive techniques (Bogdan & Biklen, 2007) most appropriate for our diverse, primarily qualitative, data sources. This allowed us to gain insight into how teachers conceptualized CT and applied it to science lessons.

First, to best understand how teachers had integrated CT into lessons, we conducted a deductive analysis of 43 lessons from both interventions using Weintrop et al.'s (2016) taxonomy. This allowed us to identify which CT practices teachers had integrated in their lessons and to determine how they were conceptualizing those practices. For instance, the excerpts "students will collect data about the observable properties of their own shoe on a work sheet"

TABLE 3 Data sources.

Data source	Collection	Analysis	Context
Field notes	Written notes by researchers including facilitators and instructors	Inductive	Methods course and STIG^{CT}. Years 1 and 2.
Lesson plans	Submitted by teachers at the end of participation in each intervention	Inductive; deductive using Weintrop et al. (2016)	Methods course and STIG^{CT}. Years 1 and 2.
Written reflections	Completed with prompts throughout participation in both interventions. Focused on learning process and implementing lessons	Inductive	Methods course and STIG^{CT}. Years 1 and 2.
Course or session artifacts	Rubrics, templates, and artifacts from learning activities. For example, written group responses to the prompt "what is CT to you?"	Inductive	Methods course and STIG^{CT}. Years 1 and 2.
Focus groups and interviews	Administered by a researcher with three to five teachers at the end of each intervention.	Inductive	Methods course and STIG^{CT}. Years 1 and 2.
Self-efficacy surveys	Administered electronically before and after each intervention.	Paired *t*-test, mixed-method approach	Methods course and STIG^{CT}. Years 1 and 2, redesigned between years.
Lesson co-design videos	Recorded during STIG^{CT} sessions, one camera per group; transcribed for analysis	Inductive	STIG^{CT}. Year 2

(R20) and "The students collect data when the class walks around the school to locate where there is erosion" (R27) were coded as claiming to integrate the practice of data collection.

This analysis showed that teachers were most likely to integrate data practices from Weintrop et al.'s (2016) taxonomy but conceptualized those practices broadly by describing any type of observation as "data collection."

With the purpose of better understanding how teachers had learned about CT and their experience in our interventions, we conducted additional inductive analyses combining data from written reflections, focus group responses, lesson plans, and observations. We specifically read the data looking for common themes based on "regularities and patterns" (Bogdan & Biklen, 2007; p. 164) that emerged from teachers' experiences. Then, for each analysis, multiple researchers developed a codebook based on a subset of those emergent themes and then applied that codebook to the available data. In each case, two or more researchers individually coded a portion of the data, discussed any discrepancies, and reflected decisions on code applications into the codebook. Then, one or more researchers completed the coding of the data based on the agreed-upon criteria.

For example, from reading the reflections and focus groups, we saw different perspectives emerging around the compatibility of CT as an innovation against the current state of science education. So, two researchers led an analysis where they identified three different perspectives (CT as an add-on, CT as new but compatible, and CT as embedded) applied those categories as a codebook to the reflections and focus groups, and examined how those perspectives were explained by teachers.

This group of analyses provided us with specific insights related to how they defined CT such as confusion or aversion toward computer science terminology; conflation between CT and science inquiry; and doubts about whether Weintrop et al.'s taxonomy was appropriate for elementary students. These insights were the basis for the framework design decisions we made and how we modified our interventions in the second year of the project.

6 | DATA FOR FRAMEWORK ANALYSIS

In the second part of the paper, we show the affordances and challenges that our framework provided during our two interventions. These affordances and challenges were identified by conducting data collection and analysis processes similar to the first year of our project with the modification of existing protocols and the addition of new data sources (see Table 3 for a summary of sources). Specifically, we made modifications to focus groups, interviews, and surveys while adding a video recording of co-design sessions.

We modified focus groups and interview protocols to better capture participant experiences in our interventions and in integrating CT into their science instruction. Our new questions focused on how teachers had implemented CT in their classrooms and the barriers they identified in implementing the lessons they designed. We also redesigned the self-efficacy survey to better reflect survey methodology and self-efficacy measurement research (for a detailed account of the design and validation of the second-year survey, see Cabrera, Byrne, Ketelhut, et al., 2021). Additionally, we added the recording of lesson co-design sessions in the STIG^{CT}. Using video cameras, we were able to capture the co-design process while allowing researchers to fully co-participate as designers instead of focusing on making observations or data collection.

Our analysis procedures were also modified and expanded in Year 2. We again conducted both deductive and inductive analyses of second year project data. The deductive analyses were again focused on the lessons teachers created. However, instead of Weintrop et al.'s taxonomy, we used our new framework as an analytical tool to understand how teachers had integrated CT. We analyzed 65 lessons from teachers in the methods course and 22 unique lessons from STIG^{CT} participants (who sometimes collaborated with peers or mentors to create lessons). For each lesson, we annotated which CT practices the teachers had claimed to integrate and developed a process to determine which of those practices we—the researchers—also identified in the lessons. Two researchers co-coded which CT practices were integrated into each lesson, discussing any disagreements to apply a consistent definition of CT practices to all lessons. A detailed account of the analysis and findings of these lessons is beyond the scope of this article and found elsewhere (Coenraad et al., 2021). However, we share some high-level insights from that analysis to provide evidence of the affordances and challenges our new framework provided.

With the purpose of better understanding how teachers had developed their CT understanding and their experiences participating in our interventions, we conducted multiple inductive analyses that integrated data from all our sources. First, we read and watched all the available data, which allowed us to identify multiple themes warranting further analysis. These themes included topics like collaboration among pre- and in-service teachers (Killen, Coenraad, Byrne, Cabrera, & Ketelhut, 2020); co-design session facilitation strategies (Cabrera, Byrne, Killen, et al., 2021); how the design of intervention activities impacted teacher learning (Killen, Coenraad, Byrne, Cabrera, Ketelhut, Mills, & Plane, in press); and a characterization of how teachers integrated CT into lessons (Cabrera, 2021). While we do not have sufficient space to detail each inductive analysis we performed, they all followed a similar procedure. First, two or more researchers created a preliminary codebook based on themes that had emerged during the preliminary read of the data. Then, those researchers applied the codebook to a portion of the data and compared their results. They discussed any differences in code application and one or more researchers completed the coding of all remaining data.

The affordances and challenges we describe in this article emerged from these analyses—particularly by combining insights from the deductive analysis of lesson plans with insights from the inductive analyses described above. For example, to demonstrate how the framework provided opportunities to discuss the boundaries of CT practices, we share participant conversations that we identified by analyzing researcher facilitator moves in co-design sessions and how those conversations were reflected in teachers' use of CT practices in their lesson plans and written reflections.

6.1 | Use of existing frameworks

While one of the main goals of the ICTTE project was to design a CT framework for elementary science teacher education, we acknowledged that we needed a starting point on which to base our instruction to teachers in each of our two intervention components. To guide our pedagogy with educators, we heeded recommendations from general teacher learning frameworks. For example, following Clarke and Hollingsworth (2002) model of teacher professional growth, we designed activities targeted to impact teacher's Personal domain (knowledge, beliefs, and attitudes) while also giving them opportunities to apply their knowledge in lesson design activities (domain of practice). Considering Gregoire's (2003) model of teacher cognition and appraisal, we designed activities that communicated the importance of CT integration to motivate teachers and push them forward through the appraisal process toward "accommodation" and true conceptual change (p. 157).

Regarding the definition of CT, we reviewed extant CT frameworks and decided to present teachers with three main sources in Year 1. We used Weintrop et al.'s (2016) framework, which is specifically oriented to CT integration into science education; Barr and Stephenson's (2011) piece, which contained specific examples of K-12 CT integration; and the Computer Science Teachers Association and International Society for Technology in Education (2011) toolkit, which expanded the view of CT to include attitudinal dispositions. Our goal with including these sources was to give teachers a diverse set of resources that they could consult to help us co-create a new framework informed by their professional knowledge. We defined CT for teachers using Weintrop et al.'s (2016) taxonomy, Barr and Stephenson's (2011) and Computer Science Teachers Association and International Society for Technology in Education's (2011) articles. However, we explained that no one definition was perfect and that tensions existed between the definitions in each source. However, during our sessions together, we focused mostly on Weintrop et al.'s taxonomy and centered instruction in both environments around this list of CT practices. This decision was based on the fact that Weintrop et al.'s (2016) framework was grounded in the observation of *science* practices, provided a concise taxonomy that teachers could use as a reference, and represented the most recent literature on the subject.

And, as we detail below, using these frameworks together as conceptual tools for teachers had limited success. Teachers were confused by the multitude of sources, found some of the technical language inaccessible, and perceived certain CT practices explained in the frameworks as developmentally inappropriate for elementary students. Therefore, after using these existing frameworks in Year 1, we leveraged insights from data analyses to inform the development of a new targeted framework for tear 2.

7 | DECISIONS BEHIND FRAMEWORK DESIGN

The final framework (Figure 1) includes 13 practices divided into four groups—reflecting its close relationship to Weintrop et al.'s (2016) taxonomy. Table 4 details each practice.

The framework was designed as a consensus document within the research team aimed at addressing some of the challenges in CT learning and science lesson design in Year 1. Below, we describe each major decision that affected the design of the framework and link those decisions to evidence from Year 1. For a summary of these design decisions, see Table 5.

7.1 | Elimination of CS jargon and reframing of programming practices

The first decision we made in our framework design was to rid the document of computational jargon and allow teachers to use less technical language to describe CT practices—particularly practices associated with programming. In the lessons teachers designed, we noticed an aversion toward integrating programming or "computational problem-solving" practices (Weintrop et al., 2016). Only one lesson, designed by a group of teachers in the STIG^{CT}, integrated

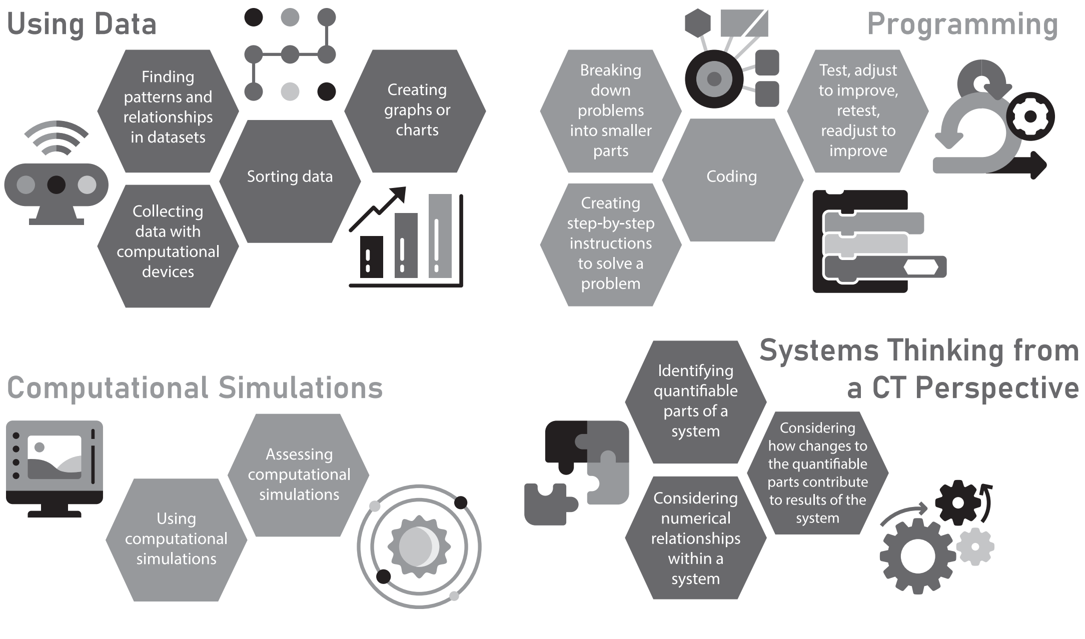
Using Data
Finding patterns and relationships in datasets
Sorting data
Creating graphs or charts
Collecting data with computational devices
Programming
Breaking down problems into smaller parts
Coding
Test, adjust to improve, retest, readjust to improve
Creating step-by-step instructions to solve a problem
Computational Simulations
Assessing computational simulations
Using computational simulations
Systems Thinking from a CT Perspective
Identifying quantifiable parts of a system
Considering how changes to the quantifiable parts contribute to results of the system
Considering numerical relationships within a system

FIGURE 1 Computational thinking (CT) integration into elementary science framework.

TABLE 4 Framework practices.

Practice	Description	Example
Using data: Collecting data with computational devices	This practice involves using a computational device to record observations and can also involve transforming data so a computational agent can process it.	Using a micro:bit to record temperature and light level or entering data into Excel spreadsheets for computational analysis.
Using data: Finding patterns and relationships in datasets	This practice involves making conclusions by analyzing collected data.	At the lower grades, analyses may be simple like counting frequencies and comparing them across groups. At higher grades, they can include computational methods like calculating and comparing means and trends.
Using data: Sorting data	This practice involves using a specific method or a computational agent to organize data by some criteria. Usually, this practice is used to make finding patterns and relationships easier for students.	Sorting daily average temperature by day to notice patterns related to seasons. Sorting the same data from highest to lowest to determine outliers and extreme values.
Using data: Creating graphs or charts	This practice involves representing observations or data in basic graphical forms Students can create the representations by following specific procedures or aim to understand how a computational agent constructs them.	At lower grades, this practice can involve simple techniques like frequency bars. At higher grades, students can use computational graphing tools like Excel to plot trends in animal populations across time.
Programming: Breaking down problems into smaller parts	This practice involves finding the successive steps that need to occur to solve a bigger problem and tackling each step individually. This practice is often called "decomposition" and can include discussions around how each step in a solution affects the rest.	In aiming to program a robot to continuously map the ocean (classroom) floor, students determine the tasks of the robot moving forward, being able to turn, avoid obstacles, and make recordings at specific periods.
Programming: Creating step-by-step instructions to solve a problem	This practice involves describing procedures in a way a computational agent can execute to solve a problem or perform a task. The emphasis should be on thinking about the constraints a computational agent possesses and how to adapt instructions to work within those constraints.	Following the example above, students translate the ideas of "moving forward," "avoiding obstacles" and "making recordings" into steps that can be executed by the robot's parts and are consistent with its capabilities.

(Continues)

TABLE 4 (Continued)

Practice	Description	Example
Programming: Coding	This practice involves using a computer to write instructions aimed at completing a task. This includes block-based programming applications.	Following the example above, students create the program that the robot will execute in the appropriate software.
Programming: Test, adjust to improve, retest, readjust to improve	This practice involves repeatedly using results from each attempt to modify a proposed solution. Importantly, this practice should embody intentional iteration (not random tinkering), where each instance represents a change based on the last iteration's results.	Following the example above, students work to code the robot to "avoid obstacles" by noticing why it failed (e.g., it did not make a wide enough turn) and making corrections to its code. In the case of overcorrection (made too wide a turn), students make a new change that is closer to the goal than their first attempt.
Simulations: Using computational simulations	This practice involves using interactive applications that represent a scientific phenomenon to understand it or test different scenarios.	Students use a simulator of deforestation, land development and rainfall to see the impact of each factor on the likelihood of flood.
Simulations: Assessing computational simulations	This practice involves comparing a simulation to the real-world phenomenon it represents and discuss the limitations of computational simulations and how they can be improved.	Students discuss how the flood simulation fails to consider additional factors like trash and consider how that factor would impact other parts of the simulation if included.
Systems thinking: Identifying quantifiable parts of a system	This practice involves recognizing how parts of a whole can be represented with numbers.	Students translate weather ideas like rainfall into numerical measures such milliliters of rain.
Systems thinking: Considering the numerical relationships within a system	This practice involves calculating how parts of a system are linked in correlational and measurable ways.	Students discuss how the number of rabbits in an area should decrease if the number of predators, or wolves, increases.
Systems thinking: Considering how changes to the quantifiable parts contribute to results of the system	This practice involves determining how an increase or decrease in quantity in one part of the system contributes to the increase or decrease of another part of the system.	To continue the example above, students engaging in this practice would determine how many fewer rabbits survive after each model iteration when one additional predator is introduced.

programming practices by having students program a Code-a-Pillar robot to make it "find food and avoid predators." While this lack of programming integration may have been due to a variety of factors like lack of comfort or confidence with programming, lack of technological resources in schools, or a preference toward other practices that are easier to integrate into science learning, we believe that inaccessible language likely *also* played a role in teachers

TABLE 5 Framework design decisions.

Design decision	Explanation	Evidence
Elimination of CS jargon and reframing of programming practices	Reworded programming practices to avoid terms like *algorithm*, *parallel processing*, *conditional*, *iteration*, or *modular*.	Teachers rarely adopted programming practices and did not use CS vocabulary in their lesson plans or reflections. Instead, they opted to explain CT practices in their own language, which sometimes evidenced interpretations of CT terms that did not retain a connection to computing.
Distinction of CT from scientific inquiry	Added words to ensure that the practice was integrated with a computational flavor—avoiding non-computational uses of typical science terms like *data*, *system*, and *model*.	In most lesson plans, we found teachers using some of the terms from the available frameworks in non-computational ways. For example, they referred to "data collection" as noting the characteristics of a shoe or "creating models" as making a drawing of a plant.
Elimination of higher-level practices	Cut practices that required a level of abstraction too high for most elementary level grades, such as creating computational models and analyzing computational systems.	As the initial CT practices taxonomy (Weintrop et al., 2016) we used was designed for college and high-school students, teachers recognized that some practices were not developmentally appropriate for K-5 grades.

avoiding the integration of these practices. And, more importantly for our purposes here, we believe this was a factor we *could* influence through the design of our framework.

One reason to believe that language played a role in teachers' hesitance to integrate programming practices was that some teachers *did* claim to integrate practices from Weintrop et al.'s (2016) Computational Problem-Solving category. However, in their lesson reflections, they only mentioned integrating *parts* of those practices, typically leaving out some technical terminology. For example, while Weintrop et al.'s taxonomy has a practice labeled "troubleshooting and debugging," seven lessons from our teachers claimed to integrate the practice of "troubleshooting" but only one mentioned "debugging"—which is a more CS-specific process of systematically assessing and correcting a solution. Focus groups further showed this difference in teachers' adoption of the word *troubleshooting* over *debugging*. A total of nine different teachers mentioned "troubleshooting" as an important part of CT, while none mentioned "debugging." For instance, two teachers talked about the importance of this practice for science learning:

Interviewer:	What kinds of science related understandings, skills, or practices do you think that computational thinking can help learners develop?
Teacher 1:	I think a big one in science is the problem solving and trying again.
Teacher 2:	Troubleshooting?

Teacher 1: Yeah, troubleshooting. Because in science, not everything is going to be straightforward. Well, in anything. But that's a really big one, I think, for science.

At the same time, we noticed that teachers described a type of systematic thinking, which two teachers called "algorithmic thinking," as a key part of CT that was not necessarily reflected in the taxonomy. For instance, four teachers mentioned integrating the CT practice of "problem decomposition" and four other teachers mentioned this term as an important part of CT in focus groups. Two teachers said they integrated a CT practice that involved solving problems "step-by-step" and four other teachers called defining and ordering "steps" a CT practice during focus groups:

> I think during the science class, it is really important for students to know how to solve the problem step by step. Or in making something, they need to think of the steps before they actually do it. Because—I think it's involving the computational thinking because they have to arrange the steps, which I think is important because if the steps are mixed-up, they won't get a good result. So, I think the—thinking out the steps clearly is important in science. (R38)

The combination of teachers' avoidance of CS-specific terms, lack of integration of CT programming practices, and propensity to use more general terms to describe a type of systematic thinking associated with CT convinced us to reframe programming practices using less technical language. Our goal was to make programming practices more accessible for teachers while leveraging terminology that they would be comfortable with. Specifically, we tried to reframe programming practices as broader than just the act of writing instructions for a computational device. We also wanted our framework reflect the type of systematic thinking strategies that teachers had described as part of CT and are often used alongside programming.

For instance, we reflected the idea of systematically "troubleshooting" in the practice labeled *Test, adjust to improve, retest, readjust to improve*. We described the concept of "problem decomposition" as *Breaking down problems into smaller parts*. And, following teachers' non-technical descriptions of algorithmic thinking, we wrote the practice of *Creating step-by-step instructions to solve a problem*.

While this may seem like a compromise in rigor or oversimplification of CT concepts, our decision was based on the difficulty teachers had with adopting technical language and their avoidance of programming as a group of practices to integrate. Our hope was that in lowering the language barrier inherent in some practices in Weintrop et al.'s taxonomy (e.g., "preparing problems for computational solutions") we could encourage more teachers to integrate these practices into their lessons. And, in grouping these practices under the umbrella of programming practices, we aimed to retain their computational nature and avoid oversimplifying them.

7.2 | Distinction of CT from scientific inquiry

Another language-related design decision we made was to purposefully address conflations between CT and scientific inquiry practices. A salient finding from the analysis of Year 1 data was that teachers often applied labels of CT practices to activities that could be more accurately described as engaging children in scientific inquiry. We mostly saw this issue in how teachers described integrating data practices, modeling and simulations, and systems thinking.

7.3 | Data practices

Teachers claimed to integrate data collection and analysis practices into 26 of their lessons, but only four of them referred to data that was numerical or could be processed in a *computational* way. In the other 22 instances, "data collection" and "analysis" were used to describe activities like exploring some phenomenon through observation, taking qualitative notes, and trying to deduce an explanation for that phenomenon from their notes. While these activities may be important to support students in developing data collection and analysis skills, we considered it imperative to clarify that CT entails a different *flavor* of these practices, one that is linked to computing and focused on how computational devices can support data uses. To be clear, we do not argue that qualitative observations are not "data." Instead, we mean to differentiate how teachers can integrate data practices *to engage students in CT* from other uses of data in science learning.

To make the distinction between scientific inquiry and CT clearer around data practices, we deleted the terms "manipulating," "analyzing," and "visualizing" data and replaced them with language that more explicitly described the activities we saw as engaging students in CT. For example, we added that data collection could be supported by *computational devices* to differentiate it from other forms of data collection like observation. We also included the practice of "sorting data" to describe the process determining a procedure to arrange data in a way that facilitates analysis—which is typically done through a computational process. We also specified the practice of *Finding Patterns and Relationships in Datasets*—emphasis on *datasets*—to differentiate it from the process of making inferences from simple observations.

7.4 | Modeling and simulations

Another set of practices we reworded with a similar aim were the modeling and simulation practices. In Year 1, 16 teachers indicated integrating CT practices in their lesson plans and reflections using the word "model" such as "modeling complex systems" and "using computational models to design and test solutions." While these practices are included in Weintrop et al.'s taxonomy, teachers used them to describe activities that involved *non-computational* models and simulations. Out of the 16 instances of teachers integrating these practices, only one teacher used the term to refer to a *computational* simulation while two others used it to describe an illustration as a model and the remaining 13 referred to physical models. While the use of visual or physical models is important in scientific inquiry and considered within the core practice of "developing and using models" in NGSS, these uses of the term do not reflect the computational nature of modeling and simulation practices *under CT*.

To make this distinction, we rephrased the practices as *Using* and *Assessing Computational Simulations*, adding the word "computational" and removing the problematic "model" altogether. While we saw a few instances of teachers also using the word "simulation" to describe a non-computational activity, we found that "model" was clearly a word that teachers could more easily interpret in non-computational ways.

7.5 | Systems thinking

Finally, our analysis of lessons and focus groups from Year 1 showed that teachers used labels of "systems thinking practices" to describe activities that explored a system in *any* way—not just computational approaches. A total of 13 lessons indicated that teachers had integrated systems thinking practices, with some of them specifying which practices were integrated such as "understanding relationships within a system" and "thinking in levels." Yet, our analysis of the lessons disagreed with those claims. We found that teachers were merely providing activities where students explored a topic which *involved* a system or were broadly interpreting "system" as anything that could be divided into different parts.

For example, a teacher designed a lesson where students watched a YouTube video about the water cycle including processes like precipitation, condensation, and evaporation. Students then worked in groups to "create a flow chart illustration of the water cycle process" and explained their illustrations. In her reflection, the teacher described this lesson as engaging students in CT when they were "able to explain a system and its components and their interactions." Yet, the consideration of the water cycle as a system did not include any computational aspect—therefore conflating systems thinking *as CT practices* with the same practices as part of more traditional science learning or scientific inquiry.

Other teachers who described students engaging in systems thinking in their lessons did so by referring to common objects that could be divided into parts as "systems." For example, a teacher designed a lesson where students analyzed the different parts of their shoes, conceptualizing each shoe as a "system." Other examples of everyday objects being described as "systems" were a pen and pretzels.

To avoid these uses of systems thinking practices that did not contain a computational aspect, we specified that systems should be analyzed in a *quantifiable* way when rewording our practices. This decision, however, was contested and highly debated between researchers with expertise in computer science and science education in the research team. On the one hand, we wanted to strengthen the connection between these practices in our framework and computing, such as specifying the exploration of *computational systems* like models and computational devices. On the other hand, we wanted to keep the practices as being in service of science learning, as opposed to fully centering computing. In the end, our use of the terms "quantifiable" and "numerical" show a compromise where we specify how to engage in these practices in computational ways while also retaining a focus on their function toward disciplinary learning.

7.6 | Elimination of higher-level practices

The final design decision we made was to limit the framework to contain only those practices that teachers deemed developmentally appropriate and to reformulate other practices to be adequate for the elementary level. We based this decision on the observation that, as expected, teachers did not integrate *all* CT practices described in the Weintrop et al.'s (2016) taxonomy into their lessons in Year 1. Because Weintrop et al.'s framework focuses on practices for the high-school and college level, it includes practices that would be developmentally inappropriate for elementary level students. While the research team knew of this framework limitation before presenting it to teachers, participating teachers in Year 1 were instrumental in determining *which* practices were above the elementary level—and how the rest of the practices could be implemented in a developmentally appropriate way. Although we presented and explained all the practices in the taxonomy to teachers, we did not observe any teachers attempting to integrate the practices of *creating computational models* or *defining systems and managing complexity*—which could indicate that teachers do not see those practices as appropriate for integration at the elementary level. Instead, teachers were most likely to integrate CT practices related to collecting and analyzing data and using models and simulations.

However, teachers may have chosen to only integrate those practices due to factors unrelated to developmental appropriateness. Additional observations informed our decision to remove higher-level practices. For example, in a focus group, a teacher suggested that differentiation across grades may be reflected in *which* CT practices they engage in. When asked about how her integration of CT in fourth grade would compare to that of her peers who taught in the second grade, she explained: “I think that a lot—the computational thinking practices that they [2nd grade teachers] engage in and the skills that they use may be different, but I think that overall, the understanding of concepts and everything is definitely enhanced through computational thinking.” (R09).

Additionally, teachers saw coding or programming as a high-level practice for elementary school, as suggested by their lack of integration of programming practices and by focus group responses: “[CT requires] a lot of planning, which we've seen with coding. It's not the kind of thing that you can just, you know, jump into. It takes a lot of front-loading and thinking things through before the actual act of doing.” (R09).

Therefore, we decided to make our framework compatible with teachers' pedagogical expertise and perceptions of developmental appropriateness. We eliminated practices that teachers did not consider appropriate for their students while leaving practices that they could find challenging but doable—like coding.

8 | AFFORDANCES AND CHALLENGES OF FRAMEWORK IMPLEMENTATION

After redesigning aspects of both interventions in our project, we used our framework as the main definition of CT with teachers in Year 2. In this section, we describe the two primary affordances and two primary challenges that our analysis indicated the framework provided in the second year of our project. To present the affordances and challenges of using the framework, we draw on evidence from field notes, lesson plans, focus group and interview responses, and co-design videos. In the following section, we expand on how to build on these affordances and counter the challenges when using the framework in educational environments.

8.1 | Affordance 1: Opportunities for CT practice boundary discussions

The new framework seemed to promote more targeted conversations around the boundaries of CT practices between participants and researchers—particularly when teachers had the opportunity to co-design lessons and ask for the rationale behind the wording of certain framework practices. These discussions, in turn, allowed researchers to stress the computational perspective of the framework and explain how, even as some practices were lacking computational technical terms, they still retained their computational nature.

For instance, the following conversation during a co-design session at the STIG^{CT} illustrates the types of exchanges that the use of the framework promoted:

Teacher: So, like looking at data in itself...

Researcher: I mean, science does that, math does that. So, there isn't really a computational thinking component to it if you're just collecting data and graphing the data, you know.

Teacher: But if you're making conclusions from the data—

Researcher: That's still science. That's scientific reasoning.

Teacher: So, I guess in order for any of these to be CT, they have to almost align with this one [points at practice labeled *Collecting data with computational devices*]. With the like—computer device. Like, that's what we're going to bring it home with.

Researcher: No, because you can do [data practices] outside of a computer device, but basically when you're thinking about *how* you go about doing it—just doing the sorting of data probably isn't computational thinking. But if you have a process by which you sort the data, when you're talking about the *process*, then you're talking about an algorithm, and *that's* part of computational thinking. Here, actually by sorting, we mean sorting or categorizing, sorting into groups. In computer science, we don't use sorting for that. It's a different term. To be able to take the data and just put it into groups is the scientific reasoning process, but to actually think about *what criteria you're using* to sort categories and what rules and what process, then you're going into computational thinking. Does that make sense?

Similar conversations around the framework practices and how they applied to different science activities occurred throughout the two interventions—but primarily when teachers had to co-design lessons during the STIG^{CT}. For example, a pre-service teacher (R35) challenged her mentor who was describing an activity and calling it a "simulation" by asking "How do we make this computational?" and "Does that count as computational?" Other similar conversations on the boundaries of CT revolved around the difference between videos and computational simulations, what the "numerical" aspect of systems thinking practices entailed, and whether programming practices could be present without a programming environment.

In the methods course, an instructor used the framework practices to differentiate between systems thinking practices within CT and systems thinking as more commonly known in science education. He particularly focused on the numerical aspect of the practices as written in the framework. Observation notes describe this moment:

> [The instructor clarified:] Computational systems thinking might be a question like: How does the heat transfer impact precipitation numerically? Whereas regular systems is just identifying the components. So, in elementary school just identifying the parts of system will set them up for computational systems thinking maybe in middle school. (Field notes).

Written reflections also showed that teachers noticed and appreciated these opportunities to discuss the boundaries of CT practices as afforded by the framework. For instance, in a description of the benefits of participating in the STIG^{CT}, a pre-service teacher wrote that the group provided "resources materials & help differentiating between what is and is not CT" (R26). Another teacher, reflecting on the session that had just finished, wrote: "interesting conversation on computational thinking definition" (R35).

8.2 | Shift toward computational perspective

We believe these opportunities to use the framework to discuss the boundaries of CT practices may have impacted how teachers described their final lessons using practices from the framework. When teachers reflected on their lessons, they often used the names of the practices in the framework to describe the activities. A detailed analysis of the lessons that teachers created is explained elsewhere and is outside of the scope of this article (Coenraad et al., 2021) but we share a few high-level insights and some examples from that analysis to demonstrate this impact. We specifically saw a shift toward a computational perspective in how teachers used and described engagement in two groups of CT practices: data practices and simulations.

8.2.1 | Data practices

In teachers' integration of CT practices in lessons, we saw a shift toward data activities involving *numerical or computational* data. Remember, in the first year of our project we found that 86% of teachers were referring to activities where they only collected or analyzed qualitative data—without a numerical or computational aspect—when they identified an activity as representing CT data practices. In Year 2, about half (48%) of instances where teachers identified data practices referred to uses of numerical data or processing data with computational tools.

As an example, in the second year a group of two pre-service teachers and their two mentors designed a lesson where students would use a coiled spring (slinky) to explore the topic of waves. The children followed a systematic procedure to test different speeds (slow, medium, fast) of waving the slinky. They measured the time a wave took to get to the other end and counted how many waves were created when moving the slinky at each speed. Students then contributed their small group data to a class-wide spreadsheet and used Excel to find averages, order their data, and spot outliers. These uses of data practices better reflect the computational or numerical nature of the CT framework and show a contrast with the mostly qualitative applications of “data” that we saw in the first year of our project.

8.2.2 | Simulations

The way teachers used Simulation practices also demonstrate a shift toward a computational perspective. Specifically, in Year 2, 17 of the 23 lessons where teachers claimed their students engaged in using computational simulations were referring to an activity that included a *computational* simulation. Clearly, the conflation between computational simulations and physical or non-computational models we saw in Year 1 was less pronounced in this second year. Some examples of these lessons included using PhET simulations for students to explore friction and gravity; an online planetarium simulation where students could identify constellations; and platforms where students could create their own electric circuits with batteries, light bulbs, and voltmeters.

Some teachers also explicitly showed an understanding of the differences between the affordances of computational simulations and those of other types of models like physical representations. Particularly, teachers saw value in simulations as they allowed students to explore invisible or hard-to-illustrate scientific phenomena. For example, a pre-service teacher reflected on how the simulations she explored would help students “explore organisms (and ecosystems) that aren't accessible to kids" (R05). Another teacher—whose wave lesson is described above—used a computational simulation *and* a physical model in her lesson to teach different aspects of waves. When reflecting on the benefits of the computational version, she noticed that the physical model (coil springs) was prone to issues like tangling and would sometimes make waves inconsistent or hard to measure. On the other hand, the computational simulation "can go on slow motion/rewind your steps and you can control more" with them. She indicated she would focus more on the computational simulation next time she taught the lesson.

Our decisions around language use in the framework may have contributed to these improvements in teachers' application of CT practice labels. The inclusion of more computational versions of each practice may also be related to our decision to differentiate CT from scientific inquiry and more typical elementary science practices. This differentiation was emphasized by the research team when discussing the framework and likely contributed to more teachers conceptualizing each practice from a computational perspective.

8.3 | Affordance 2: Centralized CT definition

A second affordance our framework provided was the centralization of CT definitions. Because CT is a contested term and we provided teachers with frameworks from different perspectives in Year 1, participants felt confused on the nature and boundaries of CT. Creating and using one consistent framework throughout Year 2 provided a centralized resource that teachers and researchers could reference when designing CT-infused science lessons.

The benefits of this centralized definition were evident during lesson co-design sessions. In these, teachers determined whether the activities they were suggesting would engage students in CT by going over the framework and assessing whether students would enact those practices. For example, a group of teachers designing a lesson where students would program a Code-a-Pillar to help it "obtain food and avoid predators" had a conversation about which practices students would engage in:

Teacher 1: ...at least the *Creating step by step instructions*, which you just said under programming. And the *Coding* and programming.

Teacher 2: Then I feel like it would be the last one [*Test, adjust to improve, retest, readjust to improve*] too, because they have to like, try it and see if—is it going to get to where I want it to get?

Teacher 1: I feel like it's almost everything on here [Programming practices], because you're breaking it down too when you're programming it, right?

These conversations around the framework practices were common in co-design sessions and allowed teachers to clearly determine whether their lesson plans would engage students in CT practices. As described above, they also provided researchers with opportunities to clarify how those practices were defined (also see Cabrera, Byrne, Killen, et al., 2021).

Some teachers also reflected on the benefits of having a unique resource for CT. For example, a pre-service teacher pointed to the framework as a helpful conceptual tool: "I really like the resources telling us what websites to look on, especially that image [framework] that you gave us on what computational thinking consists of" (R42). Other participants specifically mentioned the CT practices in the framework as part of the "tools and ideas that they would bring back to their classroom" in written reflections. These testimonies suggest that teachers appreciated having a centralized resource for CT, as they were able to reference its practices during lesson co-design sessions with researchers.

To be clear, the first affordance we presented is to *promote* discussion around the boundaries of CT practices, while the second is to *centralize* how CT is defined. We do not intend to make the claim that these conversations or our unique CT definition had a specific impact on teachers' understanding of CT. Our point is that the framework and the wording of its practices provided concrete opportunities to discuss the boundaries of CT practices as they applied to classroom activities and to provide a consistent tool for lesson design.

9 | CHALLENGE 1: SINGLE PRACTICE INTEGRATION

A challenge we encountered when implementing our framework with teachers was the integration of a single CT practice into science lessons, as opposed to a more holistic approach to integration. Although most teachers were able to integrate CT practices into their lesson plans, we noticed that some of these applications were instances of single CT practices. This tendency to integrate a single practice was most prominent in cases where teachers integrated the practices of *Using Computational Simulations* or *Coding*. We use the case of computational simulations to illustrate this challenge.

Out of the 17 lessons that integrated Simulation practices, 8 of them integrated *only* Computational Simulations. A lesson that illustrates this single practice integration started by asking students to discuss different ideas that they have about stars. It continued by showing students two educational videos on stars, their brightness, and their relative distance to Earth. Then, the students engaged in a computational simulation where they were able to zoom in and out of the Milky Way and use their computer mouse to change the angle of observation showing and hiding different stars. Finally, as an assessment, students completed a written reflection on what they had learned.

These instances of integration of a single simulation practice may be due to the low barriers of using simulations in the classroom. Teachers appreciated having free resources that students could explore on their own without requiring extensive teacher support. In focus groups, in-service teachers described simulations as the "simplest way to jump off the boat" (M1) in terms of CT integration and easy to connect to the curriculum because "they are already written for you" (M7) and "you can find one for any topic" (M3).

While we were encouraged to see how many teachers had found relevant computational simulations to integrate into their lessons, we lamented the missed opportunities to integrate other CT practices that could make those experiences more meaningful and better integrated. We elaborate on how to promote more holistic integrations of CT later in the paper.

9.1 | Challenge 2: Balancing computational perspectives with accessible language

One of the most pervasive challenges that arose from using our framework with teachers was the use of language around CT practices. We found that our rewording attempts had mixed results, achieving our intended goal with some groups of CT practices and creating new difficulties with others.

9.1.1 | A stronger computational perspective

Our framework aimed to nuance the meaning of "data" to indicate that, from a computational perspective, data should mean information that can be represented and/or analyzed by a computer. As described in our first affordance, we saw a shift toward a computational perspective in how teachers described data practices in their lessons in Year 2. However, some lesson plans from our second year of implementation also showed that some teachers were *still* likely to call students recording simple qualitative observations—which can rarely be interpreted by a computational agent—as falling under the bigger term of engaging with CT data practices.

For example, a teacher designed a lesson where 2nd grade students compared the physical characteristics of materials like felt, cotton balls, pencils, paper clips, and sandpaper. Students ordered materials along different dimensions like strength, flexibility, and texture. In her reflection, the teacher considered this activity as engaging students in "using data by recording and analyzing observable properties" (R04). While this activity met the science objective of making observations, these practices were not enacted with a computational perspective as they did not use computational devices, considered how a computer could interpret the data, nor determined a procedure for data collection or analysis that could be supported by computers.

9.1.2 | Generalization of programming practices

On the other hand, we found an interesting overcompensation effect when analyzing how teachers integrated programming practices from our framework. Because we wanted to make programming less intimidating for teachers, our framework strived to avoid using technical jargon and make programming practices more understandable for educators without a background in computer science. However, our analysis indicates that this rephrasing strategy may have led some teachers to apply programming practices labels to non-computational activities that may distort their essence. Specifically, 8 of the 18 lessons where teachers claimed to integrate programming practices took a broad interpretation of those practices that did not retain a computational perspective.

For instance, we saw some teachers use the labels of practices like *Breaking down problems into smaller parts* to describe typical processes in science instruction without a computational perspective. An example is a participant using this practice to describe an activity where students learned about recycling and sorted different trash items into bins to be reduced, reused, or recycled. The sorting process was not broken down into parts that a computational agent could help solve, nor was a systematic process (algorithm) developed to apply to each new item of trash. Instead, students compared each item to a list of characteristics that should go in each bin and then made these decisions with a partner. Nevertheless, the teacher labeled this activity as engaging students in CT: "Students were able to break the problem into smaller parts—easier to manage problems" (R56). These applications of programming practices seemed to be devoid of a connection to computing.

9.1.3 | Mixed results across groups of CT practices

The challenge of teachers applying programming practices without a computational perspective seems to contradict our first affordance, where teachers applied data practices in more computational ways. We suspect that these two seemingly opposite findings stem from our compromises in framework language. While we strove to differentiate CT from scientific inquiry, those efforts centered around data practices, which were most often conflated with scientific inquiry practices in Year 1. This led to more precise descriptions of data practices with a computational perspective, possibly explaining the changes we saw on the use of those practices. On the other hand, we tried to make programming practices more accessible for teachers who were unfamiliar with computer science technical language. Therefore, the language of these practices became *less* precise from a computational perspective, possibly explaining some of the uses of programming practices we saw in Year 2. Below, we provide suggestions for using this framework—including ways to strike a good balance to differentiate CT from scientific inquiry while making computational perspectives accessible for teachers.

A final piece of evidence that illustrates the challenge of balancing computational perspectives with accessible language regards teachers' integration of systems thinking practices. In the lesson plans where teachers claimed to integrate systems thinking practices into their lessons in Year 2, they again mostly referred to activities that involved exploring a scientific phenomenon that could be considered a "system" but ignored the quantifiable and numerical focus we intended to stress in the framework.

For example, teachers described lessons where students watched videos about "systems" like planets rotating around the sun, the seasonal cycle, and devices inspired by animal characteristics as engaging students in systems thinking CT practices. These instances of CT integration show that our efforts to describe systems thinking practices in ways that teachers can understand but retain a computational perspective may not have been successful.

Our analyses and available data do not allow us to systematically determine *why* teachers ignored or overlooked the quantifiable aspect of these practices, but a discussion during a lesson co-design session suggests that it may be related to perceptions of developmental appropriateness and math ability. In this conversation, a researcher provided an example of the practice *Considering numerical relationships within a system* where students would see a numerical relationship between number of fish and number of food sources available each year. The teachers recognized that students could interpret a graph going "up or down" but that adding numbers would be ineffective because "they are not with proportions yet."

10 | FRAMEWORK USE CONSIDERATIONS

As this article presents the first implementation of our framework and the affordances and challenges it provided in two educational environments for teachers, we make recommendations for those interested in using the framework in their own work with teachers. These considerations aim to counter the challenges and build on the affordances we identified above.

10.1 | Aim for holistic integration

The categorization of CT into practices in our framework can make it seem like integrating CT means integrating *any* of the practices in the framework. In our work, we saw some teachers integrate single practices and sometimes focus on whether an activity "counted" as CT. As described in challenge #1, about half of teachers who integrated CT through simulations *only* integrated the practice of *Using computational simulations*. We saw a similar pattern within the practice of *Coding*, where teachers integrated only a single, simple coding activity into their lesson.

Angeli et al. (2016), in their framework of CT teacher knowledge, similarly describe how teachers, when enacting the framework, should aim to minimize "compartmentalization and fragmentation" (p. 51). The authors argue that, because CT is a complex skill, students would benefit from engaging with CT through activities that allow them to practice that skill in complex ways. Specifically, they recommend using real-life issues to contextualize CT activities and sequencing activities in increasing complexity. We agree with Angeli et al.'s emphasis on holistic integration. Because CT is meant to be used in investigations of scientific phenomena, we think it is best if teachers aim to integrate multiple practices in their lessons, particularly if they create activities where one practice can support another. And, importantly, teacher educators should try to guide teachers in understanding how practices relate *to each other*, as they weave through multiple activities and grade levels.

For example, integrations of computational simulations could also include practices like collecting and analyzing data that comes from that simulation. In higher grades, students could also assess computational simulations to think about how it represents the scientific phenomenon. Teachers could also help students see how the simulations operate by defining numerical relationships between different factors of the system they simulate. Teachers could further integrate programming practices by allowing students to explore simulations in programming platforms like Scratch where they can "see inside" and edit those simulations to make them more complex or accurate.

10.2 | Stress computational perspectives

The framework practices can be interpreted in varied ways. The challenge of "balancing computational perspectives with accessible language" shows that teacher educators who use the framework should consider this balance carefully. The issue of teachers interpreting the practices without a computational perspective was most common in uses of the word "data" and in some applications of programming practices as described above. Here, we recommend that instructors or facilitators stress computational perspectives when using the framework.

This guidance comes from the transformational goals of CT and the kinds of CT applications we saw from teachers when they did *not* adopt a computational perspective. Because the goal of CT integration is to prepare students to understand and leverage computing as a way to engage with science content, we believe stressing a computational perspective is more likely to achieve that goal. The lessons where teachers did not adopt a computational perspective and instead interpreted data and programming practices more broadly seemed to resemble typical science instruction and were therefore less likely to be transformational.

This issue is especially delicate at the elementary level, where some of the instances of collecting and analyzing qualitative data or "unplugged" programming activities would be considered by others as a valid enactment of CT. To be clear, we support these practices in the elementary science classroom because they are integral to science education—regardless of whether they warrant a label of "CT practice." However, in order to enact change in how students engage with science to integrate CT, teachers (and teacher educators) need a clear distinction between what is "preparing students for a computational world" and what is business-as-usual. We believe that stressing a computational perspective can help make this distinction.

At the core of this issue is a definitional problem: how do we know which applications of these practices "count" as CT? The challenge of differentiating CT from scientific inquiry we encountered suggests that such judgments are not straightforward. One possible way to solve this difficulty is to establish a clear criterion to determine whether an educational activity helps fulfill the goal behind efforts to integrate CT. Because our arguments to integrate CT into science instruction are based on the premise that students will need to engage with computing to participate in modern science, our criterion would be contingent on preparing students for that specific goal. In other words, we would ask of each lesson design: *Are these activities labeled as computational thinking preparing students to engage with computing as a way to learn science?*

This guiding question could differentiate lessons that engage students in typical—albeit valuable—scientific practices like qualitative data collection, from those that specifically prepare them to engage with scientific content through CT. Of course, this criterion begs the follow-up enquiry: how do we know whether an activity prepares a student to engage with computing? A good start would be to aim toward Extend levels of integration (Waterman et al., 2019) where CT is an integral part of "doing science" in the classroom. These experiences should position CT as a tool that advances scientific learning while fostering student competence and confidence in computing during formative years. While we offer this position as a starting point, we believe future research on different applications of CT and their effects on the development of computing proficiency in children could help teacher educators make these judgments more accurately.

10.3 | Purposefully address equitable integration

In our motivation for this work, we explained that integrating CT at the elementary science level could promote the development of a scientific identity in *all* students before they make important decisions about academic paths. We attribute that goal to the larger initiative of CT integration into science, not to our framework design. However, the *use* of our framework should explicitly consider how its implementation can promote that goal.

Our experience using the framework with teachers suggests that if teacher educators do not specifically support teachers in integrating CT with an equity lens, teachers may inadvertently limit their integration to students who are already perceived as exceptional (Coenraad, Mills, et al., 2020; Coenraad, Plane, et al., 2020). For example, we observed that some teachers decided to integrate CT practices like *Coding* as extensions of lessons that students who finished early would get to engage in. These opportunities were then limited only to those students who were academically more advanced, preventing *all* students from engaging in CT learning. Similarly, some teachers opted for designating students with preexisting proficiency in computing as "student leaders" but these students were often boys—reinforcing gender stereotypes about who has expertise in computing.

The framework does not promote equity in CT learning opportunities by itself—it needs to be paired with an equity approach by teachers and teacher educators. When using the framework, educators should explicitly encourage teachers to integrate CT activities in ways that leverage the cultural, linguistical, and academic competences of all their students. This is particularly important considering that girls and students of color have been traditionally underrepresented in science and computing careers. By addressing equity issues and how applications of CT can ameliorate or reinforce them, teacher educators can help teachers implement CT in a way that provides learning opportunities for all their students.

Overall, we believe this framework can be an important conceptual tool for teacher educators to present CT to teachers in a way that is compatible with existing scientific practices and curricula, developmentally appropriate for elementary students, and intelligible without a computer science background. However, we also believe that the framework could be improved.

11 | FRAMEWORK IMPROVEMENTS

As part of a design-based research project, the framework is not complete—it should be continually adapted and improved based on its success in guiding CT learning for teachers. Here, we briefly suggest possible ways to improve the framework in future implementations.

Logically, these improvements are tightly related to the use suggestions we described above. For example, a beneficial edit to the framework would be to add a visual and explicit reminder to stress computational perspectives when applying each practice. Anecdotally, we have tried this change in a new year of the methods course by adding an all-encompassing umbrella icon to the framework, indicating that each practice should still fall under the "umbrella of computing." Making this reminder visual could also help teachers differentiate between scientific inquiry or typical science instruction and the CT applications we are promoting.

The second expansion of the framework could involve communicating how practices relate to each other, to promote holistic integration. Teacher educators and researchers could map how each practice provides opportunities to engage in other practices—particularly with practices from other groups. For example, we could make a clear indication that *Assessing Computational Simulations* could provide opportunities for *Coding* as students look "under the hood" to edit the simulations they assess.

A final suggested improvement could also focus on revisiting systems thinking practices and *Assessing Computational Simulations* for developmental appropriateness and intelligibility. The compromise between scientific and computational perspectives we described resulted in practices with long names that teachers seldomly integrated. It may be possible that systems thinking practices are not developmentally appropriate for the elementary level (at least the lower grades) or it may be likely that different descriptions of these practices would encourage teachers to integrate them. Future work, informed by practitioners, could help make further decisions on how the framework presents these practices.

12 | CONCLUSION

In this article, we described the design of a framework for the integration of CT into elementary science to be used with teachers. Our experiences during the first year of working with elementary teachers using existing frameworks led us to create a new resource with little technical language, clearer distinctions between CT and scientific inquiry, and only CT practices deemed developmentally appropriate for elementary students.

Our application of the framework in two learning environments with teachers allowed us to identify affordances and challenges of the framework. On the one hand, the resource promoted discussions of boundaries of CT practices while centralizing CT definitions in one document—benefit teachers highlighted as useful. On the other hand, the framework may have contributed to non-computational applications of programming practices and was incapable of fully eradicating persisting questions on the boundaries between scientific inquiry and CT—particularly in uses of "data" and "systems thinking." The conceptualization of CT as a set of practices may also have contributed to integrations of single CT practices as sufficient.

As a result, we suggest that teacher educators use this framework with important considerations. We believe that, when used with those considerations, this framework can be an important conceptual tool for teachers to understand CT and develop ways to integrate it into their elementary science instruction.

ACKNOWLEDGMENTS

The authors want to thank every teacher, proofreader, thought partner, and advisor they had in the process of ideating, writing, and refining this article. Special thanks to Dr. Randy McGinnis, Troy Sadler, and Li Ke for their contributions to the project and the thinking behind this work. This material is based upon work supported by the National Science Foundation under Grant No. 1639891. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

ORCID

Lautaro Cabrera https://orcid.org/0000-0001-6559-4258
Diane Jass Ketelhut https://orcid.org/0000-0001-5307-1281

REFERENCES

Adler, R. F., & Kim, H. (2018). Enhancing future K-8 teachers' computational thinking skills through modeling and simulations. *Education and Information Technologies*, *23*(4), 1501–1514. https://doi.org/10.1007/s10639-017-9675-1

Anderson, M. (2017). *Among high school seniors, interest in science varies by race, ethnicity*. Pew Research Center Retrieved from https://www.pewresearch.org/fact-tank/2017/01/04/among-high-school-seniors-interest-in-science-varies-by-race-ethnicity/

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, *19*(3), 47–57. https://doi.org/10.2307/jeductechsoci.19.3.47

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, *2*(1), 48–54. https://doi.org/10.1145/1929887.1929905

Basu, S., Biswas, G., Kinnebrew, J., & Rafi, T. (2015). Relations between modeling behavior and learning in a computational thinking based science learning environment. *Proceedings of the 23rd International Conference on Computers in Education, ICCE* 2015, 184–189.

Bogdan, R. C., & Biklen, S. K. (2007). Qualitative data analysis and interpretation. In *Qualitative research in education: An introduction to theories and methods* (5th ed., p. 336). Pearson.

Bower, M., & Lister, R. (2015). Teacher conceptions of computational thinking—Implications for policy and practice.

Bower, M., Wood, L., Lai, J., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education*, *42*(3), 53–72. https://doi.org/10.14221/ajte.2017v42n3.4

Brown, A. L. (1992). Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences*, *2*(2), 141–178. https://doi.org/10.1207/s15327809jls0202

Bybee, R. W., & National Science Teachers Association (2010). Teaching science as inquiry. In *The teaching of science: 21st century perspectives*. Corwin Press. https://doi.org/10.4135/9781452219677.n1

Cabrera, L. (2021). *Computational thinking in the elementary classroom: How teachers appropriate CT for science instruction* [Doctoral dissertation]. https://drum.lib.umd.edu

Cabrera, L., Byrne, V., Ketelhut, D. J., Coenraad, M., Killen, H., & Plane, J. (2021). Measuring teacher self-efficacy for integrating computational thinking in science (T-SelECTS). *Educational Innovations and Emerging Technologies*, *1*(1), 3–14.

Cabrera, L., Byrne, V., Killen, H., Coenraad, M., Ketelhut, D. J., & Plane, J. (2021). Facilitation as assessment of teacher's computational thinking knowledge during lesson co-design. Presented at the American Educational Research Association Conference 2021.

Chang, Y., & Peterson, L. (2018). Pre-service teachers' perceptions of computational thinking. *Journal of Technology and Teacher Education*, *26*(3), 353–374.

Clarke, D., & Hollingsworth, H. (2002). Elaborating a model of teacher professional growth. *Teaching and Teacher Education*, *18*(18), 947–967.

Code.org, CSTA, & ECEP Alliance. (2021). 2021 state of computer science education: Accelerating action through advocacy. Retrieved from https://advocacy.code.org/stateofcs

Coenraad, M., Cabrera, L., Killen, H., Plane, J., & Ketelhut, D. J. (2021). Computational thinking integration in elementary teachers' science lesson plans. In *Computational thinking in PreK-5: Empirical evidence for integration and future directions* (pp. 11–18). Association for Computing Machinery.

Coenraad, M., Mills, K., Byrne, V. L., & Ketelhut, D. J. (2020). Supporting teachers to integrate computational thinking equitably. In *Proceedings of Research on Equity and Sustained Participation in Engineering, Computing, and Technology* (*RESPECT 2020*), Portland, OR: IEEE.

Coenraad, M., Plane, J., Ketelhut, D. J., Cabrera, L., Mills, K., & Killen, H. (2020). STIGCT: The science teacher computational thinking inquiry group. Retrieved from https://education.umd.edu/stigct

Collins, A. (1992). Toward a design science of education. In E. Scanlon & T. O'Shea (Eds.), *New directions in educational technology* (pp. 15–22). Springer-Verlag. https://doi.org/10.1007/978-3-642-77750-9_2

Computer Science Teachers Association, & International Society for Technology in Education. (2011). Computational thinking: Leadership toolkit. Retrieved from http://www.iste.org/docs/ct-documents/ct-leadershipt-toolkit.pdf

Curzon, P., Mcowan, P. W., Plant, N., & Meagher, L. R. (2014). Introducing teachers to computational thinking using unplugged storytelling. *WiPSCE '14 Proceedings of the 9th Workshop in Primary and Secondary Computing Education*, 89–92.

Denning, P. J. (2017). Computational thinking in science. *American Scientist*, *105*(1), 13.

Dickes, A. C., Kamarainen, A., Metcalf, S. J., Gün-Yildiz, S., Brennan, K., Grotzer, T., & Dede, C. (2019). Scaffolding ecosystems science practice by blending immersive environments and computational modeling. *British Journal of Educational Technology*, *50*(5), 2181–2202. https://doi.org/10.1111/bjet.12806

Gadanidis, G., Cendros, R., & Floyd, L. (2017). Computational thinking in mathematics teacher education. *Contemporary Issues in Technology and Teacher Education*, *17*(4), 458–477. https://doi.org/10.1007/978-3-319-52691-1_13

Google Inc., & Gallup Inc. (2016). Diversity gaps in computer science: Exploring the underrepresentation of girls, Blacks and Hispanics. Retrieved from http://goo.gl/PG34aH

Gregoire, M. (2003). Is it a challenge or a threat? Of model cognition a dual-process. *Educational Psychology Review*, *15*(2), 147–179.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hammerness, K., Darling-Hammond, L., Bransford, J., Berliner, D., Cochran-Smith, M., McDonald, M., & Zeichner, K. (2012). How teachers learn and develop. In L. Darling-Hammond & J. Bransford (Eds.), *Preparing teachers for a changing world: What teachers should learn and be able to do* (Vol. 10, pp. 358–389). Jossey-Bass. Retrieved from https://login.proxy.library.msstate.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=psyh&AN=2005-13868-010&login.asp&site=ehost-live

Hestness, E., Ketelhut, D. J., McGinnis, R. J., Plane, J., Razler, B., Mills, K. M., Cabrera, L., & Gonzalez, E. (2018). *Computational thinking professional development for elementary science educators: Examining the design process*. The Society for Information Technology and Teacher Education (SITE).

Hutchins, N. M., Biswas, G., Maróti, M., Lédeczi, Á., Grover, S., Wolf, R., Blair, K. P., Chin, D., Conlin, L., Basu, S., & Mcelhaney, K. (2020). C2STEM: A system for synergistic learning of physics and computational thinking. *Journal of Science Education and Technology*, *29*, 83–100.

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, *82*, 263–279. https://doi.org/10.1016/j.compedu.2014.11.022

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, *26*(2), 175–192. https://doi.org/10.1007/s10956-016-9663-z

Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2019). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, *1–15*, 174–188. https://doi.org/10.1007/s10956-019-09798-4

Ketelhut, D. J., & Cabrera, L. (2020). Computational thinking in early childhood and elementary science and engineering education. Report for the National Academies of Sciences, Engineering and Education committee on Enhancing Science and Engineering in Prekindergarten through Fifth Grade.

Killen, H., Coenraad, M., Byrne, V., Cabrera, L., & Ketelhut, D. J. (2020). Reimagining computational thinking professional development: Benefits of a community of practice model. In *Proceedings of the International Conference of the Learning Sciences* (*ICLS 2020*), Nashville, TN: ISLS.

Killen, H., Coenraad, M., Byrne, V., Cabrera, L., Ketelhut, D. J., Mills, K. M., & Plane, J. (In Press). Teacher education to integrate computational thinking into elementary science: A design-based research study. *ACM Transactions on Computing Education*.

Lave, J., & Wenger, E. (1991). Situated learning: Legitimate peripheral participation. In *Situated learning: Legitimate peripheral participation*. Cambridge University Press. https://doi.org/10.1017/CBO9780511815355

Lee, I., Grover, S., Martin, F., Pillai, S., & Malyn-Smith, J. (2020). Computational thinking from a disciplinary perspective: Integrating computational thinking in K-12 science, technology, engineering, and mathematics education. *Journal of Science Education and Technology*, *29*(1), 1–8. https://doi.org/10.1007/s10956-019-09803-w

Malyn-smith, J., Lee, I. A., Martin, F., Grover, S., Evans, M. A., & Pillai, S. (2018). Developing a framework for computational thinking from a disciplinary perspective. *Proceedings of the International Conference on Computational Thinking Education*, *2018*, 182–184.

McGinnis, J. R., Hestness, E., Mills, K., Ketelhut, D. J., Cabrera, L., & Jeong, H. (2020). Preservice science Teachers' beliefs about computational thinking following a curricular module within an elementary science methods course. *Contemporary Issues in Technology and Teacher Education*, *20*(1), 85–107.

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record*, *108*(6), 1017–1054. https://doi.org/10.1111/j.1467-9620.2006.00684.x

Mouza, C., Yadav, A., & Ottenbreit-Leftwich, A. (2018). Developing computationally literate teachers: Current perspectives and future directions for teacher preparation in computing education. *Journal of Technology and Teacher Education*, *26*, 333–352. Retrieved from https://www.learntechlib.org/primary/p/184602/

Mouza, C., Yang, H., Pan, Y.-C., Yilmaz Ozden, S., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology*, *33*(3), 61–76. https://doi.org/10.14742/ajet.3521

National Research Council. (2011). Report of a workshop on the pedagogical aspects of computational thinking. https://doi.org/10.17226/13170

National Science and Technology Council. (2018). Charting a course for success: America's strategy for STEM education. (Issue December).

NGSS Lead States. (2013). Next generation science standards: For states, by states. (Issue November). https://doi.org/10.17226/18290

Peel, A., Sadler, T. D., & Friedrichsen, P. (2019). Learning natural selection through computational thinking: Unplugged design of algorithmic explanations. *Journal of Research in Science Teaching*, *56*(7), 983–1007. https://doi.org/10.1002/tea.21545

Rachmatullah, A., & Wiebe, E. N. (2021). Building a computational model of food webs: Impacts on middle school students' computational and systems thinking skills. *Journal of Research in Science Teaching*, *59*, 585–618. https://doi.org/10.1002/tea.21738

Rich, K. M., Yadav, A., & Larimore, R. A. (2020). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies*, *25*(4), 3161–3188. https://doi.org/10.1007/s10639-020-10115-5

Sadik, O., Ottenbreit-Leftwich, A. T., & Nadiruzzaman, H. (2017). Computational thinking conceptions and misconceptions: Progression of preservice teacher thinking during computer science lesson planning. In P. Rich & C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 221–238). Springer.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, *18*(2), 351–380. https://doi.org/10.1007/s10639-012-9240-x

Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. *Proceedings of the 16th Koli Calling International Conference on Computing Education Research—Koli calling '16*, 120–129. https://doi.org/10.1145/2999541.2999542

The Design-Based Research Collective. (2003). Design-based research: An emerging paradigm for educational inquiry. *Educational Researcher*, *32*(1), 5–8. https://doi.org/10.3102/0013189X032001005

Vincent-Ruz, P., & Schunn, C. D. (2018). The nature of science identity and its role as the driver of student choices. *International Journal of STEM Education*, *5*, 48.

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, *20*(4), 715–728. https://doi.org/10.1007/s10639-015-9412-6

Waterman, K. P., Goldsmith, L., & Pasquale, M. (2019). Integrating computational thinking into elementary science curriculum: An examination of activities that support students' computational thinking in the service of disciplinary learning. *Journal of Science Education and Technology*, *29*, 53–64. https://doi.org/10.1007/s10956-019-09801-y

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *39*(1), 195–196. https://doi.org/10.1145/1227504.1227378

Yadav, A., Krist, C., Good, J., & Nadire Caeli, E. (2018). Computational thinking in elementary classrooms: Measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, *28*, 371–400. https://doi.org/10.1080/08993408.2018.1560550

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education*, *14*(1), 1–16. https://doi.org/10.1145/2576872

Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011). Introducing computational thinking in education courses. *Proceedings of the Special Interest Group on Computer Science Education Conference SIGCSE'11*. https://doi.org/10.1145/1953163.1953297

How to cite this article: Cabrera, L., Ketelhut, D. J., Mills, K., Killen, H., Coenraad, M., Byrne, V. L., & Plane, J. D. (2023). Designing a framework for teachers' integration of computational thinking into elementary science. *Journal of Research in Science Teaching*, 1–36. https://doi.org/10.1002/tea.21888