

USE OF BLOCK-BASED CODING IN SCIENTIFIC MODELING

by

LUCAS LIMA DE VASCONCELOS

(Under the Direction of ChanMin Kim)

ABSTRACT

Scientific modeling and coding are critical skills for K-12 students. Teaching scientific modeling entails supporting K-12 students in constructing, manipulating, and developing their own models to advance knowledge of a complex system or phenomenon. Learning to code prepares students to solve problems using computational concepts and processes. This dissertation research begins the pathway toward supporting students in using block-based coding to externalize and develop their models through the process of creating simulations of science phenomena. Research indicates that teachers need professional learning on scientific modeling and teaching with coding. An instructional module and online tool named Coding in Scientific Modeling Lessons (CS-Model) were designed to support preservice science teachers' learning to code and to integrate block-based coding into scientific modeling lessons. CS-Model was designed and developed based on proposed design guidelines that emerged from review of relevant literature (Chapter 2). A qualitative pilot study was conducted to implement CS-Model in a methods of science teaching course (Chapter 3). Results indicated that study participants developed more refined models of science phenomena and perceived coding as a key skill for K-12 education but failed to design lessons in which block-based coding supports scientific modeling. Results informed recommendations for CS-Model redesign. A mixed methods study

was conducted to implement the redesigned CS-Model (Chapter 4). The study investigated if and how preservice teachers' epistemological understanding of models and modeling as well as their understanding of computer science concepts changed after participation in CS-Model. The study also examined how participants used coding in scientific modeling lessons. Results showed that most participants developed a more sophisticated epistemological understanding of models and modeling, as well as an understanding of computer science concepts. Results also indicated that participants designed lessons wherein block-based coding was used to support scientific modeling either as an exploration tool or as a research tool. Additionally, lessons focused on computer science practices rather than specific concepts. It was noteworthy that there were inconsistencies between participants' epistemological understanding of models and modeling and their lesson design. Conclusions and directions for future research are discussed.

INDEX WORDS: Scientific Models, Scientific Modeling, Block-based Coding, Science Simulations, Instructional Scaffolding, Science Education, Lesson Design

USE OF BLOCK-BASED CODING IN SCIENTIFIC MODELING

by

LUCAS LIMA DE VASCONCELOS

BA, University Federal do Ceará, Brazil, 2012

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial
Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2019

© 2019

Lucas Lima de Vasconcelos

All Rights Reserved

USE OF BLOCK-BASED CODING IN SCIENTIFIC MODELING

by

LUCAS LIMA DE VASCONCELOS

Major Professor: ChanMin Kim

Committee: Robert Branch
Lloyd Rieber
J. Steve Oliver

Electronic Version Approved:

Suzanne Barbour
Dean of the Graduate School
The University of Georgia
May 2019

DEDICATION

I dedicate this dissertation to my family, who has always emphasized the importance of education. My father, Elizio Vasconcelos, grew up in the countryside of Brazil. To get to school, he had to walk miles and cross a river every day. My dad covered his books with a plastic bag and held them above his head while walking through water. Hindsight has shown me that my father walked through the waters of a river to pursue his education so I could fly over the waters of the Atlantic Ocean to pursue my education at the University of Georgia.

I also dedicate this dissertation to my mother, Celeste Vasconcelos, who nurtured my passion for reading. My mother has always worked in schools. I have vivid memories of my mother unlocking library doors so I could choose whichever book I wanted to read. It was also my mother who encouraged me to learn English as a foreign language. I am thankful for my mother's encouragement and support.

I would also like to dedicate this dissertation to my siblings, Leticia Vasconcelos and Leones Vasconcelos. There is a word in Portuguese that does not have a direct translation in English. The word is "saudade", and it is the feeling you have when you miss somebody. Despite their saudade, my siblings have encouraged me to pursue my dreams and my PhD degree.

Finally, this dissertation is also dedicated to my husband, Craig Page, who has supported me every day and in so many ways during my journey as a PhD student. He has been emotional support, sound board, best friend, and more. Craig has shown unquestioned faith in me, and has always encouraged me to follow my dreams. I would often catch him thinking of ways to make my life easier so I could focus on my dissertation. To my amazing family: thank you!

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to my major advisor and committee chair, Dr. ChanMin Kim, who accepted me as her advisee in the second year of my PhD. Dr. Kim has offered valuable feedback, emotional support, opportunities for research collaborations, mentoring on job hunting, and more. Dr. Kim inspires me to continue growing as a scholar.

I would like to thank Dr. J. Steve Oliver, who mentored me when I was designing my pilot study, allowed me to conduct the pilot study in his course, and helped me secure the research context for the main dissertation study. Dr. Oliver met with me several times and his mentoring was instrumental for my dissertation research.

I also want to thank my committee members, Dr. Robert M. Branch and Dr. Lloyd Rieber, for the valuable feedback on my dissertation research and support with job hunting, as well as Dr. Daniel Capps, who allowed me to conduct the main dissertation study in his course. I also want to thank Dr. Michael Orey for helping me start my PhD at the University of Georgia.

I want to say thank you to Daisyane Barreto, who helped me throughout the PhD admission process, recommended me for assistantships, introduced me to professional conferences such as AECT, and more. Thank you as well to Natasha Barreto and Robert Sills, who helped me settle down in Athens and opened the doors of their house for me so many times. Natasha has also taken me under her wing as a graduate assistant. Daisyane, Natasha, and Bobby became my family in the United States.

I would like to thank Dr. Júlio Araújo for his mentorship, which ultimately helped me get accepted into the PhD program at UGA. Dr. Araújo awarded me teaching and research assistantships for several years while I was in college. He coached me when I was first learning about research, conference organization and presentations, scholarly writing, and more. Dr. Araújo inspired me to pursue a career in academia.

I want to thank Dr. Jiangmei Yuan, who I have had the pleasure to work with on research projects and publications. Dr. Yuan has become a great colleague and I have learned a lot from working with her. I also want to thank Cory Gleasman and Duygu Umutlu for their assistance with data collection during my final dissertation study.

I am beyond grateful for my friends Coeli Rodrigues, Allana Frota, Samir Nunes, Angelina Oliveira, Eloiza Lima, Clarice de Paula, Aryanni Frota, and Carlos Oliveira, all who have encouraged me in pursuing my PhD. Though we are geographically separated, our friendship remains close and strong.

I would also like to thank all the family members who have supported me in my journey, especially Regina Mota and Itacira Mota for always encouraging and being willing to help and support me in any way.

Finally, I would also like to thank all the faculty and colleagues I met during my PhD. Our interactions and conversations helped shape me into the scholar I am today.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENTS	v
LIST OF TABLES	x
LIST OF FIGURES	xii
CHAPTERS	
1 INTRODUCTION AND LITERATURE REVIEW	1
Statement of Problem.....	2
Research Purposes	7
Significance of the Study	8
Dissertation Overview	8
References.....	11
2 CODING IN SCIENTIFIC MODELING LESSONS (CS-Model).....	24
Abstract	25
Introduction.....	26
Scientific Models and Modeling: A Pragmatic Perspective	28
Instructional Scaffolding.....	33
Framework for Simulation Coding in Scientific Modeling	38
Design Guidelines	40
Coding in Scientific Modeling Lessons (CS-Model)	46
Conclusion	57

References.....	60
3 PREPARING TEACHERS TO USE BLOCK-BASED CODING IN SCIENTIFIC MODELING LESSONS.....	85
Abstract	86
Introduction.....	87
Theoretical Background.....	89
Coding in Scientific Modeling Lessons (CS-Model)	95
Research Purpose and Questions	98
Methods.....	98
Results and Discussion	110
Conclusions and Directions for Future Research.....	122
Study Limitations.....	123
References.....	125
4 CODING SCIENTIFIC MODELS: PRESERVICE TEACHERS’ EPISTEMOLOGICAL UNDERSTANDING, CODING SKILLS, AND LESSON DESIGN.....	141
Abstract.....	142
Introduction.....	143
Relevant Literature.....	144
Purpose and Research Questions	148
Methods.....	149
Results.....	162
Discussion.....	187

Conclusions and Future Research.....	197
Study Limitations.....	199
References.....	201
5 CONCLUSION.....	219
Limitations of the Studies and Future Research Directions.....	222
Implications for Research and Practice.....	224
References.....	227

APPENDICES

A PRESENTATION OF SCIENTIFIC MODELS, MODELING, AND CODING	231
B EXAMPLE OF FOLLOW-UP QUESTIONS USED IN INTERVIEW	233
C CODING PROBLEM APPLIED DURING PRE- AND POST-INTERVIEWS	234
D SLIDES FOR CODING WORKSHOP	235
E CODING WORKSHOP ACTIVITY: KNOCK KNOCK JOKE	239
F EXAMPLE OF SCIENTIFIC MODELING LESSON THAT USES CODING.....	242
G REFLECTION ACTIVITY	250
H SLIDES FOR FACE-TO-FACE CS-MODEL ACTIVITIES	251
I HYPOTHESES SHEET FOR CODING WATER FILTER SIMULATION.....	258
J DEBUGGING SHEET	261
K EXPERIMENTAL DATA SHEET	263
L HOMEWORK.....	264
M ONE PAGE OF RESEARCHERS' DATA ANALYSIS JOURNAL	265
N SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 1	266
O SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 2.....	267

P SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 3.....268

LIST OF TABLES

	Page
Table 2.1: Overview of CS-Model Module Timeline and Activities	47
Table 2.2: Overview of Water Filtration Activities	50
Table 2.3: Modification of Schwarz et al.'s (2009) Framework for the Design of CS-Model.....	56
Table 3.1: Data Sources per Research Question.....	99
Table 3.2: Participant Information.....	100
Table 3.3: Overview of Interview Protocol	106
Table 3.4: Coding Scheme Nodes, Description, and Example Excerpts	108
Table 3.5: Rubric for Lesson Analysis	110
Table 3.6: Compared Epistemic Discourse in Erica and Kathy's Team.....	113
Table 3.7: Lesson Data Analysis	121
Table 4.1: Research Questions, Data Sources, and Analysis Methods.....	149
Table 4.2: Participants' Information and Data Completion.....	150
Table 4.3: Overview of Study Timeline and Procedures	151
Table 4.4: Representative Pre- and Post-interview Questions.....	152
Table 4.5: Blocks, Computer Science and Science Concepts, and Simulation Coding	155
Table 4.6: Adaptation of Upmeier zu Belzen and Krüger's (2010) Framework of Model Knowledge and Coded Examples	158
Table 4.7: Rubric to Assess Conceptual Understanding of Coding Concepts.....	161
Table 4.8: Participants' Levels of Understanding of Models and Modeling.....	163

Table 4.9: Results from Wilcoxon Signed-Rank Tests.....	164
Table 4.10: Changes in Participants' Understanding of Conditionals.....	170
Table 4.11: Changes in Participants' Understanding of Delays	171
Table 4.12: Changes in Participants' Understanding of Loops	171
Table 4.13: Quantitative Results on Participants' Understanding of Coding Concepts.....	172
Table 4.14: Identified Misconceptions per Computer Science Concept.....	175
Table 4.15: Participants' Conceptual Understanding in Pre- and Post-interviews.....	175
Table 4.16: Analysis of Lessons	180
Table 4.17: Use of Coding to Support Scientific Modeling	183
Table 4.18: Participants' Epistemological Understanding and Lesson Design	186

LIST OF FIGURES

	Page
Figure 2.1: Conceptual framework for coding in scientific modeling.....	39
Figure 2.2: Screenshot of Scratch	48
Figure 2.3: Simulation of water filtration system using cotton	52
Figure 2.4: Example of code used in simulation.....	53
Figure 2.5: Storyboard for water filter simulation using cotton	53
Figure 3.1: CS-Model framework	94
Figure 3.2: A screen capture of the CS-Model online tool.....	97
Figure 3.3: Overview of study timeline and procedures.....	101
Figure 3.4: Physical water filter experiment and analogous simulation in Scratch.....	103
Figure 3.5: Code and storyboard from simulation with cotton.....	104
Figure 3.6: Code and storyboard from simulation with 1/4" activated charcoal	104
Figure 3.7: Code and storyboard from simulation with cotton-ground charcoal-cotton	105
Figure 4.1: Participants before and after CS-Model per model knowledge scale	163

CHAPTER 1

INTRODUCTION AND LITERATURE REVIEW

Children intuitively construct and reconstruct scientific models to make sense of physical, biological, and social phenomena long before starting school (National Research Council (NRC), 2012; Samarapungavan, Tippins, & Bryan, 2015). A scientific model is a simplified and analogue representation that one creates to better understand, explain, and predict events pertinent to a complex entity or system without firsthand exposure (Buckley, 2000, 2012; Cook, 2006; Seel, 2014, 2017). Scientific models simplify and highlight specific elements, features, relationships, and/or interactions (Buckley, 2012; Seel, 2014, 2017) though they do not perfectly represent its referent (Harrison & Treagust, 2000; NRC, 2012; Rosenblueth & Wiener, 1945). For instance, a model of the solar system embodies *elements*, such as planets and stars, *features*, such as planets' orbit, *relationships*, such as planets orbiting the Sun, and *interactions*, such as an eclipse caused by blocked sunlight. However, a model of the solar system does not illustrate all aspects of its referent, such as a planet's internal composition.

Children continuously develop their own scientific models based on experiences and prior knowledge (Schwarz et al., 2009; Seel, 2017). Experiences that conflict with children's prior knowledge of a target phenomenon promote cognitive conflict, which prompts them to diagnose issues in an existing model and refine it to validate observable evidence (Clement & Rea-Ramirez, 2008; Henderson & Tallman, 2006; Schwarz et al., 2009). This iterative process of model construction and reconstruction mirrors what scientists do to generate models and knowledge about phenomena in the world (Giere, 2004; Nersessian, 2008; Samarapungavan et

al., 2015). Scientific models serve as conceptual tools used to make sense of macroscopic or microscopic systems or phenomena, which cannot be directly observed (Barak, Ashkar, & Dori, 2011; Cook, 2006; Harrison & Treagust, 2000). Scientific models are critical for science teaching and learning as K-12 students can create and develop models to understand, explain, and investigate science concepts.

Statement of the Problem

Although students intuitively develop models before starting school, such models can be partially correct or inaccurate. One of the goals of science education is to facilitate students' development of models that will help them understand phenomena in the world (Clement, 2000; NRC, 1996, 2012; Seel, 2017). Hence, it is critical that K-12 teachers (1) address students' intuitive models and misconceptions about science phenomena; and (2) support students' construction, development, and manipulation of their own models (Gouvea & Passmore, 2017; Nelson & Davis, 2012; Osborne, 2014). To support students' scientific model development, K-12 teachers need to design instruction in which students are offered opportunities to construct, test, evaluate, and revise their own scientific models through participation in scientific inquiry (Berland et al., 2016; Krajcik & Merritt, 2012; NRC, 2012; Schwarz et al., 2009).

K-12 teachers often use expert-made scientific models solely to explain a scientific phenomenon, which curtails opportunities for students to construct and develop their own models (Schwarz et al., 2009; Schwarz & White, 2005; White & Frederiksen, 1998; Windschitl, Thompson, & Braaten, 2008). Consequently, students' engagement with scientific models is often restricted to memorizing and reciting models created *for* them (Buckley, 2012; Gilbert & Boulter, 2000; White & Frederiksen, 1998). Such teaching *about* scientific models results in students struggling to understand what is beyond a model's surface features such as their purpose

and origin, the existence of competing models to fulfill different representational needs, and the perception that models are only approximations of a referent (Clement, 2000; Harrison & Treagust, 2000; Krajcik & Merritt, 2012; Seel, 2017).

Many K-12 educators have an inaccurate understanding of and lack experience with scientific modeling (Harrison & Treagust, 2000; Justi & Gilbert, 2002; Schwarz et al., 2009). For instance, research has shown that preservice and in-service teachers (a) often do not acknowledge and understand the purpose and nature of scientific models (Grosslight, Unger, Jay, & Smith, 1991; White & Schwarz, 1999), (b) perceive a scientific model as the *right answer* to explain a phenomenon (Abell & Roth, 1995; Gilbert, 1991) rather than as an analogical tool that highlights specific features (Nersessian, 2008; Seel, 2017), (c) presume that self-made student models are merely tools that serve as evidence of prior learning (Schwarz & Gwekwerere, 2007), and (d) rarely facilitate student reflection about model limitations and encourage model improvement (Harrison & Treagust, 2000; Schwarz et al., 2009). Both preservice and in-service teachers need professional learning on how to support scientific modeling (Dass, Head, & Rushton, 2015; Duschl, Schweingruber, & Shouse, 2007; Schwarz et al., 2009)

Computer Simulations

Scientific modeling can be challenging for K-12 students who struggle to mentally simulate a phenomenon and predict consequences of actions or events (Harrison & Treagust, 2000; Krajcik & Merritt, 2012; Nersessian, 2008; Seel, 2017). For instance, students overlook complex information (Lowe, 2004), struggle to create analogies between a model and its referent in reality (Louca & Constantinou, 2003), and/or lack skills to regulate their own thought and learning processes (White & Frederiksen, 1998). To address such issues and facilitate scientific modeling, many computer-based simulation tools have been designed (e.g., Ioannidou,

Repenning, Keyser, Luhn, & Daetwyler, 2010; Papaevripidou, Constantinou, & Zacharia, 2007; Sins, Savelsbergh, van Joolingen, & van Hout-Wolters, 2009; Wieman, Adams, & Perkins, 2008; Wilensky & Rand, 2009; Xie et al., 2011). Simulations are dynamic tools that embody mathematical, computer science, and scientific principles to model a complex system, phenomenon, or events (Bowen & Deluca, 2015; Renken, Peffer, Otreel-Cass, Girault, & Chiocciariello, 2016). To use such tools, learners construct hypotheses about scientific phenomena, manipulate variables or rules, observe simulated evidence on the screen, and confirm or revise their hypotheses (Schwarz & White, 2005; Seel, 2017; Shen, Lei, Chang, & Namdar, 2014).

Mixed results have been reported about the effectiveness of computer simulations on scientific modeling (Klahr, Triona, & Williams, 2007; Winn et al., 2006; Wu, Krajcik, & Soloway, 2001) as learners do not always use them mindfully, which compromises their ability to develop more robust scientific models (Marbach-Ad, Rotbain, & Stavy, 2008; Marshall & Young, 2006). However, research indicates that *self-made* simulations can be an effective strategy to support scientific modeling (Chang, Quintana, & Krajcik, 2009; Cheng et al., 2014; Schwarz & White, 2005; van Joolingen, Aukes, Gijlers, & Bollen, 2015; Wu, McLean, & Powerful, 2001). Self-made simulations make learning meaningful because students apply relevant knowledge and skills to construct an artifact that serves as proof of learning (Girvan, Tangney, & Savage, 2013a; Harel & Papert, 1991; Kafai, 2014; Papert, 1980). Although it is difficult to pinpoint the exact combination of factors that contribute to the effectiveness of simulations, the literature indicates that simulations should (a) be associated with other instructional approaches; (b) include support structures such as scaffolding, collaborative work,

and feedback; and (c) promote higher-order student reflection (Cook et al., 2013; D'Angelo et al., 2013; Scalise et al., 2011; Seel, 2017; Smetana & Bell, 2012; Wu et al., 2001).

Block-based Coding

K-12 students can use block-based coding to construct, modify, and visualize simulations (Papert, 1980; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Wagh & Wilensky, 2012) of science phenomena while partaking in scientific modeling. During block-based coding, one writes code and externalizes their mental representations (Lehrer & Schauble, 2000; Papert, 1980) of a target phenomenon to generate a simulation, such as writing code to predict and/or explain precipitated forms of a chemical reaction. Coding simulations requires mindful engagement in scientific modeling tasks as one codes and manipulates variables of interest for a given phenomenon based on their own knowledge. Block-based coding entails sequentially snapping blocks that embody programming concepts, execute specific commands, and yield an output (Baratè, Ludovico, Mangione, & Rosa, 2015; Weintrop, 2015). Block-based coding mirrors the structure of a programming language without the burden of syntax (Harvey & Mönig, 2010; Li & Watson, 2011; Malan & Leitner, 2007; Price & Barnes, 2015). A key feature in block-based coding tools is visual feedback on whether the code sequence is valid or not (Lye & Koh, 2014; Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010; Weintrop, 2015). Feedback can be provided as a specific sound or an unexpected output on the screen, which one can use as a starting point to debug coding errors (Kim, Yuan, Vasconcelos, Shin, & Hill, 2018).

Coding has been identified as a key skill for K-12 students given the pervasiveness of computing technologies in modern society and the increasing need for workforce proficient in computing technologies and programming (Burke, 2012; K-12 Computer Science Framework, 2016; Lye & Koh, 2014; NRC, 2012). However, research has shown that the number of

computer science teachers in K-12 schools does not meet the demand for coding/programming courses (Google & Gallup, 2015a; Yadav, Gretter, Hambrusch, & Sands, 2016). Additionally, students in computer science teacher certification programs graduate with limited understanding of what teaching computer science in K-12 classrooms entails (Gal-Ezer & Stephenson, 2010). K-12 teachers have also reported either a lack of content knowledge or pedagogical knowledge (Google & Gallup, 2015b; Yadav et al., 2016) that is necessary to offer computer science instruction. K-12 teachers across all areas and grade levels need additional professional learning opportunities on coding and teaching with coding (Century et al., 2013; Yadav et al., 2016) so they can design and teach lessons that effectively integrate coding into other subject areas.

Instructional Scaffolding

One way to support teachers' integration of block-based coding into scientific modeling lessons is through instructional scaffolding. Scaffolding is defined as support that temporarily leverages one's engagement in activities whose completion is beyond their unassisted skills (Belland, 2011, 2014, 2017; Pea, 2004; Reiser, 2004). Scaffolding (a) targets specific skills that should be performed without assistance in the future (Belland, 2014, 2017; Wood, Bruner, & Ross, 1976), (b) supports performance during problem-solving tasks (neither before nor after) (Belland, 2014; Hannafin, Hill, & McCarthy, 2001; Wood et al., 1976), (c) builds on prior knowledge to augment one's skills (van de Pol, Volman, & Beishuizen, 2010; Wood et al., 1976), and (d) provides structure to one's performance while highlighting complexity in key concepts or processes (Reiser, 2002, 2004).

Instructional scaffolding is key to supporting scientific modeling (Seel, 2017; Wojnowski & Pea, 2014). Numerous studies have tested the effectiveness of various types of scaffolding strategies (e.g., prompts, hints, guiding questions, images, simulations) or different levels of

scaffolding support (e.g., Azevedo, Cromley, Moos, Greene, & Winters, 2011; Bjønness & Kolstø, 2015; Clement, 2008; Linn, 2000; Van Zee, Iwasyk, Kurose, Simpson, & Wild, 2001) on K-12 students' scientific modeling. Studies have also examined teachers' understanding and preparedness to teach scientific modeling, and they collectively recognize the importance of teachers adopting a clear scientific modeling framework (e.g., Dass et al., 2015; Schwarz & Gwekwerere, 2007; Van Hook, Huziak-Clark, Nurnberger-Haag, & Ballone-Duran, 2009). However, the literature lacks studies that aim to scaffold K-12 teachers' learning to design scientific modeling lessons.

Research Purposes

This research aimed to scaffold science teachers' learning to code science simulations and effectively integrate block-based coding into scientific modeling lessons. After a review of relevant literature, design guidelines were proposed to accomplish such purposes. The guidelines were applied to design and develop an instructional module and online tool named Coding in Scientific Modeling Lessons (CS-Model). A pilot study was conducted in a teacher education course to investigate preservice science teachers' use of epistemic discourse while coding a simulation, perceptions of teaching with coding, and use of simulation coding in scientific modeling lessons. Results from the pilot study informed the redesign and implementation of CS-Model for the final dissertation study, which examined how participation in CS-Model affects preservice science teachers' epistemological understanding of models and modeling, and coding skills. The study also examined preservice science teachers' use of coding to support scientific modeling in lessons. The ultimate goal of this dissertation research is to prepare preservice science teachers to teach scientific modeling with coding so they can offer such learning experiences to their future students.

Significance of the Study

This research is significant for three reasons. First, this research raised preservice science teachers' awareness of the importance of learning to code for K-12 students, the value of scientific models as conceptual tools and scientific modeling as an instructional approach, and the pedagogical benefits of constructing self-made simulations of science phenomena using block-based coding within scientific modeling instruction.

Second, claims have been made that students can better understand and gain skills involving concepts, practices, and core disciplinary ideas across Science, Technology, Engineering, and Mathematics (STEM) fields if they are exposed to integrated STEM curricula and lessons (K-12 Computer Science Framework, 2016; Kim, Oliver, & Jackson, 2016). This research is a first step towards full STEM integration as it bridges Science and Technology, and it contributes to disseminating the importance of STEM learning to participants, researchers, and educators.

Third, preservice science teachers who participated in this research are better prepared to design lesson units and teach scientific modeling using block-based coding to their future students. On a societal level, this study addresses K-12 schools' need for educators who have the content knowledge and pedagogical content knowledge to integrate coding into K-12 instruction (Google & Gallup, 2015b; Yadav et al., 2016).

Dissertation Overview

This dissertation has manuscript-style chapters. The first manuscript (Chapter 2) is *Coding in Scientific Modeling Lessons (CS-ModeL)*. The chapter first provides a theoretical framework to define scientific models and modeling using a pragmatic perspective. Then the paper discusses results from research on K-12 teacher preparation for teaching scientific

modeling and coding, with special emphasis on their need for professional learning. Then, a discussion on the use of science simulations as scaffolds for science learning is presented. Subsequently, the chapter proposes a conceptual framework to integrate block-based coding simulations into scientific modeling instruction and discusses foreseen pedagogical benefits. Next, the chapter presents design guidelines for professional learning on coding science simulations and designing scientific modeling lessons using coding. Each guideline is discussed based on relevant literature. Then, the chapter elucidates how each guideline was materialized in the design and development of CS-Model. Finally, the chapter discusses directions for future research. This manuscript received the Best Student Paper Award within the Instructional Technology Special Interest Group during the 2019 American Educational Research Association annual meeting in Toronto, Canada.

Chapter 3 is entitled *Preparing Teachers to Use Block-based Coding in Scientific Modeling Lessons*. This chapter reports results of a pilot study conducted with five preservice teachers and one in-service teacher attending a methods of science teaching course. The study examined how participants engage in epistemic discourse during simulation coding, perceive coding as a teaching tool, and use coding in scientific modeling lessons. Regarding epistemic discourse, results showed that (a) simulation coding stimulated preservice teachers to share models, engage in joint reflection, correct lingering misconceptions, and create a consensus model; (b) lack of error debugging skills was distracting as teachers alternated between discourse about science concepts and debugging discourse; and (c) there were few instances of conflict argumentation. Regarding perceptions of coding, preservice teachers believe that coding is a foundational skill to be taught from early grades though professional learning is needed; they also believe that teaching with coding offers pedagogical benefits, such as critical thinking.

Regarding participants' lessons, most preservice teachers failed to design authentic scientific inquiry, and their lessons either focused on development of scientific modeling or coding skills, but not both. Pilot study results and practical experience informed CS-Model revisions.

Chapter 4 is entitled *Coding Scientific Models: Preservice Teachers' Epistemological Understanding, Coding Skills, and Lesson Design*. This chapter reports results of the final dissertation study, which implemented the redesigned CS-Model instructional module and online tool. The study investigated how preservice science teachers' participation in CS-Model affected their epistemological understanding of models and modeling, and understanding of the computer science concepts loops, delays, and conditionals. The study also investigated how participants designed used coding to support scientific modeling in lessons. This was a predominantly qualitative mixed methods study. Results revealed that many participants' epistemological understanding of scientific models and modeling and understanding of computer science concepts improved. Moreover, most participants' lessons addressed practices (e.g., abstraction, debugging) rather than specific concepts. Participants used coding either as a research tool or as an exploration tool. Inconsistencies were found between participants' epistemological understanding and their lesson design.

Chapter 5 presents an overview of results from the three manuscripts and conclusions about integrating block-based coding simulations into scientific modeling. The chapter also presents a reflection on future research and practice on use of coding in scientific modeling and beyond.

References

- Abell, S. K., & Roth, M. (1995). Reflections on a fifth-grade life science lesson: Making sense of children's understanding of scientific models. *International Journal of Science Education, 17*(1), 59–74. <https://doi.org/10.1080/0950069950170105>
- Azevedo, R., Cromley, J. G., Moos, D. C., Greene, J. A., & Winters, F. I. (2011). Adaptive content and process scaffolding: A key to facilitating students' self-regulated learning with hypermedia. *Psychological Test and Assessment Modeling, 53*(1), 106–140.
- Barak, M., Ashkar, T., & Dori, Y. J. (2011). Learning science via animated movies: Its effect on students' thinking and motivation. *Computers & Education, 56*(3), 839–846. <https://doi.org/10.1016/j.compedu.2010.10.025>
- Baratè, A., Ludovico, L. A., Mangione, G. R., & Rosa, A. (2015). Playing music, playing with music: A proposal for music coding in primary school. Retrieved from <http://files.eric.ed.gov/fulltext/ED562460.pdf>
- Belland, B. R. (2011). Distributed cognition as a lens to understand the effects of scaffolds: The role of transfer of responsibility. *Educational Psychology Review, 23*(4), 577–600. <https://doi.org/10.1007/s10648-011-9176-5>
- Belland, B. R. (2014). Scaffolding: Definition, current debates, and future directions. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (4th ed, pp. 505–518). New York, NY: Springer.
- Belland, B. R. (2017). *Instructional scaffolding in STEM education*. Cham: Springer International Publishing. <https://doi.org/10.1007/978-3-319-02565-0>

- Berland, L. K., Schwarz, C. V., Krist, C., Kenyon, L., Lo, A. S., & Reiser, B. J. (2016). Epistemologies in practice: Making scientific practices meaningful for students. *Journal of Research in Science Teaching*, 53(7), 1082–1112. <https://doi.org/10.1002/tea.21257>
- Bjønness, B., & Kolstø, S. D. (2015). Scaffolding open inquiry: How a teacher provides students with structure and space. *Nordic Studies in Science Education*, 11(3), 223–237.
- Bowen, B., & Deluca, W. (2015). Comparing traditional versus alternative sequencing of instruction when using simulation modeling. *Journal of STEM Education: Innovations and Research*, 16(1), 5.
- Buckley, B. C. (2000). Interactive multimedia and model-based learning in biology. *International Journal of Science Education*, 22(9), 895–935. <https://doi.org/10.1080/095006900416848>
- Buckley, B. C. (2012). Model-based learning. In N. M. Seel (Ed.), *Encyclopedia of the sciences of learning* (pp. 2300–2303). Boston, MA: Springer. Retrieved from http://www.springerlink.com/index/10.1007/978-1-4419-1428-6_589
- Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *Journal of Media Literacy Education*, 4(2), 121–135.
- Century, J., Lach, M., King, H., Rand, S., Heppner, C., Franke, B., & Westrick, J. (2013). *Building an operating system for computer science*. Chicago, IL: CEMSE, University of Chicago with UEI. Retrieved from <http://outlier.uchicago.edu/computerscience/OS4CS/>
- Chang, H., Quintana, C., & Krajcik, J. S. (2009). The impact of designing and evaluating molecular animations on how well middle school students understand the particulate nature of matter. *Science Education*, 94(1), 73–94. <https://doi.org/10.1002/sce.20352>

- Cheng, M., Lin, J., Chang, Y., Li, H., Wu, T., & Lin, D. (2014). Developing explanatory models of magnetic phenomena through model-based inquiry. *Journal of Baltic Science Education, 13*(3), 351-360.
- Clement, J. (2000). Model based learning as a key research area for science education. *International Journal of Science Education, 22*(9), 1041–1053.
<http://dx.doi.org/10.1080/095006900416901>
- Clement, J. J. (2008). *Creative model construction in scientists and students: The role of imagery, analogy, and mental simulation*. Dordrecht, NL: Springer.
- Clement, J. J., & Rea-Ramirez, M. A. (2008). *Model-based learning and instruction in science*. Dordrecht, Netherlands: Springer.
- Cook, D. A., Hamstra, S. J., Brydges, R., Zendejas, B., Szostek, J. H., Wang, A. T., ... Hatala, R. (2013). Comparative effectiveness of instructional design features in simulation-based education: Systematic review and meta-analysis. *Medical Teacher, 35*(1), e867–e898.
<http://dx.doi.org/10.3109/0142159X.2012.714886>
- Cook, M. P. (2006). Visual representations in science education: The influence of prior knowledge and cognitive load theory on instructional design principles. *Science Education, 90*(6), 1073–1091. <https://doi.org/10.1002/sce.20164>
- Craik, K. J. W. (1943). *The nature of explanation*. Cambridge: Cambridge University Press.
- D'Angelo, C., Rutstein, D., Harris, C., Haertel, G., Bernard, R., & Borokhovski, E. (2013). *Review of computer-based simulations for STEM learning in K-12 education*. Menlo Park, CA: SRI International.

- Dass, K., Head, M. L., & Rushton, G. T. (2015). Building an understanding of how model-based inquiry is implemented in the high school chemistry classroom. *Journal of Chemical Education*, 92(8), 1306–1314. <http://doi.dx.org/10.1021/acs.jchemed.5b00191>
- Duschl, R. A., Schweingruber, H. A., & Shouse, A. W. (Eds.). (2007). *Taking science to school: Learning and teaching science in grades K-8*. Washington, DC: National Academies Press.
- Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation is critical. *ACM Inroads*, 1(1), 61–66.
- Giere, R. N. (2004). How models are used to represent reality. *Philosophy of Science*, 71(1), 742–752.
- Gilbert, J. K. (1991). Model building and a definition of science. *Journal of Research in Science Teaching*, 28(1), 73–79.
- Gilbert, J. K., & Boulter, C. J. (2000). *Developing models in science education*. Boston, MA: Kluwer Academic Publishers.
- Girvan, C., Tangney, B., & Savage, T. (2013). SLurtles: Supporting constructionist learning in Second Life, 61(1), 115–132. <https://doi.org/10.1016/j.compedu.2012.08.005>
- Google, & Gallup. (2015b). *Images of computer science: Perceptions among students, parents and educators in the U.S.* Retrieved from <http://g.co/cseduresearch>
- Google, & Gallup. (2015a). *Searching for computer science: Access and barriers in U.S. K-12 education*. Retrieved from <http://g.co/cseduresearch>
- Gouvea, J., & Passmore, C. (2017). ‘Models of’ versus ‘models for’: Toward an agent-based conception of modeling in the science classroom. *Science & Education*, 26(1–2), 49–63. <https://doi.org/10.1007/s11191-017-9884-4>

- Grosslight, L., Unger, C., Jay, E., & Smith, C. L. (1991). Understanding models and their use in science: Conceptions of middle and high school students and experts. *Journal of Research in Science Teaching*, 28(1), 73–79. <http://dx.doi.org/10.1002/tea.3660280907>
- Hannafin, M. J., Hill, J., & McCarthy, J. (2001). Designing resource-based learning and performance support systems. In D. Wiley (Ed.), *Learning objects* (pp. 99–130). Bloomington, IN: AECT.
- Harel, I., & Papert, S. (1991). *Constructionism: Research reports and essays, 1985-1990*. Norwood, NJ: Ablex Publishing.
- Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education*, 22(9), 1011–1026. <http://dx.doi.org/10.1080/095006900416884>
- Harvey, B., & Mönig, J. (2010). Bringing “no ceiling” to Scratch: Can one language serve kids and computer scientists. *Proceedings of Constructionism*, 1–10. Retrieved from: <https://snap.berkeley.edu/BYOB.pdf>
- Henderson, L., & Tallman, J. (2006). *Stimulated recall and mental models*. Lanham, MD: Scarecrow Press Inc.
- Ioannidou, A., Repenning, A., Keyser, D., Luhn, L., & Daetwyler, C. (2010). Mr. Vetro: A collective simulation for teaching health science. *International Journal of Computer-Supported Collaborative Learning*, 5(2), 141–166. <https://doi.org/10.1007/s11412-010-9082-8>
- Justi, R. S., & Gilbert, J. K. (2002). Science teachers’ knowledge about and attitudes towards the use of models and modelling in learning science. *International Journal of Science Education*, 24(12), 1273–1292. <https://dx.doi.org/10.1080/09500690210163198>

- K-12 Computer Science Framework. (2016). Retrieved from <http://www.k12cs.org>
- Kafai, Y. (2014). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 35–46). New York, NY: Cambridge University Press.
- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, *46*(5), 767–787.
<https://doi.org/10.1007/s11251-018-9453-5>
- Kim, E., Oliver, J. S., & Jackson, D. F. (2016). Connecting the imperatives of STEM, NGSS, deep learning and assessment: A conceptual paper. In *Proceedings of the National Association for Research in Science Teaching*. Baltimore, MD.
- Klahr, D., Triona, L. M., & Williams, C. (2007). Hands on what? The relative effectiveness of physical versus virtual materials in an engineering design project by middle school children. *Journal of Research in Science Teaching*, *44*(1), 183–203.
<https://doi.org/10.1002/tea.20152>
- Krajcik, J., & Merritt, J. (2012). Engaging students in scientific practices: What does constructing and revising models look like in the science classroom? *The Science Teacher*, *79*(3), 38–41.
- Lehrer, R., & Schauble, L. (2000). Developing model-based reasoning in mathematics and science. *Journal of Applied Developmental Psychology*, *21*(1), 39–48.
[http://dx.doi.org/10.1016/S0193-3973\(99\)00049-0](http://dx.doi.org/10.1016/S0193-3973(99)00049-0)
- Li, F. W. B., & Watson, C. (2011). Game-based concept visualization for learning programming. In *Proceedings of the Third International ACM Workshop on Multimedia Technologies for Distance Learning* (pp. 37–42). Scottsdale, AZ: ACM.

- Linn, M. C. (2000). Designing the knowledge integration environment. *International Journal of Science Education*, 22(8), 781–796.
- Louca, L., & Constantinou, C. (2003). The use of computer-based microworlds for developing modeling skills in physical science: An example from light. *International Journal of Science Education*. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.510.536>
- Lowe, R. (2004). Interrogation of a dynamic visualization during learning. *Learning and Instruction*, 14(3), 257–274. <https://dx.doi.org/10.1016/j.learninstruc.2004.06.003>
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. <https://doi.org/10.1016/j.chb.2014.09.012>
- Malan, D. J., & Leitner, H. H. (2007). Scratch for budding computer scientists (Vol. 39, pp. 223–227). Presented at the ACM SIGCSE'07, Convington, KY.
- Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, 10(4), 1–15. <https://doi.org/10.1145/1868358.1868363>
- Marbach-Ad, G., Rotbain, Y., & Stavy, R. (2008). Using computer animation and illustration activities to improve high school students' achievement in molecular genetics. *Journal of Research in Science Teaching*, 45(3), 273–292. <https://doi.org/10.1002/tea.20222>
- Marshall, J. A., & Young, E. S. (2006). Preservice teachers' theory development in physical and simulated environments. *Journal of Research in Science Teaching*, 43(9), 907–937.
- National Research Council. (1996). *National science education standards: Observe, interact, change, learn*. Washington, DC: National Academies Press.

- National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC: The National Academies Press.
- Nelson, M. M., & Davis, E. A. (2012). Preservice elementary teachers' evaluations of elementary students' scientific models: An aspect of pedagogical content knowledge for scientific modeling. *International Journal of Science Education, 34*(12), 1931–1959.
<http://dx.doi.org/10.1080/09500693.2011.594103>
- Nersessian, N. J. (2008). *Creating scientific concepts*. Cambridge, MA: MIT Press.
- Osborne, J. (2014). Teaching scientific practices: Meeting the challenge of change. *Journal of Science Teacher Education, 25*(2), 177–196. <https://doi.org/10.1007/s10972-014-9384-1>
- Papaevripidou, M., Constantinou, C. P., & Zacharia, Z. C. (2007). Modeling complex marine ecosystems: An investigation of two teaching approaches with fifth graders. *Journal of Computer Assisted Learning, 23*(2), 145–157. <https://doi.org/10.1111/j.1365-2729.2006.00217.x>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Pea, R. D. (2004). The social and technological dimensions of scaffolding and related theoretical concepts for learning, education, and human activity. *The Journal of the Learning Sciences, 13*(3), 423–451. http://dx.doi.org/10.1207/s15327809jls1303_6.
- Price, T. W., & Barnes, T. (2015). Comparing textual and block interfaces in a novice programming environment. In *Proceedings of the Eleventh Annual International Computing Education Research* (pp. 91–99). Omaha, NE: ACM Press.
<https://doi.org/10.1145/2787622.2787712>

- Reiser, B. J. (2002). Why scaffolding should sometimes make tasks more difficult for learners. In *Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community* (pp. 255–264). International Society of the Learning Sciences. Retrieved from <http://dl.acm.org/citation.cfm?id=1658652>
- Reiser, B. J. (2004). Scaffolding complex learning: The mechanisms of structuring and problematizing student work. *The Journal of the Learning Sciences*, *13*(3), 273–304.
- Renken, M., Peffer, M., Otrell-Cass, K., Girault, I., & Chiocciariello, A. (2016). *Simulations as scaffolds in science education*. Cham: Springer.
- Rosenblueth, A., & Wiener, N. (1945). The role of models in science. *Philosophy of Science*, *12*(4), 316–321. <https://doi.org/10.1086/286874>
- Samarapungavan, A., Tippins, D., & Bryan, L. (2015). A modeling-based inquiry framework for early childhood science learning. In K. C. Trundle & M. Saçkes (Eds.), *Research in Early Childhood Science Education* (pp. 259–277). Dordrecht, Netherlands: Springer. https://doi.org/10.1007/978-94-017-9505-0_12
- Scalise, K., Timms, M., Moorjani, A., Clark, L., Holtermann, K., & Irvin, P. S. (2011). Student learning in science simulations: Design features that promote learning gains. *Journal of Research in Science Teaching*, *48*(9), 1050–1078. <http://dx.doi.org/10.1002/tea.20437>
- Schwarz, C. V., & Gwekwerere, Y. N. (2007). Using a guided inquiry and modeling instructional framework (EIMA) to support pre-service K-8 science teaching. *Science Education*, *91*(1), 158–186.
- Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., ... Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific

- modeling accessible and meaningful for learners. *Journal of Research in Science Teaching*, 46(6), 632–654. <https://doi.org/10.1002/tea.20311>
- Schwarz, C. V., & White, B. Y. (2005). Metamodeling knowledge: Developing students' understanding of scientific modeling. *Cognition and Instruction*, 23(2), 165–205. http://dx.doi.org/10.1207/s1532690xci2302_1
- Seel, N. M. (2014). Model-based learning and performance. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (pp. 465–484). New York, NY: Springer New York.
- Seel, N. M. (2017). Model-based learning: A synthesis of theory and research. *Educational Technology Research and Development*, 65(4), 931–966. <https://doi.org/10.1007/s11423-016-9507-9>
- 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>
- Shen, J., Lei, J., Chang, H., & Namdar, B. (2014). Technology-enhanced, modeling-based instruction (TMBI) in science education. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (pp. 529–540). New York, NY: Springer New York. https://doi.org/10.1007/978-1-4614-3185-5_41
- Sins, P. H. M., Savelsbergh, E. R., van Joolingen, W. R., & van Hout-Wolters, B. H. A. M. (2009). The relation between students' epistemological understanding of computer

- models and their cognitive processing on a modelling task. *International Journal of Science Education*, 31(9), 1205–1229. <http://dx.doi.org/10.1080/09500690802192181>
- Smetana, L. K., & Bell, R. L. (2012). Computer simulations to support science instruction and learning: A critical review of the literature. *International Journal of Science Education*, 34(9), 1337–1370. <http://doi.doi.org/10.1080/09500693.2011.605182>
- van de Pol, J., Volman, M., & Beishuizen, J. (2010). Scaffolding in teacher–student interaction: A decade of research. *Educational Psychology Review*, 22(3), 271–296. <https://doi.org/10.1007/s10648-010-9127-6>
- Van Hook, S. J., Huziak-Clark, T. L., Nurnberger-Haag, J., & Ballone-Duran, L. (2009). Developing an understanding of inquiry by teachers and graduate student scientists through a collaborative professional development program. *Electronic Journal of Science Education*, 13(2), 30–61.
- van Joolingen, W. R., Aukes, A. V. A., Gijlers, H., & Bollen, L. (2015). Understanding Elementary Astronomy by Making Drawing-Based Models. *Journal of Science Education and Technology*, 24(2–3), 256–264. <http://dx.doi.org/10.1007/s10956-014-9540-6>
- Van Zee, E. H., Iwasyk, M., Kurose, A., Simpson, D., & Wild, J. (2001). Student and teacher questioning during conversations about science. *Journal of Research in Science Teaching*, 38(2), 159–190.
- Wagh, A., & Wilensky, U. (2012). Evolution in blocks: Building models of evolution using blocks. In *Proceedings from Constructionism: Theory, Practice, and Impact*. Athens, Greece. Retrieved from http://www.aditiwagh.org/files/publications/WaghWilensky2012_Constructionism.pdf

- Weintrop, D. (2015). Blocks, text, and the space between: The role of representations in novice programming environments. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 301–302). Atlanta, GA: IEEE.
- White, B., & Schwarz, C. (1999). Alternative approaches to using modeling and simulation tools for teaching science. In P. Ramsden (Ed.), *Computer modeling and simulation in science education*. New York: Springer-Verlag.
- White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118.
https://doi.org/10.1207/s1532690xci1601_2
- Wieman, C. E., Adams, W. K., & Perkins, K. K. (2008). PhET: Simulations that enhance learning. *Science, 322*(5902), 682–683. <http://dx.doi.org/10.1126/science.1161948>
- Wilensky, U., & Rand, W. (2009). *An introduction to agent-based modeling: Modeling natural, social, and engineered complex systems with NetLogo*. Cambridge, MA: MIT Press.
- Windschitl, M., Thompson, J., & Braaten, M. (2008). Beyond the scientific method: Model-based inquiry as a new paradigm of preference for school science investigations. *Science Education, 92*(5), 941–967. <http://doi.doi.org/10.1002/sce.20259>
- Winn, W., Stahr, F., Sarason, C., Fruland, R., Oppenheimer, P., & Lee, Y. (2006). Learning oceanography from a computer simulation compared with direct experience at sea. *Journal of Research in Science Teaching, 43*(1), 25–42. <https://doi.org/10.1002/tea.20097>
- Wojnowski, B. S., & Pea, C. H. (2014). *Models and approaches to STEM professional development*. Arlington, VA: National Science Teachers Association.

- Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100. <https://doi.org/10.1111/j.1469-7610.1976.tb00381.x>
- Wu, H., Krajcik, J. S., & Soloway, E. (2001). Promoting conceptual understanding of chemical representations: Student's use of a visual tool in the classroom. *Journal of Research in Science Teaching*, 38(7), 821–842.
- Wu, Y., McLean, J. E., & Powerful, M. (2001). Promoting conceptual understanding of chemical representations: Students' use of a visualization tool in the classroom. *Journal of Research in Science Teaching*, 38(7), 821–842. <http://dx.doi.org/10.1002/tea.1033>
- Xie, C., Tinker, R., Tinker, B., Pallant, A., Damelin, D., & Berenfeld, B. (2011). Computational experiments for science education, 332(6037), 1516–1517. <http://dx.doi.org/10.1126/science.1197314>
- Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education*, 26(4), 235–254. <http://dx.doi.org/10.1080/08993408.2016.1257418>

CHAPTER 2

CODING IN SCIENTIFIC MODELING LESSONS (CS-Model)¹

¹ Vasconcelos, L., & Kim, C. Submitted to *Educational Technology Research and Development*, 11/13/2018.

Abstract

Learning standards for K-12 science education emphasize the importance of engaging students in practices that scientists perform in their profession. Such a practice-oriented approach challenges the long-standing teaching methods of model memorization and regurgitation. K-12 teachers are expected to engage students in scientific modeling, which entails constructing, testing, evaluating, and revising their own models of science phenomena while pursuing an epistemic goal. However, conceptualizing scientific models that often involve unobservable science phenomena is daunting for students. One way to support students' scientific modeling is through use of block-based coding to generate simulations that serve as artifacts for self-expression and reasoning about target science phenomena. However, preservice and in-service science teachers often hold a deficient understanding of scientific modeling instruction and lack experience teaching with coding. Professional learning on use of block-based coding in scientific modeling instruction is needed. Based on pertinent literature, five guidelines are proposed to inform teacher educators striving to offer such professional learning experiences. The guidelines informed the design and development of Coding in Scientific Modeling Lessons (CS-Model), which is a module and an online tool for scaffolding teachers' learning to use block-based coding to create science simulations, and integrate simulation coding activities into scientific modeling lessons, respectively.

Keywords: scientific modeling, epistemic agency, block-based coding, simulations, STEM learning

Introduction

K-12 students are to be offered authentic science learning experiences that mirror what scientists do in reality (National Research Council (NRC), 2012; NGSS Lead States, 2013). Students need to perform *practices* that reflect not only scientists' procedural tasks but also the cognitive challenges they experience when attempting to make sense of and advance knowledge about phenomena in the world (Osborne, 2014; Passmore, Gouvea, & Giere, 2014). Such a practice-oriented view of science education challenges the long-standing culture of passive science learning through memorization and regurgitation of models (Duschl, 2008; Gouvea, Passmore, & Jamshidi, 2014; Schwarz et al., 2009) imposed by teachers and textbooks. K-12 students are to be addressed as *epistemic agents*, i.e., individuals who actively take part in the scientific enterprise by constructing knowledge themselves (Berland et al., 2016; Gouvea & Passmore, 2017; Knuuttila, 2011; Stroupe, 2014), which entails not only learning about science but also doing science (Hodson, 2014; NRC, 2012; Osborne, 2014).

One way to promote students' epistemic agency is by helping them create, manipulate, and develop their own models to better understand and advance their thinking about science phenomena (Berland et al., 2016; Gouvea & Passmore, 2017; NRC, 2012; Samarapungavan, Tippins, & Bryan, 2015). Developing hypotheses and models to explain the surrounding world is an innate trait of human cognition (Giere, 1988; Passmore, Schwarz, & Mankowski, 2016) that intuitively starts from a young age (Clement, 2000; Seel, 2014). Different from day-to-day intuitive sensemaking, school science education is to promote students' *systematic* construction, testing, evaluation, and revision of their own ideas and hypotheses and support development of models that are theoretically and empirically robust (Chu, Deurmeyer, & Quek, 2017; Gouvea & Passmore, 2017; Knuuttila, 2005b, 2011; NRC, 2012; Schwarz et al., 2009; Schwarz & White,

2005). Such a systematic process of model construction and development is called scientific modeling (Cheng et al., 2014; Hokayem & Schwarz, 2014; Kim & Oliver, 2018; Nelson & Davis, 2012; Samarapungavan et al., 2015; Schwarz et al., 2009).

Scientific modeling is daunting for K-12 students who are expected to create and develop more sophisticated models (Hokayem & Schwarz, 2014; Kim & Oliver, 2018; Samarapungavan et al., 2015) of phenomena that often involve microscopic or macroscopic elements and processes (Barak, Ashkar, & Dori, 2011; Cheng et al., 2014; Cook, 2006; Gilbert, 2008). Students struggle to conceptualize observable evidence onto abstract models (Barak et al., 2011; Harrison & Treagust, 2000; Shen, Lei, Chang, & Namdar, 2014; Wouters, Paas, & van Merriënboer, 2008), such as creating and refining a model of electromagnetic fields based on observed interactions among charged particles. Students need support to successfully create and develop their models.

One way to support K-12 students' scientific modeling is through block-based coding (also known as block-based programming). Block-based coding involves selecting, dragging, dropping, and connecting blocks that embody programming commands to create an animated behavior on the screen (Baratè, Ludovico, Mangione, & Rosa, 2015; Burke, 2012; Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010; Resnick et al., 2009). Block-based coding serves as a self-expression tool that students can use to create self-made artifacts (Holbert & Wilensky, 2014, 2018; Merrill, 2017; Papert, 1980) that materialize and simulate their own models of a target phenomenon (Wagh & Wilensky, 2012; Wilensky & Rand, 2009). Coding simulations of science phenomena is a promising approach to promote students' epistemic agency through authentic tasks such as (a) simplifying complex systems in simulations, (b) expressing ideas about non-observable phenomena, (c) creating and testing hypotheses, (d) observing and

analyzing patterns of behavior, (e) explaining concepts to peers, and (f) externalizing and advancing models using simulations as conceptual tools (Berland et al., 2016; Giere, 2006; Kim & Oliver, 2018; Knuuttila, 2011; NRC, 2012). However, integrating simulation coding activities into scientific modeling instruction is challenging for K-12 science teachers who possess limited content knowledge and pedagogical content knowledge of scientific modeling and teaching with coding.

This research question guided this chapter: *How can teacher educators scaffold science teachers' learning to code and to integrate coding into scientific modeling lessons?* The following section defines scientific models and modeling from a pragmatic perspective. Then research findings on K-12 teachers' experiences with scientific modeling and coding are presented. The subsequent section presents a definition of instructional scaffolding and a discussion on findings of empirical studies on the use of simulations as visual scaffolds for scientific modeling. Next, a framework for integrating simulation coding into scientific modeling is presented along with guidelines for teacher educators striving to design professional learning for K-12 science teachers. The last section presents Coding in Scientific Modeling Lessons (CS-Model), an instructional module and online tool that was designed and developed based on proposed guidelines.

Scientific Models and Modeling: A Pragmatic Perspective

Scientific models and modeling have been studied in several disciplinary fields including but not restricted to information science, instructional psychology, mathematics, and physics. This chapter discusses scientific models and modeling within the field of science education and adopts a pragmatic theoretical perspective. Such a perspective is aligned with the *science-as-practice* shift in K-12 science education, which emphasizes that students need opportunities to

design, evaluate, and re-design their own models of a phenomenon to fulfill intended epistemic goals (Gilbert & Justi, 2016; Gouvea & Passmore, 2017; Knuuttila, 2005, 2011; Mahr, 2012).

For a review of other theoretical, philosophical, and epistemological accounts of models and modeling, see Gilbert and Justi (2016).

Scientific Models

Philosophers of science have considered scientific models as representational tools that simplify and depict complex phenomena in the world (Morrison & Morgan, 1999; Suárez, 1999). This definition accounts for a dual model-referent relationship and emphasizes representational accuracy as an underlying principle to guide model creation and development. However, this definition does not reflect what motivates real scientists to create and use models: an epistemic goal that involves better understanding and/or generating knowledge about a target phenomenon (Berland et al., 2016; Gouvea & Passmore, 2017; Passmore et al., 2016; Stroupe, 2014).

In response, philosophers of science have adopted a pragmatic perspective on models and modeling to account for a triad model-referent-agent relationship. Models are defined as *epistemic tools* that an epistemic agent designs, manipulates, and re-designs to represent a phenomenon and pursue specific epistemic goals (Chakravartty, 2010; Giere, 2004; Gouvea & Passmore, 2017; Knuuttila, 2005, 2011; Mahr, 2012; Passmore et al., 2014). Likewise, it is important to consider scientific models not only as *models of* a referent but also as *models for* a specific purpose (Giere, 2004, 2006; Gilbert & Justi, 2016; Gouvea & Passmore, 2017; Knuuttila, 2005, 2011).

Scientific models are context dependent. An epistemic agent can design and re-design alternative models of the same referent (Buckley, 2012; Harrison & Treagust, 2000; National Research Council, 2012; Schwarz et al., 2009; Schwarz & White, 2005) to investigate different

aspects of the same phenomenon based on specific epistemic goals (Gouvea & Passmore, 2017; Passmore et al., 2016). Scientific models are only *approximations* (Gilbert, 1991; Harrison & Treagust, 2000; Krajcik & Merritt, 2012) that allow “indirect representation and analysis” (Knuuttila, 2011, p. 266) of a system’s key properties. For example, computer-based hurricane simulations are dynamic models that mimic and help scientists predict select features such as wind speed and storm surge though they do not determine the *exact* hurricane path.

Scientific Modeling

Scientific modeling is a creative and generative process that entails constructing and manipulating scientific models to advance knowledge about specific aspects of a system or phenomenon (Knuuttila, 2005a, 2011; Knuuttila & Boon, 2011; NRC, 2012; Nelson & Davis, 2012). Contrary to day-to-day sensemaking, in which an epistemic agent *intuitively* creates hypotheses to explain phenomena in world, scientific modeling involves *systematically* performing scientific practices to solve complex science problems (Lehrer & Shauble, 2006; Louca & Zacharia, 2012; Nelson & Davis, 2012; Osborne, 2017; Windschitl, Thompson, & Braaten, 2008). Numerous frameworks attempt to describe such scientific modeling practices. While a comprehensive review is beyond the scope of this chapter, it is notable that four major scientific modeling practices arise from the literature: construct, test, evaluate, and revise models (Coll, France, & Taylor, 2005; Hokayem & Schwarz, 2014; Nelson & Davis, 2012; Schwarz et al., 2009).

To construct a model, an epistemic agent selects specific elements, features, and/or interactions (Harrison & Treagust, 2000; Knuuttila & Boon, 2011; Krajcik & Merritt, 2012; NRC, 2012; Passmore et al., 2016) from a target phenomenon or system. Only information that is relevant to the agent’s epistemic goals is included in the model (Osborne, 2014; Windschitl,

2001; Windschitl, Thompson, & Braaten, 2008). For instance, climatologists work with computer programmers to select key data from atmospheric observations and use computer code to create simulations that show dynamic models of hurricanes and storms (Meyer, Broad, Orlove, & Petrovic, 2013; National Aeronautics and Space Administration (NASA), 2016).

Subsequently, experiments are designed and carried out to test a model, which materializes one's own hypotheses and ideas about the phenomenon (Berland et al., 2016; Osborne, 2014; Passmore et al., 2016). Testing involves designing experiments to produce and collect empirical evidence that is pertinent to one's epistemic goals (Hernández, Couso, & Pintó, 2015; Kim & Oliver, 2018; Schwarz & White, 2005; White & Frederiksen, 1998). Using the hurricane simulation as an example, one can run multiple iterations of computer-based models and observe how hurricane trajectory changes over time.

Then one evaluates their model and initial hypotheses by comparing them to collected empirical evidence and pertinent theories. One can use criteria such as accuracy, explanatory power, and utility to determine the extent to which a model is helpful in fulfilling their epistemic goals (Gilbert & Justi, 2016; Knuuttila & Boon, 2011). Such an evaluation process helps one identify inconsistencies in the model as well as generate theoretically and empirically robust explanations about how and why a phenomenon occurs (Gouvea & Passmore, 2017; Knuuttila, 2005a). If the model is not deemed satisfactory based on theories, data, and epistemic purposes, one will need to revise the model and iteratively perform scientific modeling practices of model construction, testing, evaluation, and revision to generate more sophisticated models (Buckley, 2012; Grosslight, Unger, Jay, & Smith, 1991; Kim & Oliver, 2018; Samarapungavan et al., 2015).

Engaging K-12 students in authentic cognitive challenges and scientific modeling practices that real scientists experience brings numerous benefits to science learning. For instance, it helps students (a) better understand science concepts and authentic scientific inquiry practices (Bamberger & Davis, 2013; Bau et al., 2017; Kim & Oliver, 2018; Namdar & Shen, 2015; NRC, 2012), (b) perceive the flexible and dynamic nature of models as context-dependent tools that help fulfill an agent's goals (Grosslight et al., 1991; Schwarz et al., 2009; Schwarz & White, 2005), (c) develop an accurate understanding of science as a disciplinary field (Halloun, 2006; Hokayem & Schwarz, 2014; Windschitl et al., 2012), (d) recognize their place in a community whose members share similar epistemic aims and adopt socially negotiated scientific practices (Ahlstroms, 2010; Damsa, Kirschner, Andriessen, Erkens, & Sins, 2010; Stroupe, 2014), and (e) impact students' decision to pursue STEM career paths.

Science teachers and scientific modeling.

Preservice and in-service K-12 teachers often hold an inaccurate understanding of models and scientific modeling (Akerson et al., 2009; Harrison & Treagust, 2000; Justi & Gilbert, 2002; Kenyon, Davis, & Hug, 2011; Krell & Krüger, 2016; Reinisch & Krüger, 2018; Windschitl & Thompson, 2004; Windschitl et al., 2008), which negatively affects their teaching. For example, teachers often perceive models as mere representations of a target phenomenon rather than as investigative tools (Danusso, Tesla, & Vicentini, 2010; Krell & Krüger, 2016; Lin, 2014). Consequently, they use models from textbooks solely to illustrate and explain science concepts (Krell, Upmeier zu Belzen, & Krüger, 2012). A common misconception among teachers is that models from textbooks are *the only correct answer* rather than an alternative form of representation (Abell & Roth, 1995; Gilbert, 1991; Harrison & Treagust, 2000; Hokayem & Schwarz, 2014). As a result, teachers fail to engage students in evaluating the strengths and

limitations of alternative models based on specific purposes (Harrison & Treagust, 2000) and raise students' awareness to the flexible nature of models (Grosslight et al., 1991; White & Schwarz, 1999).

Teachers' content knowledge and pedagogical content knowledge affect the quality and types of activities integrated into their teaching (Driel & Verloop, 1999). For instance, teachers exposed to rote memorization and factual repetition end up employing such techniques in their lessons (Momsen, Long, Wyse, & Ebert-May, 2010). In scientific modeling instruction, teachers expect students to recall imposed models to demonstrate learning *about* science concepts and facts (Buckley, 2012; Gilbert & Boulter, 2000; Horikoshi, 2015; Nassiff & Czerwinski, 2014; Schwarz & Gwekwerere, 2007). Such teaching thwarts opportunities for students to engage in authentic scientific modeling practices. Offering professional learning on scientific modeling to preservice and in-service science teachers (Crawford & Cullin, 2004; Dass et al., 2015; Kim & Oliver, 2018; NRC, 2012) to help them develop both content knowledge and pedagogical content knowledge (Weiss & Pasley, 2006) is critical. During professional learning, teachers need opportunities to experience scientific modeling from a student's perspective as well as examine instructional methods and learning materials on scientific modeling (Stammen, Malone, & Irving, 2018) from an educator's perspective.

Instructional Scaffolding

Instructional scaffolding temporarily supports accomplishment of tasks that are beyond one's unassisted skills so that such skills can be applied in the future without assistance (Belland, 2011, 2014, 2017; Kim & Hannafin, 2011; Wood et al., 1976). Scaffolds provide support *during* task engagement (Belland, 2014; Hannafin, Hill, & McCarthy, 2001; Wood et al., 1976), build upon one's prior knowledge, continuously assess learning progress (van de Pol, Volman, &

Beishuizen, 2010; Wood et al., 1976), and provide structure for task accomplishment while highlighting complex concepts or processes pertinent to the task (Reiser, 2002, 2004).

Scaffolding entails controlling the amount of assistance offered for skill development (Belland, 2017; Seethaler, Fuchs, Fuchs, & Compton, 2012; van de Pol et al., 2010; Wood et al., 1976), which involves adding or fading scaffolding strategies (Collins, Brown, & Holum, 1991; Collins, Brown, & Newman, 1989; Koedinger & Corbett, 2006), directing one's efforts to target skills, and determining *when* assistance is needed (Wood, 2003). Scaffolding is expected to fade over time so that learners take responsibility for independent task accomplishment (Belland, 2014; Collins et al., 1989; Collins et al., 1991; Hannafin & Hill, 2008).

Simulations as Scaffolds for Scientific Modeling

Simulations are models that dynamically represent elements, features, and functions of phenomena in the real world (Bowen & Deluca, 2015; de Jong & van Joolingen, 1998; Smetana & Bell, 2012). Simulations enable one to visualize and investigate a phenomenon (Krajcik & Merritt, 2012; NRC, 2012) that is complex and/or dangerous for direct exposure (Akpan, 2002; Buckley, 2000; Cook, 2006; Smetana & Bell, 2012). Computer-based simulation tools have been widely used as visual scaffolds for scientific modeling. Students use such simulations to test hypotheses about a phenomenon by manipulating rules or parameters and observing outcomes (Bowen & Deluca, 2015; Rutten, van Joolingen, & van der Veen, 2012; Windschitl, 2001). Simulations allow design and redesign of experiments (Bell & Smetana, 2008; Bowen & Deluca, 2015; Smetana & Bell, 2012) while serving as a medium for visualizing unobservable phenomena (Barak et al., 2011; Eichinger, Nakhleh, & Auberry, 2000; Tao & Gunstone, 1999).

However, research reviews that analyze the effectiveness of using computer simulations for science learning found mixed results (Scalise et al., 2011; Smetana & Bell, 2012). Some

studies show positive learning gains from using simulations (Kumar & Sherwood, 2007; Stern, Barnea, & Shauli, 2008; Wu, Krajcik, & Soloway, 2001; Zacharia, 2003) although other studies found no significant difference between experimental groups exposed to simulations and control groups (Klahr, Triona, & Williams, 2007; Marbach-Ad, Rotbain, & Stavy, 2008; Marshall & Young, 2006; Winberg & Berg, 2007; Winn et al., 2006). Such contradictory results are partly due to methodological and pedagogical issues such as use of simulations (a) without high-quality scaffolding strategies, (b) dissociated of self-driven scientific inquiry, (c) with complicated or uninteresting features, and (d) as replacements rather than supplements to other teaching methods (D'Angelo et al., 2014; Seel, 2017; Shen et al., 2014; Smetana & Bell, 2012; Trundle & Bell, 2010; Wu et al., 2001; Zacharia, 2003). To enhance the effectiveness of simulations as scaffolds, it is critical to train teachers to design instructional strategies and support structures that will meaningfully aid students' engagement with simulations in scientific modeling instruction.

Block-based Code Simulations as Scaffolds for Scientific Modeling

Block-based coding is a programming language created to make computer science more accessible to novice learners, such as K-12 students (K-12 Computer Science Framework Steering Committee, 2016; Lye & Koh, 2014; Wing, 2006; Yadav, Gretter, Hambruch, & Sands, 2016). Block-based coding tools (a) engage users in recognizing blocks from a palette rather than memorizing text-based programming vocabulary; (b) contain a concise number of meaningful coding units, which reduces complexity and cognitive load; (c) prevent errors as incompatible blocks cannot be snapped together; and (d) offer immediate visual feedback on created simulations through code execution (Basu et al., 2017; Maloney, Peppler, Kafai, Resnick, & Rusk, 2008; Resnick et al., 2009; Weintrop & Wilensky, 2015).

Pedagogical affordances of simulation coding for scientific modeling.

Using block-based coding to support scientific modeling is a promising approach. First, it embodies constructivist and constructionist learning principles as one engages in active learning activities that involve creating, manipulating, and reasoning with an artifact that externalizes their own scientific models (Girvan, Tangney, & Savage, 2013; Harel & Papert, 1991; Pellas & Peroutseas, 2016; Sengupta, Farris, & Wright, 2012; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013). This approach is aligned with the science-as-practice shift in science education, which reinforces the need to engage students in expressing and creating their own models of science phenomena (Gouvea & Passmore, 2017; NRC, 2008; Osborne, 2014; Sengupta et al., 2013).

Second, block-based coding is a visual programming language, i.e., it provides visual feedback on code execution (Hundhausen & Brown, 2007; Kelleher & Pausch, 2005) and allows visualization of what is not observable, such as one's own scientific models, which often involve microscopic or macroscopic phenomena (Sengupta & Wilensky, 2009; Wagh & Wilensky, 2012). Visual feedback is expected to facilitate sensemaking (Schwarz et al., 2009), identification of "hidden" misconceptions (Clement, 2008), and communication about complex concepts (Bell, Gess-Newsome, & Luft, 2008).

Third, block-based coding tools (e.g., Scratch, Blockly, Alice) are domain- and topic-independent. This allows their application to interdisciplinary learning contexts wherein one employs computer science concepts and processes (e.g., coding, debugging, abstraction), as well as knowledge and skills pertinent to other fields, to solve problems. Interdisciplinary learning prepares one to be a better problem solver in the real world (Kim, Oliver, & Jackson, 2016; NGSS Lead States, 2013). Fourth, block-based coding allows fast prototyping (Grover, Pea, &

Cooper, 2015; Kehtarnavaz & Gope, 2006; Wagh & Wilensky, 2012) as one can easily assemble and reassemble coding blocks to modify or improve an animation or simulation output. In other words, block-based coding facilitates rapid construction, testing, evaluation, and revision of models and hypotheses (Soloway, 1993), which are core scientific modeling practices. Fifth, block-based coding tools have a low threshold for required programming knowledge, provide an open-ended space for exploration of alternative solutions during problem solving, and can be adapted to more complex and sophisticated coding projects (Grover et al., 2015; Repenning, Webb, & Ioannidou, 2010; Resnick et al., 2009). Thus, block-based coding is accessible to learners at various levels of expertise.

It is important to note that block-based coding tools, such as Scratch, were originally designed to teach programming to novice learners through game design. But sophisticated block-based coding tools, such as Scratch, allow creation of simulations (Maloney et al., 2010) that can be used as test beds for knowledge generation (Gelbart, Brill, & Yarden, 2009; Rutten et al., 2012), sources of data collection, and tools to support development of scientific models (Sengupta et al., 2013). For instance, one can design and code a simulation that controls how a water sprite goes through different states of matter based on slider buttons that represent factors such as temperature and pressure. Designing such a simulation entails several levels of abstraction (Sengupta et al., 2013; Wing, 2006, 2008) as one creates algorithms that embody conceptualized variables from a target science phenomenon. In other words, one expresses relationships between independent and dependent variables using science and computer science knowledge. Visual displays of data (e.g., graphs, numbers) can also be designed, coded, and integrated into the simulation to show quantifiable evidence over time (Sengupta & Wilensky, 2009; Wilensky & Reisman, 2006). Given such pedagogical affordances, it is reasonable to say

that coding one's own simulations of science phenomena is a promising approach that empowers one to enact their epistemic agency and promotes interdisciplinary STEM learning.

Framework for Simulation Coding in Scientific Modeling

This section presents a framework (Figure 2.1) that shows simulation coding as an approach to support creation and development of scientific models. Driven by an epistemic goal, an epistemic agent selects relevant features of a science phenomenon to be included in the simulation (Knuuttila, 2011; Lehrer & Schauble, 2006; Seel, 2017). Then one employs coding skills by identifying code blocks that can dynamically materialize the selected features of the target phenomenon. In other words, one connects science knowledge and coding skills to design and construct a simulation. For example, an epistemic agent who seeks to investigate solar eclipses can use the code block *repeat forever* to simulate continuous rotation of the moon around the Earth. To constructing this simulation, one expresses applies the programming concept infinite loops as well as their model about the orbit of the moon.

After constructing the simulation and code, the epistemic agent executes the code to test their own hypotheses and ideas of the science phenomenon (Biffi et al., 2016; Lehrer & Schauble, 2006; Schwarz & White, 2005). Then one evaluates if the simulation output aligns with their own model and with experiential data (if available – see dotted line in Figure 2.1). If the simulation output is deemed satisfactory, one accepts the simulation as a dynamic model of the target science phenomenon. However, a scientific model is rarely created in one attempt, and as such requires revisions (Buckley, 2012; Seel, 2014, 2017). In this case, one rejects it and revises the code, thus performing an iterative cycle (Kim & Oliver, 2018; Nelson & Davis, 2012; Schwarz et al., 2009) until the simulation satisfies one's needs. The final simulation output, as well as the reasoning process involved in creating and coding a simulation of a science

phenomenon has potential to help one further develop their own scientific models and vice-versa (see double-headed arrow in Figure 2.1).

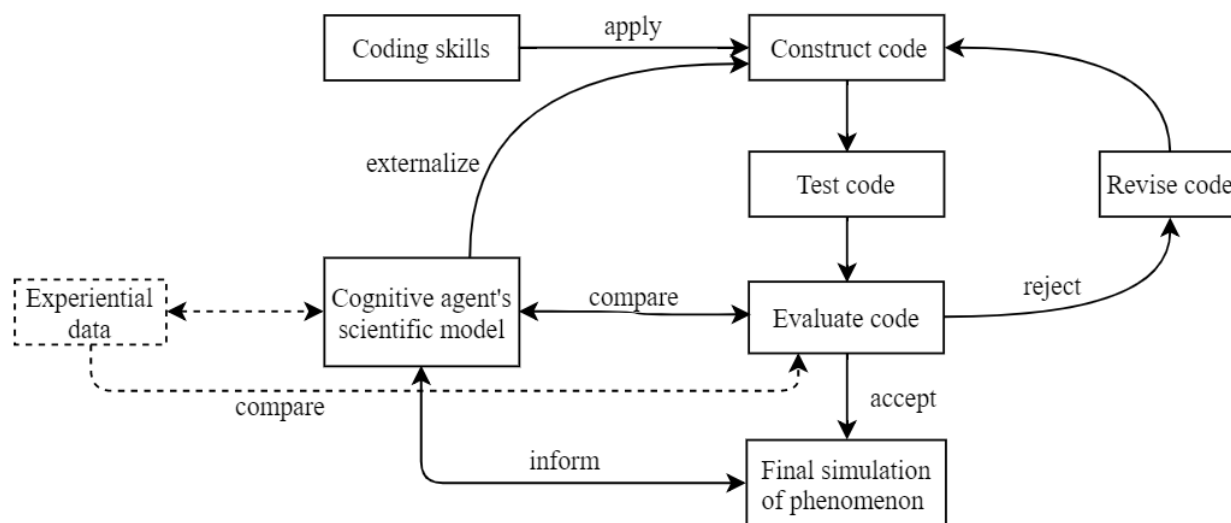


Figure 2.1. Conceptual framework for coding in scientific modeling.

Coding and Teacher Preparation

Coding is a key skill for K-12 students regardless of their future career paths (Goode, Flapan, & Margolis, 2018; K-12 Computer Science Framework Steering Committee, 2016; Obama, 2016). Teaching with coding prepares students for future problem solving in jobs that involve use of coding and computational thinking (Barr & Stephenson, 2011; Computer Science Teachers Association (CSTA) & International Society for Technology in Education (ISTE), 2011; NRC, 1999; National Science and Technology Council (NSTC), 2016; Tucker et al., 2003). Integrating coding into scientific modeling is supported by Next Generation Science Standards (NGSS Lead States, 2013), which argue that students should be exposed to learning experiences in which they apply shared concepts, ideas, and practices from Science, Technology, Engineering, and Mathematics (STEM). Research and professional learning that aim at preparing teachers to integrate coding into STEM teaching are growing in number, though few initiatives

focus on use of block-based coding in scientific modeling instruction (Project Growing Up Thinking Scientifically (GUTS), n.d.; Sengupta et al., 2013).

K-12 students often lack opportunities to learn to code before high school (Google & Gallup, 2015b). This is partly because the U.S. lacks a nation-wide computer science curriculum for K-12 schools (Paul, 2016) and for teacher preparation programs (Gal-Ezer & Stephenson, 2010). Research has found that even computer science teachers lack pedagogical knowledge of teaching with coding (Yadav et al., 2016). Hence, it is not surprising that teachers from other disciplines feel intimidated and overwhelmed by the recent urge to teach with coding. In-service and preservice teachers, regardless of their disciplinary field, need scaffolding on learning to code and to integrate coding into their lessons. It is important to emphasize that the argument in this chapter is not that coding should be used in every lesson. Teachers should have the basic knowledge, preparation, and autonomy to strategically identify subjects and topics that benefit from coding-enhanced instruction, such as scientific modeling.

Design Guidelines

This section presents guidelines for teacher educators seeking to design and develop professional learning that scaffolds science teachers' learning to code simulations of science phenomena and to integrate simulation coding into scientific modeling lessons. Each guideline is discussed and supported by literature.

Guideline 1: Combine Simulation Coding with Authentic Scientific Inquiry

Scientific modeling instruction aims to offer learning experiences that mirror scientists' authentic model creation and manipulation in service of an epistemic aim (Knuuttila, 2005b; Morrison, 2015; Morrison & Morgan, 1999; NRC, 2012). Educators striving to integrate simulation coding into scientific modeling should design activities that reflect scientists'

authentic use of simulations for visualization, data collection, pattern observation, and more. However, use of simulations in science classrooms is often restricted to manipulation of variables (Shen et al., 2014) to observe how a phenomenon occurs (Schwarz & Gwekwerere, 2007), which does not reflect the epistemic essence of scientific modeling (Schwarz et al., 2009). Research reviews indicate that use of simulations dissociated from authentic scientific inquiry is not effective (D'Angelo et al., 2014; Smetana & Bell, 2012). Likewise, several studies show that combining simulations with scientific inquiry promotes scientific reasoning and yields meaningful learning gains (Ioannidou, Repenning, Keyser, Luhn, & Daetwyler, 2010; Klahr et al., 2007; Trundle & Bell, 2010; Zacharia, 2003, 2005; Zacharia, Olympiou, & Papaevripidou, 2008).

Professional learning aiming to support teachers' learning to code a simulation while pursuing an epistemic goal should engage teachers in scientific inquiry, which entails generating research questions, designing experiments, developing hypotheses, collecting data, evaluating results, revising theory, and more (Akerson et al., 2009; NRC, 1996; Rutten et al., 2012; Schwarz & White, 2005). Engaging science teachers in such activities is critical so they “develop knowledge and understanding of ... how scientists study the natural world” (NRC, 1996, p. 23). Partaking in authentic use of simulations helps K-12 teachers understand content knowledge development from a student's perspective and pedagogical implications for teaching scientific modeling with simulations from an educator's perspective (Dass et al., 2015; Stammen et al., 2018).

Guideline 2: Tailor Coding Activities to a Scientific Modeling Framework

Participating in and designing lessons that use simulation coding to support scientific modeling are daunting tasks for teachers who are not familiar with scientific modeling practices

and/or coding concepts. To reduce complexity during professional learning, teacher educators can adopt a step-by-step scientific modeling framework and tailor coding activities around its steps. Using a framework is beneficial for K-12 teachers to understand the iterative cycle of constructing, testing, evaluating, and revising models (Dass et al., 2015; Seel, 2017; Van Hook, Huziak-Clark, Nurnberger-Haag, & Ballone-Duran, 2009), especially if teachers have limited classroom experience. A scientific modeling framework serves as a scaffold that structures and problematizes (Reiser, 2002, 2004) simulation coding during scientific modeling instruction. Specifically, a framework provides a clear structure and highlights key elements that teachers need to attend to while engaging in activities as a student and designing scientific modeling instructional materials as an educator (Dass et al., 2015).

Although scientific modeling frameworks abound in the literature, four main scientific modeling steps arise: construct, evaluate, test, and revise models (Dass et al., 2015; Hokayem & Schwarz, 2014; Kim & Oliver, 2018; NRC, 2012). Teacher educators can design coding tasks and tailor them around such steps so that teachers can: (a) construct a model while assembling code blocks to generate a simulation, (b) test a model and their hypotheses by executing simulation code, (c) evaluate accuracy and adequacy of simulation output and code, and (d) revise a model by re-designing and debugging code. Engaging teachers in such activities wherein simulation coding enables them to run experiments involving science phenomena helps teachers perceive the shared concepts and processes in computer science and science (NRC, 2012; NGSS Lead States, 2013), as well as prepare them to design interdisciplinary lessons for their future students.

Guideline 3: Promote Creation and Development of Self-made Simulations

As previously mentioned, K-12 teachers often present students with computer-based simulations that have been created for them and expect students to test predictions and generate explanations about a given phenomenon based on manipulation of variables (Shen et al., 2014; Wieman, Adams, & Perkins, 2008). Such use of simulations thwarts students' epistemic agency because students do not initiate scientific modeling practices to answer self-generated research questions, fulfill their own epistemic goals, and focus on personally relevant topics.

One way to support epistemic agency is through creation and development of self-made simulations (Chang, Quintana, & Krajcik, 2010; Schwarz et al., 2009; van Joolingen, 2015) with block-based coding. Research has found that self-generated visual representations lead to a deeper understanding of a target phenomenon and development of more sophisticated scientific models (Chang, Quintana, & Krajcik, 2010; Louca & Constantinou, 2003; Schwarz, 2009; van Joolingen, 2015; Wilkerson-Jerde, Gravel, & Macrander, 2015; Wu et al., 2001). In this approach, coding serves as an "object to think with" (Papert, 1980, p. 11), that is, a physical tool that also serves as a mind tool to help one learn about their own thoughts. Using block-based coding to generate simulations is a constructivist and constructionist approach (Harel & Papert, 1991; Kafai, 2012; Papert, 1980) that helps one transpose abstract models from their mind onto concrete visual representations (Li & Watson, 2011; Pellas & Peroutseas, 2016; Wagh & Wilensky, 2012). Self-generated simulations serve as artifacts to not only externalize prior knowledge but also "systematically explore hypothetical situations, interact with a simplified version of a process or system, change the time-scale of events, and practice tasks and solve problems in a realistic environment" (Rutten et al., 2012, p. 136). It is critical to train teachers to

create and code self-made simulations so that they can support their future students in doing the same.

Guideline 4: Design Scaffolds for Engagement in Coding *and* Scientific Modeling

Instructional scaffolding is critical, effective, and recommended to support scientific modeling (Wojnowski & Pea, 2014) that involves use of science simulations (D'Angelo et al., 2014; Smetana & Bell, 2012). Given that simulation coding entails application of both science and computer science knowledge, scaffolds should be designed to support both scientific model progression (Kenyon et al., 2011; White & Frederiksen, 1998) and learning to code (Kim, Yuan, Vasconcelos, Shin, & Hill, 2018; Sengupta et al., 2013).

There is an overall consensus in the scientific modeling literature that high-quality scaffolds (e.g., reflection prompts) are critical to support sensemaking about science phenomena and development of scientific models (Azevedo, Cromley, Moos, Greene, & Winters, 2011; Buckley et al., 2004; Chang et al., 2010; D'Angelo et al., 2014; Seel, 2017; Smetana & Bell, 2012; van Joolingen, 2015). Scaffolds on scientific model progression guide (a) selection of key aspects of the phenomenon to be simulated, (b) identification of misconceptions, (c) self-regulation of one's own learning processes, and (d) reflection on the nature and purpose of models.

Scaffolds are also essential to promote mindful engagement with coding in which one purposefully constructs code and debugs errors (Kim, Yuan, Vasconcelos, Shin, & Hill, 2017; Kim et al., 2018; Lewis, 2012) based on hypothesized implications for modeling a target phenomenon (Sengupta et al., 2013). For instance, scaffolds should target (a) abstraction processes in which one connects coding blocks to key features of the science phenomenon, (b) reflection on how code parameters materialize representational accuracy, and (c) mindful use of

debugging strategies to modify code and/or fix errors to change simulation output. Teacher educators need to devise scaffolds to support teachers' engagement in simulation coding and design of scientific modeling lessons that use simulation coding.

Guideline 5: Assess Both Scientific Model Development and Coding Skills

Stemming from an interdisciplinary perspective in which simulation coding involves both science and computer science knowledge, it is critical to devise assessment methods that focus both on scientific model development and coding skills. Assessment methods are critical to identify the challenges one encounters and to determine the best way to address them. As such, teacher educators need to design and implement strategies that assess teacher's coding skills (Brennan & Resnick, 2012; Grover, Cooper, & Pea, 2014; Grover & Pea, 2013) as well as scientific model development. Thoughtful and systematic assessment that includes formative and summative methods (Brennan & Resnick, 2012) is achievable as illustrated below.

Formative assessment offers insights on students' learning processes (Piech, Sahami, Koller, Cooper, & Blikstein, 2012) and compensates for potential misleads during summative assessment (Werner, Denner, Campe, & Kawamoto, 2012). Examples of formative assessment methods that focus on both coding and scientific modeling include (a) artifact-based explanations about how coding concepts are employed (Werner et al., 2012) to simulate models, (b) use of computer science and science terminology during activities (Lemke, 1990) and in assignments, (c) peer feedback on coding (Grover et al., 2014) and scientific modeling, and (d) documented code construction and debugging strategies to simulate key aspects of a target phenomenon. Teacher educators should use formative assessments in professional learning to identify strengths and gaps in teachers' scientific model development as well as coding skills.

Summative assessment methods are equally important as they facilitate evaluation of one's models of science phenomena and coding skills at the end of a scientific modeling unit. Examples of summative assessment methods include (a) creation and/or explanation of code for specific scientific modeling scenarios, (b) debugging faulty codes to improve simulation accuracy (Fields, Searle, Kafai, & Min, 2012), (c) transferring coding skills and/or scientific models to problem solving (Han Koh, Basawapatna, Bennett, & Repenning, 2010), and (d) examination of student-generated artifacts (Werner et al., 2012), such as their designed simulations of a science phenomenon.

Coding in Scientific Modeling Lessons (CS-Model)

The proposed design guidelines informed the design and development of Coding in Scientific Modeling Lessons (CS-Model). CS-Model is an instructional module and online tool that aim at scaffolding teachers' learning to (a) use block-based coding to create simulations of a science phenomenon and (b) integrate code-based simulations into lesson design to support scientific modeling. The following sections present activities in the CS-Model module (see Table 2.1 for an overview), discuss features in the CS-Model online tool, and indicate which guideline(s) informed their design.

CS-Model Module

CS-Model is a four-week module that encompasses face-to-face and asynchronous online activities. The first activity in week 1 of the CS-Model module is a face-to-face coding workshop that was designed based on the assumption that in-service and preservice science teachers have little to no background experience with coding. During the workshop, teachers become familiarized with the block-based coding tool Scratch, review and practice code error debugging, and complete coding tasks to learn to use blocks that will be needed in subsequent

simulation coding activities. This coding workshop is critical to leverage teachers' skills, prevent coding errors during upcoming and more complex activities, and train teachers to thoughtfully engage in error debugging (Kim et al., 2018). While coding, teachers complete reflection prompts that scaffold code debugging such as *What is your hypothesis to identify and fix the error?* (Guideline 4). Throughout CS-Model, all coding activities occur in pairs so teachers benefit from peer scaffolding (Belland, 2014, 2017; Lai & Law, 2006) (Guideline 4). Pair coding is an effective strategy for novice learners (Hahn, Mentz, & Meyer, 2009; McDowell, Werner, Bullock, & Fernald, 2002).

Table 2.1

Overview of CS-Model Module Timeline and Activities

Week	Delivery	Activity	Intended Learning Outcomes
1	Face-to-face	Coding workshop	<ul style="list-style-type: none"> - Become familiarized with block-based coding language and tool (Scratch) - Practice coding concepts (loops, variables, conditionals, and delays)
	Asynchronous online	<ul style="list-style-type: none"> - Analyze instructional materials and code-based simulations - Reflect on pedagogical implications of teaching scientific modeling with coding 	<ul style="list-style-type: none"> - Expand repertoire of teaching strategies (framework, scaffolds, assessments) to support simulation coding in scientific modeling - Understand pedagogical implications of teaching scientific modeling with coding in teachers' areas of science teaching
2	Face-to-face	Use block-based coding simulations and analogous lab experiments to develop models of a science phenomenon while pursuing an epistemic goal	<ul style="list-style-type: none"> - Construct, test, evaluate, and revise models through simulation coding - Practice authentic scientific modeling practices - Identify connections between coding and science concepts
	Asynchronous online	Design scientific modeling lessons that use simulation coding	<ul style="list-style-type: none"> - Apply coding knowledge, choose a science phenomenon, and design a scientific modeling lesson that uses block-based coding simulations

All coding activities occur in Scratch (Figure 2.2), which is a block-based language and coding environment. Scratch was selected because it is a low-floor, high-ceiling, and wide-wall tool, i.e., it is easy for novice learners to use, adaptable to increasingly advanced projects, and conducive of alternative problem-solving strategies (Grover et al., 2015; Repenning et al., 2010; Resnick et al., 2009). Scratch enables users to express their models using a programming language and receive immediate visual feedback on the artifacts they create (Li & Watson, 2011; Pellas & Peroutseas, 2016). Scratch can be used to create self-made simulations of science phenomena (Guideline 3) that embody one's scientific models and serve as test beds to investigate how a phenomenon occurs (Guideline 1). Research, including a study with preservice teachers (Bell, Frey, & Vasserman, 2014), shows positive attitudes towards Scratch as a tool used to learn programming concepts (Maloney et al., 2008; Weintrop, 2015; Weintrop & Wilensky, 2015, 2016).

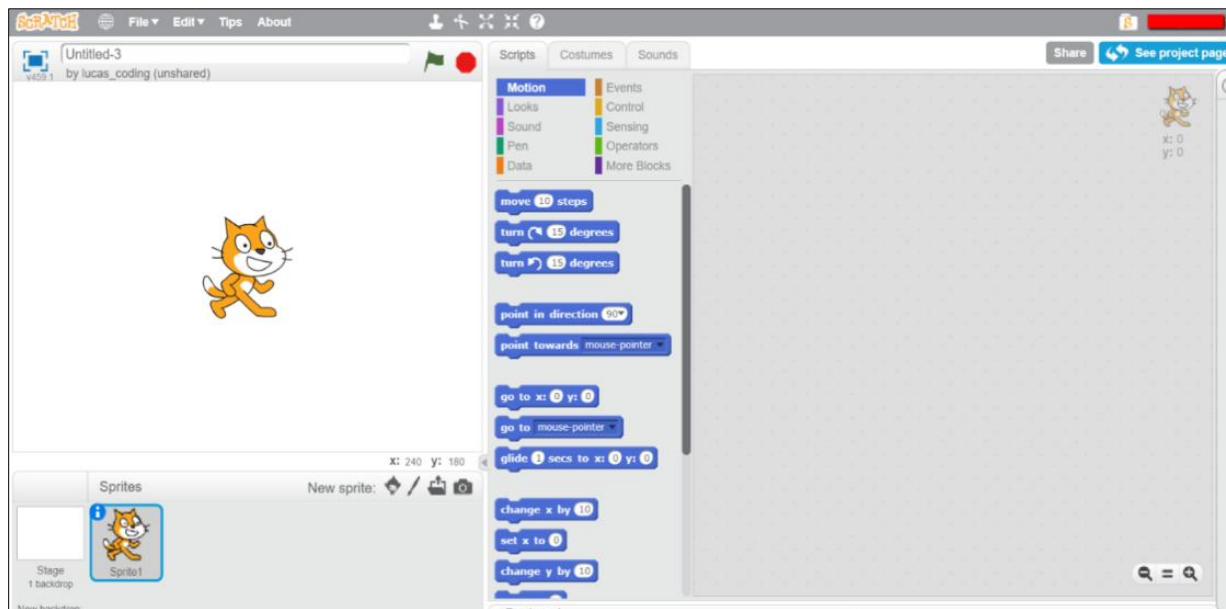


Figure 2.2. Screenshot of Scratch.

After the coding workshop, teachers asynchronously review instructional materials related to simulation coding in scientific modeling instruction. They (a) read an article about

coding in schools (Merrill, 2017), (b) review GUTS (n.d.) lessons (Guideline 1) that tackle topics pertinent to their teaching contexts (e.g., biology, chemistry), (c) explore existing Scratch simulations on topics relevant to their teaching, and (d) complete a reflection activity on their perceptions of pedagogical implications for teaching scientific modeling with block-based coding (Guideline 4). These activities are expected to help teachers further develop content and pedagogical content knowledge (Dass et al., 2015; Weiss & Pasley, 2006) about ways to use a scientific modeling framework in lessons (Guideline 2); create strong connections between science and computer science concepts from lessons (ACM K-12 Taskforce, 2003; Sengupta et al., 2013); and expand their repertoire of teaching strategies, scaffolds, and assessments (Guidelines 4 and 5).

In week 2, teachers participate in authentic scientific modeling activities (Guideline 1) that involve coding simulations of water filtration systems and conducting analogous physical experiments (Table 2.2). The activities involving physical experiments are adapted from Kim and Oliver (2018). These authors did not include coding or simulations in their lesson unit. CS-Model activities are tailored around Schwarz et al.'s (2009) scientific modeling framework (Guideline 2) which includes these steps: (a) anchor phenomenon, (b) construct model, (c) test model, (c) evaluate model, (d) revise model, and (e) use model to predict or explain. This framework was selected because it aligns with our understanding that scientific modeling instruction should promote epistemic agency by enabling epistemic agents to create and develop their own models as well as pursue their own epistemic goals (Gouvea & Passmore, 2017; Knuuttila, 2005b, 2011; Schwarz, 2009; Schwarz et al., 2009). Additionally, the framework typifies a pragmatic approach to modeling as it addresses scientific models as flexible artifacts designed for specific contexts and goals (Schwarz et al., 2009; Schwarz & White, 2005).

Table 2.2

Overview of Water Filtration Activities

Activities	Scientific Modeling Steps
1. Present problem: create an effective water filter to clear river water at campground - Review scenario, constraints, and rules for problem - Discuss possible science ideas to relevant to the problem - Brainstorm and select dependent and independent variables to investigate - Create research question(s) to guide the investigative process	Anchor problem
2. Create hypotheses about phenomenon - In pairs, create hypotheses about the effectiveness of water filters using individual filtration materials - Design experiments using multiple layers of combined filtration materials and create hypotheses about their effectiveness	Use model to predict or explain
3. Code simulations analogous to their predicted water filter experiments - Externalize models about effectiveness of individual combined filtration materials on water quality (color), input-output ratio, and filtration time - Debug and document coding errors (e.g., error, hypothesized cause of error, proposed solution)	Construct model
4. Conduct physical experiments to test the effectiveness of water filter designs using individual and combined filtration materials - Record empirical data on water quality (color), input-output ratio, and filtration time	Test model
5. Evaluate models based on empirical data - Compare experiment results with original hypotheses - Evaluate whether hypotheses are confirmed or refuted by collected data - Discuss and decide how models can be redesigned for improved results	Evaluate model
6. Generate evidence-based explanations for experiment results	Use model to predict or explain
7. Revise code and simulations of water filtration systems based on refined models and collected data	Revise model

Teachers work in pairs during all CS-Model face-to-face activities. Pair work is expected to promote epistemic dialogue, that is, collaboratively crafting explanations about underlying mechanisms of a science phenomenon and solving a problem (De Vries, Lund, & Baker, 2002). Pair work is conducive of peer scaffolding (Guideline 4) on scientific modeling and coding tasks. For instance, teachers share their models, negotiate meaning, reach a consensus

model, and jointly decide how to code a simulation of a science phenomenon. During coding activities, one teacher manipulates the computer while the other documents their code errors, hypothesized cause of the error, and debugging strategies (Guideline 4). These roles are reserved as teachers code different simulations.

To start face-to-face activities, teachers are presented with a problem scenario, constraints, and a rule, which are respectively: (a) they run out of drinkable water at a campground, (b) there are several materials available (bottle, cotton, gravel, and different sizes of activated charcoal), and (c) they need to create an effective filtration system to remove impurities of river water. Using whole-class discussion as peer scaffolding (Guideline 4), teachers discuss science ideas involved in the problem, identify independent and dependent variables to investigate, and create research question(s) to guide the inquiry process. For instance, teachers can examine how using different filtration materials affects water quality (color as a proxy), input-output ratio, and filtration time. Next, teachers work in pairs to formulate hypotheses about water filter designs (e.g., how cotton affects dependent variables) using one filtration material at a time or a combination of them. Although the overarching problem presented to teachers determines the scope of their inquiry to some extent, subsequent activities are designed to foster teachers' epistemic agency as they pursue their own epistemic goals through construction and development of water filtration systems (Guideline 1).

Subsequently, teachers code simulations, which are analogous to their designed experiments, to externalize hypotheses and predict the most effective water filter design (Guidelines 1 and 3). Creating and coding water filter simulations from scratch would be time-consuming. Thus, teachers are provided with simulations that are previously designed and partially coded so they can focus on coding specific blocks that represent key variables in water

filtration experiments. Figure 2.3 shows a screenshot of a Scratch simulation using cotton. The simulation is designed so that it is visually aligned with physical experiments. For example, the simulation features dots that represent impurities retained by cotton in the bottle, different water colors inside and outside the filter, and use of light green color for water output to represent the limited effect of using cotton to clear up water.

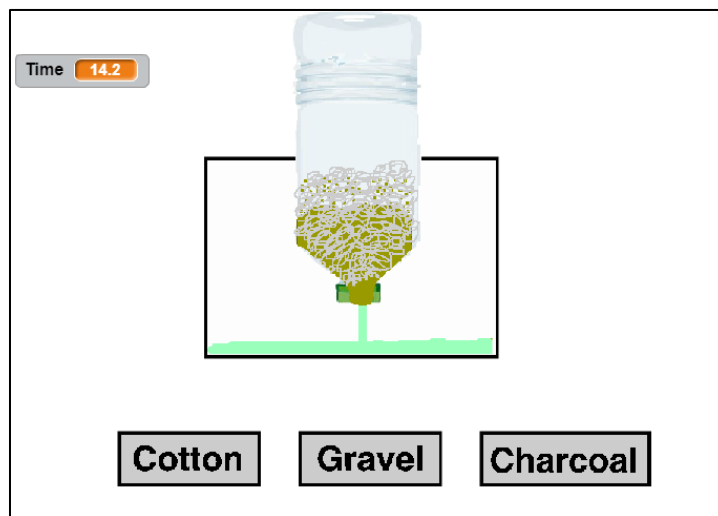


Figure 2.3. Simulation of water filtration system using cotton.

To code simulations, teachers complete blank code block parameters so that each button (Figure 2.3) activates the corresponding filtration material and simulates predicted water quality, input-output ratio, and filtration time. Specifically, teachers code the blocks *when I receive* with cotton, gravel, or charcoal and *broadcast* with clear, dark, or brackish water to create independent-dependent variable relationships (conditionals). Teachers enter a number in the *repeat* block to determine how many water filtration frames (loops) are executed for the water sprite, which simulates input-output ratio. A high the number of loops causes the water sprite to simulate a higher amount of water leaving the filter. Teachers enter a number in *wait__secs* to control delay between sprite frames, and consequently, simulate overall filtration time. Figure 2.4 shows an example of incomplete (left) and complete (right) code blocks. In this example,

cotton yields brackish water, a high input-output ratio given that only 15 out of 29 (total) frames are executed (cotton absorbs water), and slow water filtration given the 1-second time interval between animation frames (15 seconds total). The time counter (Figure 2.3) starts and stops with the simulation to show total filtration time. The simulation storyboard is presented in Figure 2.5.

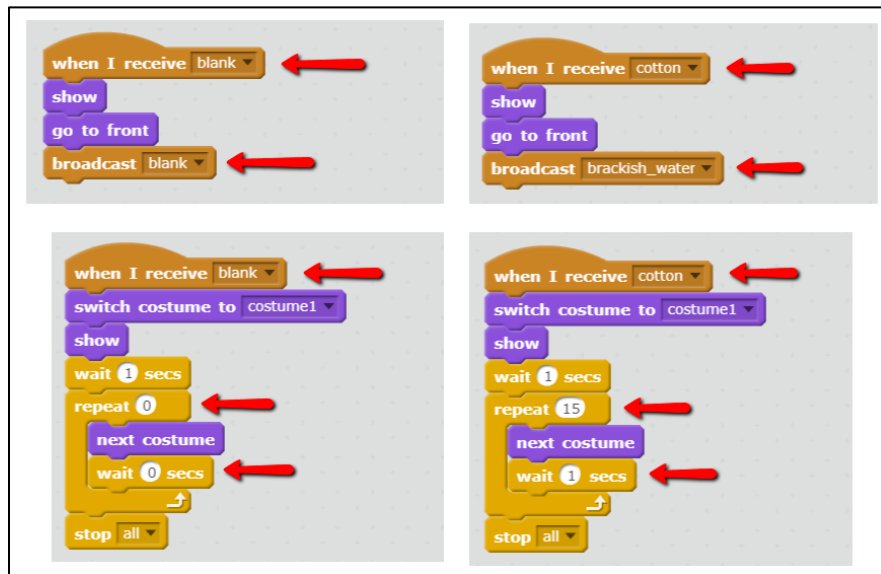


Figure 2.4. Example of code used in simulation.

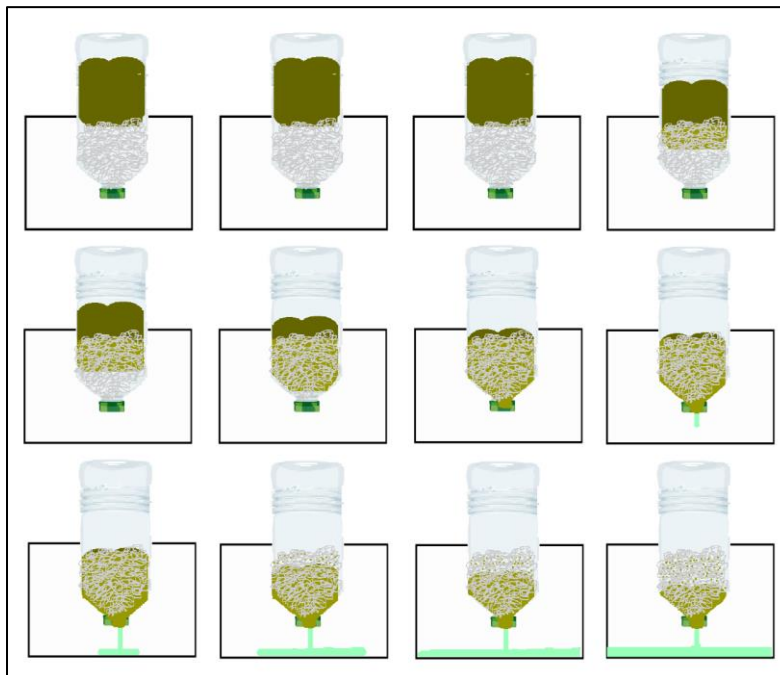


Figure 2.5. Storyboard for water filter simulation using cotton.

Subsequently, teachers code simulations that express their predictions about the water filtration experiment using small, medium, and big activated charcoal sizes, as well as combined materials. While coding, teachers construct, test, evaluate, and revise code and simulation output. This process allows teachers to use such code-based simulations to test their models about the effectiveness of different materials, visualize dynamic representations of key variables over time, compare the simulation output against their own predictions, and construct a model of an effective water filtration system (Guideline 3).

Next, teachers conduct analogous water filtration experiments in a science lab (Guideline 1). Teachers gather data during each experiment by taking pictures of water quality (color) before and after filtration, writing down water input and output volume to calculate the ratio, and recording filtration time. Then teachers compare data to their hypotheses, evaluate whether data refute or confirm them, and identify aspects in their models that need to be revised. Moreover, teachers generate explanations about experiment results and revise their scientific models on effective water filtration systems. Finally, teachers revise the simulations of water filtration experiments to express and solidify their revised models.

CS-Model features several assessment methods on coding and scientific model development (Guideline 5). Formative assessments include responses to reflection prompts (e.g., *How do different materials affect input-output ratio?*) offered during whole-class discussion, self-generated hypotheses about the experiment, documentation of code error debugging, and documentation of collected empirical data, iterative process of model development during simulation coding and physical experiments. Summative assessments include revised simulations, an essay in which teachers explain how coding materializes water filtration variables, and a problem scenario wherein teachers need to apply coding skills.

The CS-Model Tool

CS-Model is an online tool designed to scaffold teachers' learning to integrate simulation coding into scientific modeling lessons, and it is hosted in Qualtrics. Besides traditional lesson plan features (grade, subject, topic, duration, and materials), CS-Model contains links to the NGSS learning standards and the K-12 Computer Science Framework (2016) so that teachers can review and select science and computer science standards respectively for their lesson. Additionally, CS-Model prompts design of learning objectives for each discipline. These features are expected to help teachers design interdisciplinary lessons in which science and computer science are meaningfully interwoven (NRC, 2012; Sengupta et al., 2013) rather than connected as add-ons.

CS-Model provides a seven-step framework adapted from Schwarz et al. (2009) (Table 2.3) to scaffold design of scientific modeling activities. Coding tasks are integrated into scientific modeling steps (Guideline 2). Each step is accompanied by hints, and both the steps and hints serve as scaffolds (Guideline 4) that structure and problematize (Reiser, 2002, 2004) lesson design by respectively creating a clear structure for integrating coding into scientific modeling and by highlighting key elements/processes that teachers should attend to. CS-Model steps scaffold design of lessons that combine simulation coding activities and scientific modeling activities (Guideline 1) as strategies to help one construct and develop models while pursuing epistemic goals. It is important to note that CS-Model features *assessment*, a step that was not included in Schwarz et al.'s (2009) framework. The step *assessment* scaffolds design of formative and summative methods to assess development of coding skills and scientific model (Guideline 5). Such a step is critical to identify challenges and determine the type and amount of support that is needed to improve one's learning.

Table 2.3

Modification of Schwarz et al.'s (2009) Framework for the Design of CS-Model

Schwarz et al. (2009)	CS-Model
Anchoring phenomena <ul style="list-style-type: none"> - Introduce driving questions and phenomena for a particular concept - Use a phenomenon that may necessitate using a model to figure it out 	Anchor phenomenon <ul style="list-style-type: none"> - Introduce driving questions and phenomena for a particular concept. Use a phenomenon that may necessitate using a model to figure it out - Explain the concept of scientific models - Introduce coding as a strategy to create visual representations of scientific models
Construct a model <ul style="list-style-type: none"> - Create an initial model expressing an idea or hypothesis - Discuss purpose and nature of models 	Construct model <ul style="list-style-type: none"> - Facilitate student reflection about agents, relationships, and interactions in the model - Prompt students to use coding to create an initial model expressing an idea or hypothesis - Encourage students to debug coding errors - Offer students opportunities for reflection about the purpose and nature of models - Offer students opportunities for explaining their hypotheses/ideas and how they use coding to simulate their models
Empirically test the model <ul style="list-style-type: none"> - Investigate the phenomena predicted and explained by the model 	Test model <ul style="list-style-type: none"> - Design activities in which students investigate the phenomenon predicted and explained by the model - Design opportunities for peer feedback on student-generated scientific model and code-based simulation
Evaluate the model <ul style="list-style-type: none"> - Return to the model and compare it with empirical findings - Discuss qualities for evaluation and revision 	Evaluate model <ul style="list-style-type: none"> - Provide students with opportunities to compare model with empirical findings and/or peer feedback - Encourage students to identify and discuss criteria for model evaluation and revision - Facilitate reflection on how to improve their model - Facilitate reflection on how to re-write the code to revise their model
Revise the model <ul style="list-style-type: none"> - Change the model to fit new evidence - Compare competing models and construct a consensus model 	Revise model <ul style="list-style-type: none"> - Engage students in comparing competing models - Promote opportunities for constructing/coding a consensus model - Offer students opportunities to revise the model and the code to fit their most current thinking and/or new experiential data - Encourage students to debug coding errors
Use the model to predict or explain <ul style="list-style-type: none"> - Apply model to predict and explain other phenomena 	Use model to predict or explain <ul style="list-style-type: none"> - Design activities in which students use model and code knowledge to predict and explain phenomena

- Not included	Assessment - Create formative assessment strategies to monitor ongoing student learning/performance in scientific model progression and coding skills, and provide them with feedback - Create summative assessment strategies to evaluate student learning/performance in scientific model progression and coding skills at the end of the lesson
----------------	--

Conclusion

School science education standards emphasize the importance of designing learning experiences that engage students in performing authentic practices related to scientists' profession (Gouvea & Passmore, 2017; NGSS Lead States, 2013; Osborne, 2014). Scientific modeling is one of such practices, and it entails constructing, testing, evaluating, and revising models to pursue an epistemic goal related to a science phenomenon (Duschl, 2008; Gouvea & Passmore, 2017; Osborne, 2014; Stroupe, 2014). Scientific modeling instruction challenges traditional science teaching, which involves model memorization and recall, by fostering students' epistemic agency through construction and development of their own models (Berland et al., 2016; Gouvea & Passmore, 2017; Knuuttila & Boon, 2011; Osborne, 2014). Scientific modeling instruction is expected to help students better understand science concepts, science as a discipline, and what it means to conduct authentic scientific inquiry.

However, conceptualizing abstract models of unobservable science phenomena (Barak et al., 2011; Cheng et al., 2014; Windschitl et al., 2008) is challenging for K-12 students. Computer-based simulations have been extensively used as visual scaffolds for construction and development of scientific models though research has found that simulations are not always effective (D'Angelo et al., 2014; Smetana & Bell, 2012). Teachers should not assume that manipulation of simulation variables and observation of what happens on the computer screen will lead to development of more robust models (Marshall & Young, 2006; Shen et al., 2014).

Use of simulations dissociated from scientific inquiry prevents students from developing epistemic agency through modeling.

One way to support scientific modeling and promote epistemic agency in K-12 science classrooms is by using block-based coding to simulate science phenomena. This constructivist and constructionist approach (Girvan et al., 2013; Kafai, 2012; Papert, 1980) entails supporting students in externalizing their own models by constructing a simulation which serves as a self-expression artifact, as an object to reason with (Holbert & Wilensky, 2018), and as a tool to help them accomplish epistemic goals related to the simulated phenomenon. More specifically, self-made simulations serve as test beds to run virtual experiments, test hypotheses, visualize how unobservable elements occur, generate evidence-based explanations, make predictions, and advance one's knowledge about a target phenomenon. Simulation coding has potential to help students (a) develop more sophisticated models of a science phenomenon; (b) learn computational concepts and processes, such as coding and debugging; and (c) develop interdisciplinary problem solving that requires borrowing concepts and processes from science and computer science.

Research indicates that science teachers often lack a refined understanding of scientific models and modeling (Kenyon et al., 2011; Krell & Krüger, 2016; Schwarz et al., 2009; Windschitl et al., 2008), and K-12 teachers lack pedagogical knowledge of teaching with coding even if they graduate from computer science certification programs (Gal-Ezer & Stephenson, 2010; Google & Gallup, 2015a; Yadav et al., 2016). Exemplary lessons that integrate block-based coding into scientific modeling have been made available to teachers at no cost (GUTS, n.d.), and a framework for integrating computational thinking and scientific modeling into K-12 instruction (Sengupta et al., 2013) has been proposed. However, the literature on integrating

simulation coding into scientific modeling is still scarce. No study has attempted to scaffold teachers' learning to (a) code science simulations and (b) integrate coding into scientific modeling lessons. This literature gap motivated this chapter.

After reviewing pertinent literature, five guidelines are proposed for design and development of professional learning for preservice and in-service science teachers. The guidelines are (a) combine simulation coding with authentic scientific inquiry, (b) tailor coding activities to a scientific modeling framework, (c) promote creation and development of self-made simulations, (d) design scaffolds for engagement in coding and scientific modeling, and (e) assess both scientific model development and coding skills. These guidelines informed the design and development of Coding in Scientific Modeling Lessons (CS-Model), an instructional module and online tool for teacher educators to scaffold science teachers' learning to code science simulations and integrate simulation coding into scientific modeling lessons.

CS-Model is a promising approach that embodies contemporary educational trends in the learning sciences: preparing teachers for interdisciplinary STEM teaching, integrating computational concepts and processes into K-12 education, promoting K-12 students' epistemic agency, and offering authentic learning experiences that entail application of STEM-related practices for problem solving. The CS-Model module and tool are a first step towards a transformative research agenda that challenges the individualized teaching of science, technology, engineering, and mathematics and supports a more integrated paradigm of STEM teaching and learning. As such, CS-Model is expected to benefit teacher educators striving to offer interdisciplinary professional learning involving coding and scientific modeling to preservice and in-service science teachers. The CS-Model module and online tool will be implemented in future empirical studies.

References

- Abell, S. K., & Roth, M. (1995). Reflections on a fifth-grade life science lesson: Making sense of children's understanding of scientific models. *International Journal of Science Education, 17*(1), 59–74. <https://doi.org/10.1080/0950069950170105>
- ACM K-12 Taskforce. (2003). *A Model Curriculum for K-12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee*. New York, NY: CSTA.
- Akerson, V. L., Townsend, J. S., Donnelly, L. A., Hanson, D. L., Tira, P., & White, O. (2009). Scientific modeling for inquiring teachers network (SMIT'N): The influence on elementary teachers' views of nature of science, inquiry, and modeling. *Journal of Science Teacher Education, 20*(1), 21–40. <http://dx.doi.org/10.1007/s10972-008-9116-5>
- Akpan, J. P. (2002). Which comes first: Computer simulation of dissection or a traditional laboratory practical method of dissection. *Electronic Journal of Science Education, 6*(4). Retrieved from <http://www.physicsclassroom.com/class/light/u1212e.cfm>
- Azevedo, R., Cromley, J. G., Moos, D. C., Greene, J. A., & Winters, F. I. (2011). Adaptive content and process scaffolding: A key to facilitating students' self-regulated learning with hypermedia. *Psychological Test and Assessment Modeling, 53*(1), 106–140.
- Bamberger, Y. M., & Davis, E. A. (2013). Middle-school science students' scientific modelling performances across content areas and within a learning progression. *International Journal of Science Education, 35*(2), 213–238. <https://doi.org/10.1080/09500693.2011.624133>
- Barak, M., Ashkar, T., & Dori, Y. J. (2011). Learning science via animated movies: Its effect on students' thinking and motivation. *Computers & Education, 56*(3), 839–846. <https://doi.org/10.1016/j.compedu.2010.10.025>

- Baratè, A., Ludovico, L. A., Mangione, G. R., & Rosa, A. (2015). Playing music, playing with music: A proposal for music coding in primary school. In *International Association for Development of the Information Society*. Retrieved from <http://files.eric.ed.gov/fulltext/ED562460.pdf>
- 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, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM*, 60(6), 72–80.
- Bell, R., Gess-Newsome, J., & Luft, J. (2008). *Technology in the secondary science classroom*. Arlington, VA: NSTA Press.
- Bell, R. L., & Smetana, L. K. (2008). Using computer simulations to enhance science teaching and learning. In R. L. Bell, J. G. Newsome, & J. Luft (Eds.), *Technology in the Secondary Science Classroom* (pp. 23–32). Arlington, VA: NSTA Press.
- Bell, S., Frey, T., & Vasserman, E. (2014). Spreading the word: Introducing pre-service teachers to programming in the K-12 classroom. In *Proceedings of SIGCSE '14* (pp. 187–192). Atlanta, GA: ACM Press. <https://doi.org/10.1145/2538862.2538963>
- Belland, B. R. (2011). Distributed cognition as a lens to understand the effects of scaffolds: The role of transfer of responsibility. *Educational Psychology Review*, 23(4), 577–600. <https://doi.org/10.1007/s10648-011-9176-5>
- Belland, B. R. (2014). Scaffolding: Definition, current debates, and future directions. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (4th ed, pp. 505–518). New York, NY:

Springer.

- Belland, B. R. (2017). *Instructional scaffolding in STEM education*. Cham: Springer International Publishing. <https://doi.org/10.1007/978-3-319-02565-0>
- Berland, L. K., Schwarz, C. V., Krist, C., Kenyon, L., Lo, A. S., & Reiser, B. J. (2016). Epistemologies in practice: Making scientific practices meaningful for students. *Journal of Research in Science Teaching*, 53(7), 1082–1112. <https://doi.org/10.1002/tea.21257>
- Biffi, D., Hartweg, B., de la Fuente, Y., Patterson, M., Stewart, M., Simanek, E., & Weinburgh, M. (2016). Developing an educational tool to model food chains. *Electronic Journal of Science Education*, 20(1), 40–53.
- Bowen, B., & Deluca, W. (2015). Comparing traditional versus alternative sequencing of instruction when using simulation modeling. *Journal of STEM Education: Innovations and Research*, 16(1), 5.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada* (pp. 1–25). Retrieved from <http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf>
- Buckley, B. C. (2000). Interactive multimedia and model-based learning in biology. *International Journal of Science Education*, 22(9), 895–935. <https://doi.org/10.1080/095006900416848>
- Buckley, B. C. (2012). Model-based learning. In N. M. Seel (Ed.), *Encyclopedia of the sciences of learning* (pp. 2300–2303). Boston, MA: Springer. Retrieved from http://www.springerlink.com/index/10.1007/978-1-4419-1428-6_589
- Buckley, B. C., Gobert, J. D., Kindfield, A. C. H., Horwitz, P., Tinker, R. F., Gerlits, B., ...

- Willett, J. (2004). Model-based teaching and learning with BioLogica™: What do they learn? How do they learn? How do we know? *Journal of Science Education and Technology*, 13(1), 23–41.
- Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. *Journal of Media Literacy Education*, 4(2), 121–135.
- Chakravartty, A. (2010). Informational versus functional theories of scientific representation. *Synthese*, 172, 197–213.
- Chang, H. Y., Quintana, C., & Krajcik, J. (2010). The impact of designing and evaluating molecular animations on how well middle school students understand the particulate nature of matter. *Science Education*, 94(1), 73–94.
- Cheng, M., Lin, J., Chang, Y., Li, H., Wu, T., & Lin, D. (2014). Developing explanatory models of magnetic phenomena through model-based inquiry. *Journal of Baltic Science Education*, 13(3), 351–360.
- Chu, S. L., Deuermeyer, E., & Quek, F. (2017). Supporting scientific modeling through curriculum-based making in elementary school science classes. *International Journal of Child-Computer Interaction*. <https://doi.org/10.1016/j.ijcci.2017.09.002>
- Clement, J. (2000). Model based learning as a key research area for science education. *International Journal of Science Education*, 22(9), 1041–1053.
<http://dx.doi.org/10.1080/095006900416901>
- Clement, J. J. (2008). *Creative model construction in scientists and students: The role of imagery, analogy, and mental simulation*. Dordrecht, NL: Springer.
- Coll, R. K., France, B., & Taylor, I. (2005). The role of models/and analogies in science education: Implications from research. *International Journal of Science Education*, 27(2),

183–198. <https://doi.org/10.1080/0950069042000276712>

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing and mathematics. In L. B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (pp. 453-491). Hillsdale, NJ: Erlbaum Associates.

Collins, Allan, Brown, J. S., & Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. *American Educator*, 15(3), 6–11.

Computer Science Teachers Association & International Society for Technology in Education. (2011). Computational thinking in K-12 education: Leadership toolkit. Retrieved from <http://www.iste.org/docs/ct-documents/ct-leadership-toolkit.pdf?sfvrsn=4>

Cook, M. P. (2006). Visual representations in science education: The influence of prior knowledge and cognitive load theory on instructional design principles. *Science Education*, 90(6), 1073–1091. <https://doi.org/10.1002/sce.20164>

Crawford, B., & Cullin, M. (2004). Supporting perspective teachers' conceptions of modeling in science. *International Journal of Science Education*, 26(11), 1379–1401.

D'Angelo, C., Rutstein, D., Harris, C., Bernard, R., Borokhovski, E., & Haertel, G. (2014). *Simulations for STEM learning: Systematic review and meta-analysis*. Menlo Park, CA: SRI International.

Danusso, L., Tesla, I., & Vicentini, M. (2010). Improving prospective teachers' knowledge about scientific models and modeling: Design and evaluation of a teacher education intervention. *International Journal of Science Education*, 32(7), 871–905.

Dass, K., Head, M. L., & Rushton, G. T. (2015). Building an understanding of how model-based inquiry is implemented in the high school chemistry classroom. *Journal of Chemical*

- Education*, 92(8), 1306–1314. <http://doi.dx.org/10.1021/acs.jchemed.5b00191>
- de Jong, T., & van Joolingen, W. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, 68(1), 179–201.
- De Vries, E., Lund, K., & Baker, M. (2002). Computer-mediated epistemic dialogue: Explanation and argumentation as vehicles for understanding scientific notions. *Journal of the Learning Sciences*, 11(1), 63–103. https://doi.org/10.1207/S15327809JLS1101_3
- Driel, J. H. V., & Verloop, N. (1999). Teachers' knowledge of models and modelling in science. *International Journal of Science Education*, 21(11), 1141–1153.
- Duschl, R. (2008). Science education in three-part harmony: Balancing conceptual, epistemic, and social learning goals. *Review of Research in Education*, 32(1), 268–291. <https://doi.org/10.3102/0091732X07309371>
- Eichinger, D. C., Nakhleh, M. B., & Auberry, D. L. (2000). Using a computer simulation before dissection to help students learn anatomy. *Journal of Computers in Mathematics and Science Teaching*, 19(3), 253–275.
- Fields, D. A., Searle, K. A., Kafai, Y., & Min, H. S. (2012). Debuggems to assess student learning in e-textiles. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (p. 699). Raleigh, North Carolina, USA: ACM.
- Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation is critical. *ACM Inroads*, 1(1), 61–66.
- Gelbart, H., Brill, G., & Yarden, A. (2009). The impact of a web-based research simulation in bioinformatics on students' understanding of genetics. *Research in Science Education*, 39(5), 725–751.
- Giere, R. N. (1988). *Explaining science: A cognitive approach*. University of Chicago Press.

- Giere, R. N. (2004). How models are used to represent reality. *Philosophy of Science*, 71, 742–752.
- Giere, R. N. (2006). The role of agency in distributed cognitive systems. *Philosophy of Science*, 73(5), 710–719. <https://doi.org/10.1086/518772>
- Gilbert, J. K. (2008). Visualization: An emergent field of practice and enquiry in science education. In J. K. Gilbert, M. Reiner, & M. Nakhlel (Eds.), *Visualization: Theory and practice in science education* (pp. 3–24). New York, NY: Springer.
- Gilbert, J. K., & Boulter, C. J. (2000). *Developing models in science education*. Netherlands: Kluwer Academic Publishers.
- Gilbert, J. K., & Justi, R. (2016). *Modelling-based teaching in science education*. Cham: Springer Nature.
- Gilbert, S. W. (1991). Model building and a definition of science. *Journal of Research in Science Teaching*, 28(1), 73–79. <https://doi.org/10.1002/tea.3660280107>
- Girvan, C., Tangney, B., & Savage, T. (2013). SLurples: Supporting constructionist learning in Second Life, 61(1), 115–132. <https://doi.org/10.1016/j.compedu.2012.08.005>
- Goode, J., Flapan, J., & Margolis, J. (2018). Computer science for all: A school reform framework for broadening participation in computing. In W. G. Tierney, Z. B. Corwin, & A. Ochsner (Eds.), *Diversifying digital learning: Online literacy and educational opportunity* (pp. 45–65). Baltimore, MD: Johns Hopkins University Press.
- Google, & Gallup. (2015a). *Images of computer science: Perceptions among students, parents and educators in the U.S.* Retrieved from <http://g.co/cseduresearch>
- Google, & Gallup. (2015b). *Searching for computer science: Access and barriers in U.S. K-12 education*. Retrieved from <http://g.co/cseduresearch>

- Gouvea, J., & Passmore, C. (2017). 'Models of' versus 'models for': Toward an agent-based conception of modeling in the science classroom. *Science & Education*, 26(1–2), 49–63. <https://doi.org/10.1007/s11191-017-9884-4>
- Gouvea, J. S., Passmore, C., & Jamshidi, A. (2014). How teachers' understandings of models and model-based reasoning influence shifts in their pedagogy. Presented at the NARST Conference, Pittsburgh, PA.
- Grosslight, L., Unger, C., Jay, E., & Smith, C. L. (1991). Understanding models and their use in science: Conceptions of middle and high school students and experts. *Journal of Research in Science Teaching*, 28(1), 73–79. <http://dx.doi.org/10.1002/tea.3660280907>
- Grover, S., Cooper, S., & Pea, R. (2014). Assessing computational learning in K-12. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 57–62). Uppsala, Sweden: ACM. <https://doi.org/10.1145/2591708.2591713>
- 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>
- Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, 25(2), 199–237. <https://doi.org/10.1080/08993408.2015.1033142>
- Hahn, J. H., Mentz, E., & Meyer, L. (2009). Assessment strategies for pair programming. *Journal of Information Technology Education Research*, 8, 273–284.
- Halloun, I. (2006). *Modeling theory in science education*. Dordrecht, The Netherlands: Springer.
- Han Koh, K., Basawapatna, A., Bennett, V., & Repenning, A. (2010). Towards the automatic recognition of computational thinking for adaptive visual language learning. In

- Proceedings of the 2010 Conference on Visual Languages and Human Centric Computing (VL/HCC 2010)* (pp. 59–66). Madrid, Spain: IEEE Computer.
- Hannafin, M. J., & Hill, J. (2008). Resource-based learning. In M. Spector, D. Merrill, J. van Merriënboer, & M. Driscoll (Eds.), *Handbook of research on educational communications and technology* (Vol. 3, pp. 525–536). New York: Lawrence Erlbaum.
- Hannafin, M. J., Hill, J., & McCarthy, J. (2001). Designing resource-based learning and performance support systems. In D. Wiley (Ed.), *Learning objects* (pp. 99–130). Bloomington, IN: AECT.
- Harel, I., & Papert, S. (1991). *Constructionism: Research reports and essays, 1985-1990*. Norwood, NJ: Ablex Publishing.
- Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education*, 22(9), 1011–1026.
<http://dx.doi.org/10.1080/095006900416884>
- Hernández, M. I., Couso, D., & Pintó, R. (2015). Analyzing students' learning progressions throughout a teaching sequence on acoustic properties of materials with a model-based inquiry approach. *Journal of Science Education and Technology*, 24(2–3), 356–377.
<https://doi.org/10.1007/s10956-014-9503-y>
- Hodson, D. (2014). Learning science, learning about science, doing science: Different goals demand different learning methods. *International Journal of Science Education*, 36(15), 2534–2553. <https://doi.org/10.1080/09500693.2014.899722>
- Hokayem, H., & Schwarz, C. (2014). Engaging fifth graders in scientific modeling to learn about evaporation and condensation. *International Journal of Science and Mathematics Education*, 12(1), 49–72. <https://doi.org/10.1007/s10763-012-9395-3>

- Holbert, N., & Wilensky, U. (2014). Constructible authentic representations: Designing video games that enable players to utilize knowledge developed in-game to reason about science. *Technology, Knowledge and Learning*, 19(1–2), 53–79.
<https://doi.org/10.1007/s10758-014-9214-8>
- Holbert, N., & Wilensky, U. (2018). Designing educational video games to be objects-to-think-with. *Journal of the Learning Sciences* 28(1), 32–72.
<https://doi.org/10.1080/10508406.2018.1487302>
- Horikoshi, R. (2015). Illustrating catalysis with interlocking building blocks: A binap-ruthenium complex catalyzed asymmetric hydrogenation. *Journal of Chemical Education*, 92(1), 332–335.
- Hundhausen, C. D., & Brown, J. L. (2007). What you see is what you code: A “live” algorithm development and visualization environment for novice learners. *Journal of Visual Languages and Computing*, 18(1), 22–47.
- Ioannidou, A., Repenning, A., Keyser, D., Luhn, L., & Daetwyler, C. (2010). Mr. Vetro: A collective simulation for teaching health science. *International Journal of Computer-Supported Collaborative Learning*, 5(2), 141–166. <https://doi.org/10.1007/s11412-010-9082-8>
- Justi, R. S., & Gilbert, J. K. (2002). Science teachers’ knowledge about and attitudes towards the use of models and modelling in learning science. *International Journal of Science Education*, 24(12), 1273–1292. <https://dx.doi.org/10.1080/09500690210163198>
- K-12 Computer Science Framework Steering Committee. (2016). *K–12 computer science framework*. Retrieved from <http://www.k12cs.org>
- Kafai, Y. (2012). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge handbook of the*

- learning sciences* (pp. 35–46). New York, NY: Cambridge University Press.
- Kehtarnavaz, N., & Gope, C. (2006). DSP system design using LabVIEW and Simulink: A comparative evaluation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. Retrieved from http://server3.eca.ir/isi/forum/DSP_SYSTEM_DESIGN_USING_LABVIEW_AND_SIMULINK.pdf
- Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys*, 37(2), 83–137.
- Kenyon, L., Davis, E. A., & Hug, B. (2011). Design approaches to support preservice teachers in scientific modeling. *Journal of Science Teacher Education*, 22(1), 1–21. <https://doi.org/10.1007/s10972-010-9225-9>
- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2017). Prospective elementary teachers' debugging during block-based visual programming. Presented at the American Educational Research Association, San Antonio, TX.
- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, 46(5), 767–787. <https://doi.org/10.1007/s11251-018-9453-5>
- Kim, E., Oliver, J. S., & Jackson, D. F. (2016). Connecting the imperatives of STEM, NGSS, deep learning and assessment: A conceptual paper. Presented at the National Association for Research in Science Teaching, Baltimore, MD.
- Kim, M. C., & Hannafin, M. J. (2011). Scaffolding problem solving in technology-enhanced learning environments (TELEs): Bridging research and theory with practice. *Computers*

- & *Education*, 56(2), 403–417. <https://doi.org/10.1016/j.compedu.2010.08.024>
- Kim, Y., & Oliver, J. S. (2018). Supporting preservice teachers' use of modeling: Building a water purifier. *Innovations in Science Teacher Education*, 3(1), 1–14.
- Klahr, D., Triona, L. M., & Williams, C. (2007). Hands on what? The relative effectiveness of physical versus virtual materials in an engineering design project by middle school children. *Journal of Research in Science Teaching*, 44(1), 183–203.
<https://doi.org/10.1002/tea.20152>
- Knuuttila, T. (2005a). *Models as epistemic artefacts: Toward a non-representationalist account of scientific representation*. Helsinki, Finland: University of Helsinki.
- Knuuttila, T. (2005b). Models, representation, and mediation. *Philosophy of Science*, 72(1), 1260–1271.
- Knuuttila, T. (2011). Modelling and representing: An artefactual approach to model-based representation. *Studies in History and Philosophy of Science*, 42(2), 262–271.
<https://doi.org/10.1016/j.shpsa.2010.11.034>
- Knuuttila, T., & Boon, M. (2011). How do models give us knowledge? The case of Carnot's ideal heat engine. *European Journal for the Philosophy of Science*, 1(3), 309–334.
- Koedinger, K. R., & Corbett, A. (2006). Cognitive tutors: Technology bringing learning sciences to the classroom. In K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 61–78). Cambridge, UK: Cambridge University Press.
- Krajcik, J., & Merritt, J. (2012). Engaging students in scientific practices: What does constructing and revising models look like in the science classroom? *The Science Teacher*, 79(3), 38–41.
- Krell, M., & Krüger, D. (2016). Testing models: A key aspect to promote teaching activities

- related to models and modelling in biology lessons? *Journal of Biological Education*, 50(2), 160–173. <https://doi.org/10.1080/00219266.2015.1028570>
- Krell, M., Upmeier zu Belzen, A., & Krüger, D. (2012). Students' understanding of the purpose of models in different biological contexts. *International Journal of Biology Education*, 2(2). Retrieved from <http://dergipark.gov.tr/download/article-file/90020>
- Kumar, D. D., & Sherwood, R. D. (2007). Effect of a problem based simulation on the conceptual understanding of undergraduate science education students. *Journal of Science Education and Technology*, 16(3), 239–246. <https://doi.org/10.1007/s10956-007-9049-3>
- Lai, M., & Law, N. (2006). Peer scaffolding of knowledge building through collaborative groups with differential learning experiences. *Journal of Educational Computing Research*, 35(2), 123–144.
- Lehrer, R., & Schauble, L. (2006). Cultivating model-based reasoning in science education. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 371–388). New York, NY: Cambridge University Press.
- Lemke, J. L. (1990). *Talking science: Language, learning and values*. Westport, CT: Ablex Publishing.
- Lewis, C. M. (2012). The importance of students' attention to program state: A case study of debugging behavior. In *Proceedings of the ninth annual international conference on International computing education research* (pp. 127–134). Auckland, New Zealand: ACM.
- Li, F. W. B., & Watson, C. (2011). Game-based concept visualization for learning programming. In *Proceedings of the Third International ACM Workshop on Multimedia Technologies*

- for Distance Learning* (pp. 37–42). Scottsdale, AZ: ACM.
- Lin, J. W. (2014). Elementary school teachers' knowledge of model functions and modeling processes: A comparison of science and non-science majors. *International Journal of Science and Mathematics Education*, 12(1), 1197–1220.
- Louca, L., & Constantinou, C. (2003). The use of computer-based microworlds for developing modeling skills in physical science: An example from light. *International Journal of Science Education*. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.510.536>
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. <https://doi.org/10.1016/j.chb.2014.09.012>
- Mahr, B. (2012). On the epistemology of models. In G. Abel & J. Conant (Eds.), *Rethinking epistemology* (Vol. 2, pp. 249–300). Berlin, Boston: De Gruyter. <https://doi.org/10.1515/9783110253573.301>
- Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. In *Proceedings of the 2008 Special Interest Group on Computer Science Education* (Vol. 40, pp. 367–371). New York, NY: ACM. <https://doi.org/10.1145/1352135.1352260>
- Maloney, J. H., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, 10(4), 1–15. <https://doi.org/10.1145/1868358.1868363>
- Marbach-Ad, G., Rotbain, Y., & Stavy, R. (2008). Using computer animation and illustration activities to improve high school students' achievement in molecular genetics. *Journal of*

- Research in Science Teaching*, 45(3), 273–292. <https://doi.org/10.1002/tea.20222>
- Marshall, J. A., & Young, E. S. (2006). Preservice teachers' theory development in physical and simulated environments. *Journal of Research in Science Teaching*, 43(9), 907–937.
- McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. In *Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education* (pp. 38–42). Cincinnati, Kentucky, USA: ACM Press. <https://doi.org/10.1145/563351.563353>
- Merrill, S. (2017). The future of coding in schools. Retrieved from <https://www.edutopia.org/article/future-coding-schools>
- Meyer, R., Broad, K., Orlove, B., & Petrovic, N. (2013). Dynamic simulation as an approach to understanding hurricane risk response: Insights from the Stormview lab. *Risk Analysis*, 33(8), 1532–1552. <https://doi.org/10.1111/j.1539-6924.2012.01935.x>
- Momsen, J. L., Long, T. M., Wyse, S. A., & Ebert-May, D. (2010). Just the facts? Introductory undergraduate biology courses focus on low-level cognitive skills. *CBE - Life Sciences Education*, 9(4), 435–440.
- Morrison, M. (2015). *Reconstructing reality: Models, mathematics, and simulations*. New York, NY: Oxford University Press.
- Morrison, M., & Morgan, M. S. (1999). Models as mediating instruments. In M. M. & M. M. S. (Eds.), *Models as mediators: Perspectives on natural and social sciences* (pp. 10–37). Cambridge University Press. <https://doi.org/10.1017/CBO9780511660108.003>
- Namdar, B., & Shen, J. (2015). Modeling-oriented assessment in K-12 science education: A synthesis of research from 1980 to 2013 and new directions. *International Journal of Science Education*, 37(7), 993–1023. <https://doi.org/10.1080/09500693.2015.1012185>

- Nassiff, P., & Czerwinski, W. A. (2014). Using paperclips to explain empirical formulas to students. *Journal of Chemical Education*, 91(11), 1934–1938.
- National Research Council. (1996). *National science education standards: Observe, interact, change, learn*. Washington, DC: National Academies Press.
- National Research Council. (1999). *Being fluent with information technology*. The National Academies Press.
- National Research Council. (2008). *Taking science to school: Learning and teaching science in grades K–8*. Washington, DC: National Academy Press.
- National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC: The National Academies Press.
- National Science Aeronautics Association. (2016). *NASA scientists explain the art of creating digital hurricanes*. Retrieved from <https://www.nasa.gov/feature/goddard/2016/nasa-scientists-explain-the-art-of-creating-digital-hurricanes>
- National Science and Technology Council. (2016). *The national artificial intelligence research and development strategic plan*. The National Academies Press.
- Nelson, M. M., & Davis, E. A. (2012). Preservice elementary teachers' evaluations of elementary students' scientific models: An aspect of pedagogical content knowledge for scientific modeling. *International Journal of Science Education*, 34(12), 1931–1959. <http://dx.doi.org/10.1080/09500693.2011.594103>
- NGSS Lead States. (2013). *Next Generation Science Standards: For states, by states*. Washington, DC: National Academies Press.
- Obama, B. (2016). *Computer science for all*. Retrieved from <https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all>

- Osborne, J. (2014). Teaching Scientific Practices: Meeting the Challenge of Change. *Journal of Science Teacher Education*, 25(2), 177–196. <https://doi.org/10.1007/s10972-014-9384-1>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Passmore, C., Gouvea, J. S., & Giere, R. (2014). Models in science and in learning science: Focusing scientific practice on sense-making. In M. R. Matthews (Ed.), *International handbook of research in history, philosophy and science teaching* (pp. 1171–1202). Dordrecht, Netherlands: Springer. https://doi.org/10.1007/978-94-007-7654-8_36
- Passmore, C., Schwarz, C. V., & Mankowski, J. (2016). Developing and using models. In C. V. Schwarz, C. Passmore, & B. J. Reiser (Eds.), *Helping students make sense of the world using next generation science and engineering practices* (pp. 109–134). Arlington, VA: NSTA Press. <https://doi.org/10.2505/9781938946042>
- Paul, A. M. (2016). The coding revolution. *Scientific American*, 35(1), 42–49. <https://doi.org/10.1038/scientificamerican0816-42>
- Pellas, N., & Peroutseas, E. (2016). Gaming in Second Life via Scratch4SL: Engaging high school students in programming courses. *Journal of Educational Computing Research*, 54(1), 108–143. <https://doi.org/10.1177/0735633115612785>
- Piech, C., Sahami, M., Koller, D., Cooper, S., & Blikstein, P. (2012). Modeling how students learn to program. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (pp. 153–160). Raleigh, North Carolina, USA: ACM Press. <https://doi.org/10.1145/2157136.2157182>
- Project Growing Up Thinking Scientifically (GUTS). (n.d.). Retrieved from <http://www.projectguts.org/resources>

- Reinisch, B., & Krüger, D. (2018). Preservice biology teachers' conceptions about the tentative nature of theories and models in biology. *Research in Science Education, 48*(1), 71–103. <https://doi.org/10.1007/s11165-016-9559-1>
- Reiser, B. J. (2002). Why scaffolding should sometimes make tasks more difficult for learners. In *Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community* (pp. 255–264). International Society of the Learning Sciences. Retrieved from <http://dl.acm.org/citation.cfm?id=1658652>
- Reiser, B. J. (2004). Scaffolding complex learning: The mechanisms of structuring and problematizing student work. *The Journal of the Learning Sciences, 13*(3), 273–304.
- Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 265–269). ACM.
- Resnick, M., Maloney, J., Andrés, H., Rusk, N., Eastmond, E., Brennan, K., ... Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60–67.
- Rutten, N., van Joolingen, W. R., & van der Veen, J. T. (2012). The learning effects of computer simulations in science education. *Computers & Education, 58*(1), 136–153. <https://doi.org/10.1016/j.compedu.2011.07.017>
- Samarapungavan, A., Tippins, D., & Bryan, L. (2015). A modeling-based inquiry framework for early childhood science learning. In K. C. Trundle & M. Saçkes (Eds.), *Research in Early Childhood Science Education* (pp. 259–277). Dordrecht, Netherlands: Springer. https://doi.org/10.1007/978-94-017-9505-0_12
- Scalise, K., Timms, M., Moorjani, A., Clark, L., Holtermann, K., & Irvin, P. S. (2011). Student learning in science simulations: Design features that promote learning gains. *Journal of*

- Research in Science Teaching*, 48(9), 1050–1078. <https://doi.org/10.1002/tea.20437>
- Schwarz, C. (2009). Developing preservice elementary teachers' knowledge and practices through modeling-centered scientific inquiry. *Science Education*, 93(4), 720–744. <https://doi.org/10.1002/sce.20324>
- Schwarz, C. V., & Gwekwerere, Y. N. (2007). Using a guided inquiry and modeling instructional framework (EIMA) to support pre-service K-8 science teaching. *Science Education*, 91(1), 158–186.
- Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., ... Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners. *Journal of Research in Science Teaching*, 46(6), 632–654. <https://doi.org/10.1002/tea.20311>
- Schwarz, C. V., & White, B. Y. (2005). Metamodeling knowledge: Developing students' understanding of scientific modeling. *Cognition and Instruction*, 23(2), 165–205. http://dx.doi.org/10.1207/s1532690xci2302_1
- Seel, N. M. (2014). Model-based learning and performance. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (pp. 465–484). New York, NY: Springer New York.
- Seel, N. M. (2017). Model-based learning: A synthesis of theory and research. *Educational Technology Research and Development*, 65(4), 931–966. <https://doi.org/10.1007/s11423-016-9507-9>
- Seethaler, P. M., Fuchs, L. S., Fuchs, D., & Compton, D. L. (2012). Predicting first graders' development of calculation versus word-problem performance: The role of dynamic assessment. *Journal of Educational Psychology*, 104(1), 224.

<https://doi.org/10.1037/a0024968>

Sengupta, P., Farris, A. V., & Wright, M. (2012). From agents to aggregation via aesthetics:

Learning mechanics with visual agent-based computational modeling. *Technology, Knowledge & Learning*, *17*(1–2), 23–42.

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>

Sengupta, P., & Wilensky, U. (2009). Learning electricity with NIELS: Thinking with electrons

and thinking in levels. *International Journal of Computers for Mathematical Learning*, *14*(1), 21–50.

Shen, J., Lei, J., Chang, H., & Namdar, B. (2014). Technology-enhanced, modeling-based

instruction (TMBI) in science education. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of research on educational communications and technology*

(pp. 529–540). New York, NY: Springer New York. https://doi.org/10.1007/978-1-4614-3185-5_41

Smetana, L. K., & Bell, R. L. (2012). Computer simulations to support science instruction and

learning: A critical review of the literature. *International Journal of Science Education*, *34*(9), 1337–1370. <https://doi.org/10.1080/09500693.2011.605182>

Soloway, E. (1993). Should we teach students to program? *Communications of the ACM*, *36*(10),

21–24.

Stammen, A., Malone, K., & Irving, K. (2018). Effects of modeling instruction professional

development on biology teachers' scientific reasoning skills. *Education Sciences*, *8*(3), 1–

19. <https://doi.org/10.3390/educsci8030119>
- Stern, L., Barnea, N., & Shauli, S. (2008). The effect of a computerized simulation on middle school students' understanding of the kinetic molecular theory. *Journal of Science Education and Technology*, *17*(4), 305–315. <https://doi.org/10.1007/s10956-008-9100-z>
- Stroupe, D. (2014). Examining classroom science practice communities: How teachers and students negotiate epistemic agency and learn science-as-practice. *Science Education*, *98*(3), 487–516. <https://doi.org/10.1002/sce.21112>
- Suárez, M. (1999). Theories, models, and representations. In L. Magnani, N. J. Nersessian, & P. Thagard (Eds.), *Model-based reasoning in scientific discovery* (pp. 75–83). New York: Kluwer.
- Tao, P.-K., & Gunstone, R. F. (1999). The process of conceptual change in force and motion during computer-supported physics instruction. *Journal of Research in Science Teaching*, *36*(7), 859.
- Trundle, K. C., & Bell, R. L. (2010). The use of a computer simulation to promote conceptual change: A quasi-experimental study. *Computers & Education*, *54*(4), 1078–1088. <https://doi.org/10.1016/j.compedu.2009.10.012>
- Tucker, A., Deek, F., Jones, J., McCowan, D., Stephenson, C., & Verno, A. (2003). A model curriculum for K-12 computer science. *Final Report of the ACM K-12 Task Force Curriculum Committee, CSTA*. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.442.7961&rep=rep1&type=pdf>
- van de Pol, J., Volman, M., & Beishuizen, J. (2010). Scaffolding in teacher–student interaction: A decade of research. *Educational Psychology Review*, *22*(3), 271–296. <https://doi.org/10.1007/s10648-010-9127-6>

- Van Hook, S. J., Huziak-Clark, T. L., Nurnberger-Haag, J., & Ballone-Duran, L. (2009). Developing an understanding of inquiry by teachers and graduate student scientists through a collaborative professional development program. *Electronic Journal of Science Education, 13*(2).
- van Joolingen, W. R. (2015). Understanding elementary astronomy by making drawing-based models. *Journal of Science Education and Technology, 24*(2), 256–264.
- Wagh, A., & Wilensky, U. (2012). Evolution in blocks: Building models of evolution using blocks. In *Proceedings from Constructionism: Theory, Practice, and Impact*. Athens, Greece. Retrieved from http://www.aditiwagh.org/files/publications/WaghWilensky2012_Constructionism.pdf
- Weintrop, D. (2015). Blocks, text, and the space between: The role of representations in novice programming environments. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 301–302). Atlanta, GA: IEEE.
- Weintrop, D., & Wilensky, U. (2015). To block or not to block, that is the question: Students' perceptions of blocks-based programming. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 199–208). ACM Press. <https://doi.org/10.1145/2771839.2771860>
- Weintrop, D., & Wilensky, U. (2016). Bringing blocks-based programming into high school computer science classrooms. Presented at the Annual Meeting of the American Educational Research Association (AERA), Washington, DC, USA.
- Weiss, I. R., & Pasley, J. D. (2006). *Scaling up instructional improvement through teacher professional development: Insights from the local systemic change initiative*. CPRE Policy Briefs. Retrieved from https://repository.upenn.edu/cpre_policybriefs/32

- Werner, L., Denner, J., Campe, S., & Kawamoto, D. (2012). The fairy performance assessment: Measuring computational thinking in middle school. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 215–220). ACM. Retrieved from <http://dl.acm.org/citation.cfm?id=2157200>
- White, B., & Schwarz, C. (1999). Alternative approaches to using modeling and simulation tools for teaching science. In P. Ramsden (Ed.), *Computer modeling and simulation in science education*. New York: Springer-Verlag.
- White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118.
https://doi.org/10.1207/s1532690xci1601_2
- Wieman, C., Adams, W. K., & Perkins, K. K. (2008). PhET: Simulations that enhance learning. *Science, 322*(1), 682–683.
- Wilensky, U., & Rand, W. (2009). *An introduction to agent-based modeling: Modeling natural, social, and engineered complex systems with NetLogo*. Cambridge, MA: MIT Press.
- Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories - an embodied modeling approach. *Cognition and Instruction, 24*(2), 171–209.
- Wilkerson-Jerde, M. H., Gravel, B. H., & Macrander, C. A. (2015). Exploring shifts in middle school learners' modeling activity while generating drawings, animations, and computational simulations of molecular diffusion. *Journal of Science Education and Technology, 24*(2), 396–415.
- Winberg, T. M., & Berg, C. A. R. (2007). Students' cognitive focus during a chemistry laboratory exercise: Effects of a computer-simulated prelab. *Journal of Research in*

- Science Teaching*, 44(8), 1108–1133. <https://doi.org/10.1002/tea.20217>
- Windschitl, M. (2001). Using simulations in the middle school: Does assertiveness of dyad partners influence conceptual change? *International Journal of Science Education*, 23(1), 17–32.
- Windschitl, M., & Thompson, J. (2004). Inquiry in pre-service classrooms: Epistemological and methodological aspects. In *Proceedings of the National Association of Research in Science Teaching Conference*. Vancouver, Canada.
- Windschitl, M., Thompson, J., & Braaten, M. (2008). Beyond the scientific method: Model-based inquiry as a new paradigm of preference for school science investigations. *Science Education*, 92(5), 941–967. <http://doi.doi.org/10.1002/sce.20259>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717–3725. <https://doi.org/10.1098/rsta.2008.0118>
- Winn, W., Stahr, F., Sarason, C., Fruland, R., Oppenheimer, P., & Lee, Y. (2006). Learning oceanography from a computer simulation compared with direct experience at sea. *Journal of Research in Science Teaching*, 43(1), 25–42. <https://doi.org/10.1002/tea.20097>
- Wojnowski, B. S., & Pea, C. H. (2014). *Models and approaches to STEM professional development*. Arlington, VA: National Science Teachers Association.
- Wood, D. (2003). The why? what? when? and how? of tutoring: The development of helping and tutoring skills in children. *Literacy, Teaching and Learning*, 7(1/2), 1.
- Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100. <https://doi.org/10.1111/j.1469->

7610.1976.tb00381.x

- Wouters, P., Paas, F., & van Merriënboer, J. J. G. (2008). How to optimize learning from animated models: A review of guidelines based on cognitive load. *Review of Educational Research, 78*(3), 645–675. <https://doi.org/10.3102/0034654308320320>
- Wu, H., Krajcik, J., & Soloway, E. (2001). Promoting understanding of chemical representations: Students' use of a visualization tool in the classroom. *Journal of Research in Science Teaching, 38*(7), 821–842.
- Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education, 26*(4), 235–254. <https://doi.org/10.1080/08993408.2016.1257418>
- Zacharia, Z. (2003). Beliefs, attitudes, and intentions of science teachers regarding the educational use of computer simulations and inquiry-based experiments in physics. *Journal of Research in Science Teaching, 40*(8), 792–823. <https://doi.org/10.1002/tea.10112>
- Zacharia, Z. (2005). The impact of interactive computer simulations on the nature and quality of postgraduate science teachers' explanations in physics. *International Journal of Science Education, 27*(1), 1741–1767.
- Zacharia, Z., Olympiou, G., & Papaevripidou, M. (2008). Effects of experimenting with physical and virtual manipulatives on students' conceptual understanding in heat and temperature. *Journal of Research in Science Teaching, 45*(9), 1021–1035.

CHAPTER 3
PREPARING TEACHERS TO USE BLOCK-BASED CODING IN SCIENTIFIC MODELING
LESSONS²

² Vasconcelos, L., & Kim, C. Submitted to *Computers & Education*, 3/1/2019.

Abstract

Driven by innate epistemic agency, children attempt to explain science phenomena from a young age. In grades K-12, students need to construct and further develop their models of science phenomena as part of scientific modeling instruction, which helps develop their epistemic agency. Creating science simulations with block-based coding is a constructionist approach that allows expression, visualization, and simulation of models. However, research shows that teachers need training on scientific modeling and on integrating coding into their lessons. The present study reports the implementation of an instructional module and online system, called Coding in Scientific Modeling Lessons (CS-ModeL), which aim at supporting teachers in learning to code and to use coding in scientific modeling lessons. This study examined preservice teachers' epistemic discourse during simulation coding, perceptions of coding for future teaching, and coding-enhanced scientific modeling lessons. Findings revealed that simulation coding fosters epistemic discourse and leads to correction of misconceptions. However, lack of debugging skills and preparation for conflict argumentation is detrimental for epistemic discourse. Participants perceive coding as a beneficial skill for K-12 students though they cannot teach with coding unassisted. Participants' lessons focused on scientific modeling or coding, but not both. Recommendations for future research using CS-ModeL are provided.

Keywords: Scientific modeling, block-based coding, simulations, epistemic discourse, lesson design

Introduction

Children attempt to understand and explain science concepts from a young age. To comprehend the concept of temperature, children observe and touch objects with different temperatures, construct hypotheses about the concept, consult more knowledgeable others, ponder the consequences of manipulating objects at extreme temperatures, and refine their concept over time (De Vries, Lund, & Baker, 2002; Engel, 2013; Samarapungavan, Tippins, & Bryan, 2015; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013). Developing models of science phenomena at a young age is driven by children's innate curiosity, intuition, and epistemic agency. As a result, children often construct inaccurate models of science phenomena, such as assuming that sweaters are heat generation sources rather than heat retention tools. In school, they rely on teacher support in refining intuitively constructed models through participation in systematic scientific inquiry (Berland et al., 2016; Gouvea & Passmore, 2017; Osborne, 2014; Stroupe, 2014). Scientific modeling is an instructional approach that aims to fulfill this need as its premise focuses on addressing K-12 students as epistemic agents capable of creating, testing, evaluating, and revising their own models of science phenomena, much like real scientists (National Research Council (NRC), 2012; NGSS Lead States, 2013).

However, K-12 students are rarely offered opportunities to construct and refine their own models of science phenomena. For example, students need support to construct models of concepts such as magnetic force, attraction, and repulsion by manipulating charged particles and/or computer simulations. Instead, they are often provided teacher- or textbook-generated models that they memorize and recall to demonstrate conceptual understanding (Buckley, 2012; Harrison & Treagust, 2000; Schwarz et al., 2009) of magnetic fields. Ultimately, learning *about* models rather than learning *with* models encourages students to become passive learners rather

than active epistemic agents. It prevents students from (a) using models as authentic tools to help them pursue an epistemic goal, (b) developing an accurate understanding of the scientific enterprise, and (c) recognizing themselves as agents capable of conducting systematic scientific inquiry to advance knowledge (Cheng et al., 2014; Hokayem & Schwarz, 2014; Nelson & Davis, 2012; Windschitl, Thompson, & Braaten, 2008).

Students' epistemic agency could be promoted through block-based coding to construct, test, evaluate, and revise simulations that materialize their own scientific models of target science phenomena. This constructionist approach uses simulations as self-expression artifacts (Holbert & Wilensky, 2018; Papert, 1980) that students create in order to externalize existing knowledge, visualize a dynamic representation on the computer screen, reason about science phenomena, and generate more refined models (De Vries et al., 2002; Knuuttila, 2011; Passmore, Gouvea, & Giere, 2014; Renken, Peffer, Otrell-Cass, Girault, & Chiocciariello, 2016; Sengupta et al., 2013). Coding simulations engages students in representational and epistemic practices, which are critical to understanding what scientists do in their profession (Lehrer & Schauble, 2006; NRC, 2008; Sengupta et al., 2013). However, the idea of integrating coding into K-12 science teaching could be overwhelming for most teachers.

In the present study, we researched methods of preparing teachers to integrate coding into scientific modeling lessons. In the following sections, we discuss the theoretical foundations for this study. Next, we present Coding in Scientific Modeling Lessons (CS-Model), which is an instructional module and online tool for scaffolding science teachers' learning to code and integrate coding into scientific modeling lessons. We then report and discuss results of a pilot study in which we examined K-12 teachers' experiences with CS-Model, their perceptions of teaching scientific modeling with coding, and their scientific modeling lessons.

Theoretical Background

Scientific Models and Modeling

Scientific models have often been considered representations that one creates to simply *illustrate* specific features of an entity or system (Craik, 1943; Giere, 1988; Johnson-Laird, 1989). But this definition does not explain what scientific models are for. Constructing a scientific model is motivated by one's epistemic goals to generate knowledge about its referent in real life (Berland et al., 2016; Gouvea & Passmore, 2017; Knuuttila, 2005a, 2011; Osborne, 2014; Suárez, 2003), such as when students use Styrofoam balls to understand and explain the relative distance between planets and the Sun. Similarly, scientists design computer simulations to advance their thinking of phenomena, such as the development of the solar system. Therefore, scientific models are not only representations but also *epistemic tools* that help one fulfill their epistemic goals (Gouvea & Passmore, 2017; Knuuttila, 2005a). In the present study, we adopt a pragmatic perspective to define scientific models as conceptual tools that one designs to simulate, visualize, explain, predict, and/or advance their knowledge about a complex system or phenomenon (Krajcik & Merritt, 2012; NRC, 2012; Samarapungavan et al., 2015; Seel, 2017).

The process of constructing, testing, evaluating, and revising scientific models is named scientific modeling (Gilbert & Justi, 2016; Kim & Oliver, 2018; Nelson & Davis, 2012; Samarapungavan et al., 2015; Schwarz et al., 2009). Scientific modeling is an inherently pragmatic, iterative, and dynamic process. One can design or redesign various models that highlight different features of the same phenomenon to address one's specific epistemic goals (Gilbert & Justi, 2016; Knuuttila, 2005a; Krajcik & Merritt, 2012; Schwarz & White, 2005). For instance, architects design paper-based house blueprints that depict construction specifications (e.g., dimensions, plumbing system) and a three-dimensional interactive model floor plan to help

clients visualize, understand, and make decisions on design options. Architects and clients then engage in a cyclical process of constructing, testing, evaluating, and revising a construction project until it is deemed satisfactory.

Promoting epistemic agency in scientific modeling instruction.

The Next Generation Science Standards (NGSS) challenge passive learning in K-12 school education and emphasize the need to engage students in activities wherein they perform authentic scientific practices, much like real Science, Technology, Engineering, and Mathematics (STEM) professionals (Duschl, 2008; Gouvea & Passmore, 2017; NGSS Lead States, 2013; Stroupe, 2014). Likewise, K-12 teachers are to support students in performing authentic practices and constructing knowledge to solve complex problems (Dass, Head, & Rushton, 2015; Passmore, Schwarz, & Mankowski, 2016; Sneider, Stephenson, Schafer, & Flick, 2014). An underlying premise to such a practice-oriented instructional approach is epistemic agency, which is defined as intentional, mindful, and responsible action taken to advance knowledge (Engel, 2013). The implication of this premise for science instruction is that NGSS learning standards urge educators to address students as epistemic agents, i.e., individuals capable of regulating the rules, methods, and standards of their own learning processes (Elgin, 2013).

In K-12 scientific modeling instruction, teachers are to promote student-driven inquiry (Dass et al., 2015; Kim & Oliver, 2018; Namdar & Shen, 2015; NRC, 2012; NGSS Lead States, 2013; Samarapungavan et al., 2015). Specifically, scientific modeling activities should build upon students' prior knowledge and support them in constructing and refining their own scientific models while pursuing personally-relevant epistemic goals related to a target science phenomenon. This approach is expected to help students develop a conceptual understanding of science concepts and an accurate notion of what it means to conduct authentic scientific

investigations (De Vries et al., 2002; Gouvea & Passmore, 2017; Schwarz, Passmore, & Reiser, 2017).

K-12 teachers unprepared for scientific modeling.

Designing scientific modeling instruction that engages students in authentic scientific inquiry may be perceived by teachers as a daunting task that requires a significant amount of work (Kenyon, Davis, & Hug, 2011; Stroupe, 2014). Research indicates that both preservice and in-service teachers need support to further develop their content and pedagogical content knowledge of scientific models and modeling. For instance, teachers often understand models as tools that simply represent reality (Driel & Verloop, 1999; Krell & Krüger, 2016) and as the *only* correct form to explain a phenomenon (Justi & Gilbert, 2002). Some perceive modeling as a synonym for scientific inquiry (Windschitl et al., 2008) or as a method to assess learning (Schwarz & Gwekwerere, 2007). Teachers have acknowledged that they lack understanding of scientific inquiry and scientific reasoning skills (Stammen, Malone, & Irving, 2018; Zhang, Parker, Koehler, & Eberhardt, 2015). Furthermore, teachers fail to deliver authentic student-driven inquiry, which is key to scientific modeling instruction (Akerson et al., 2009).

Professional learning for both preservice and in-service teachers is critical. Preservice teachers who experience inquiry-based instruction during teacher certification courses tend to use inquiry-based methods once they become in-service teachers (Adamson et al., 2003). K-12 educators need immersive experiences so they engage in authentic scientific modeling practices from a student's perspective and should also receive support to develop exemplary scientific modeling instruction and learning materials (Krell & Krüger, 2016; Reinisch & Krüger, 2018; Stammen et al., 2018; Weiss & Pasley, 2006). During professional learning, teachers need to be regarded as epistemic agents so they construct, manipulate, and develop their own scientific

models while solving complex problems. Such experiences prepare teachers to support their future students in doing the same. To foster teachers' epistemic agency in scientific modeling professional learning, teacher educators can promote and assess teachers' on-task epistemic discourse. Epistemic discourse entails collaborative dialogue about concepts or knowledge underlying a problem with the ultimate goal of solving it (Asterhan, 2013; Asterhan & Schwarz, 2009; De Vries et al., 2002). The present research examined teachers' epistemic discourse during professional learning on scientific modeling and coding, which adds to a scarce literature.

Integrating Simulation Coding into Scientific Modeling

Block-based coding.

Learning to code is critical for twenty-first century learners (K-12 Computer Science Framework Steering Committee, 2016; NRC, 2010) given the increasing impact of technologies in modern society, especially at the workplace. Integrating coding and other computer science skills (e.g., computational thinking, debugging, abstraction) into K-12 education equips students to solve complex interdisciplinary problems and to fill increasing computer-enhanced jobs (Lye & Koh, 2014; National Academy of Engineering (NAE) & NRC, 2014; Wing, 2006). Furthermore, it could raise students' interest in pursuing careers in computer science. In this article, we propose integrating block-based coding into K-12 scientific modeling instruction.

Block-based coding is a programming language created to make learning to program easier and more accessible to novice learners (Maloney, Resnick, Rusk, Silverman, & Eastmond, 2010; Price & Barnes, 2015; Weintrop, 2015). Each block embodies a programming concept (e.g., loops, variables). Block-based coding entails selecting blocks from a palette and stacking them to generate an animated artifact on the screen (Maloney et al., 2010). Contrary to text-based programming, block-based coding does not require memorization and recall of commands and

syntactic rules, which prevents errors (Li & Watson, 2011; Maloney et al., 2010; Price & Barnes, 2015). Learning to code with blocks is perceived by K-12 students as easy and enjoyable (Weintrop & Wilensky, 2015).

Simulation coding in scientific modeling.

Block-based coding is a valuable tool for scientific modeling instruction as it allows learners to create simulations of science phenomena. Simulations are dynamic models that embody rules or underlying mechanisms of a complex process or system (Bowen & Deluca, 2015; de Jong & van Joolingen, 1998; Rutten, van Joolingen, & van der Veen, 2012). Through computer simulations, one can design virtual experiments, test hypotheses, manipulate variables, and advance their thinking (Bowen & Deluca, 2015; Shen, Lei, Chang, & Namdar, 2014; Smetana & Bell, 2012; Wilkerson-Jerde, Gravel, & Macrander, 2015).

Coding simulations of science phenomena as part of scientific modeling instruction embodies constructivist and constructionist learning principles. It entails constructing an artifact based on one's prior knowledge, using it to reason about a concept or phenomenon, and refining the artifact while also developing one's scientific model (Harel & Papert, 1991; Holbert & Wilensky, 2018; Kafai, 2012; Papert, 1980; Sengupta et al., 2013). Simulation coding in scientific modeling instruction should be driven by an authentic epistemic need to generate knowledge about a phenomenon (Renken et al., 2016). To code a simulation, one externalizes their prior knowledge of a phenomenon (Buckley, 2012; Dass et al., 2015; Shen et al., 2014), selects key features (Gouvea & Passmore, 2017; Knuuttila, 2011; Sengupta et al., 2013) to be included in the simulation, and applies coding skills to construct code that materializes such features (Sengupta et al., 2013). For instance, one can use the block *wait*, which embodies the programming concept of delay, to simulate and examine an experiment involving chemical

reaction time. Subsequently, one can execute the simulation to test their hypotheses and code. Once the simulation is played out on the screen, one evaluates the code while comparing the simulation output with their own scientific model and with experiential data, if available (see dotted line in Figure 3.1). If the simulation output accurately represents one's hypotheses and satisfies its creator's epistemic goals, one accepts it as the final product. However, if the output is not satisfactory, then one iteratively performs the construct-test-evaluate-revise cycle until the simulation and code are deemed suitable. The process of coding a simulation of a science phenomenon (Figure 3.1) helps one develop a more refined scientific model.

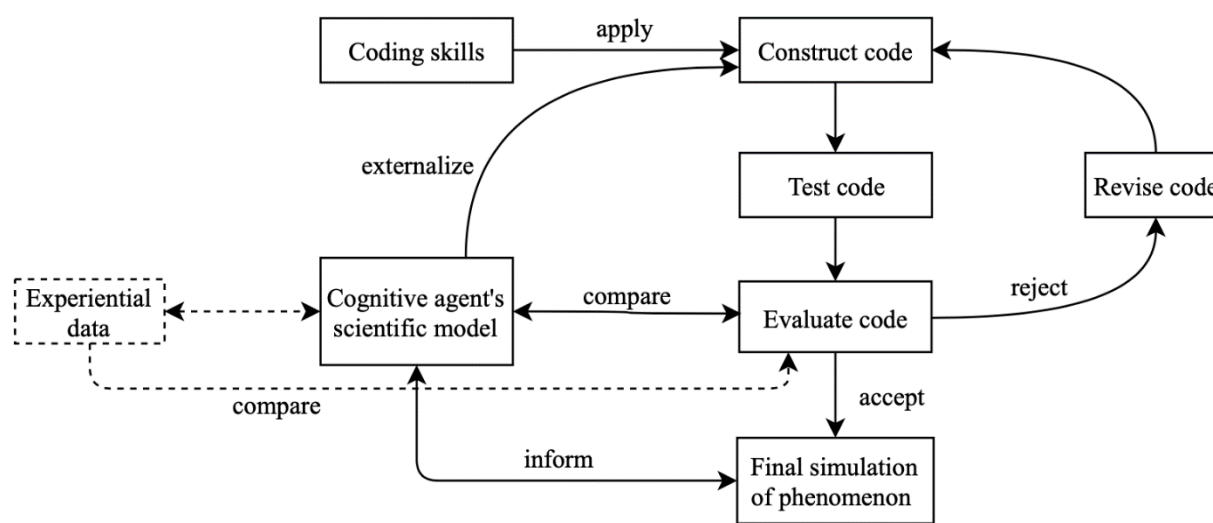


Figure 3.1. CS-Model framework. Source: Vasconcelos and Kim (under review).

Teachers unprepared for teaching with coding.

K-12 educators have been urged to integrate coding and other computer science skills into their lessons (Kim, Oliver, & Jackson, 2016; NAE & NRC, 2014; NRC, 2010, 2012). However, non-computer science teachers need professional learning to do so. Research shows that even certified computer science teachers need professional learning on teaching with coding to K-12 grades (Google & Gallup, 2015a, 2015b; Yadav, Gretter, Hambrusch, & Sands, 2016;

Yadav, Stephenson, & Hong, 2017). It is critical to offer such professional learning to promote not only content knowledge of coding concepts but also pedagogical content knowledge (Weiss & Pasley, 2006) on teaching with coding.

Coding in Scientific Modeling Lessons (CS-Model)

CS-Model is an instructional module and online tool designed to support science teachers in learning to code simulations and use coding to support scientific modeling in their lessons. The design of CS-Model was informed by five guidelines on professional learning on coding and scientific modeling for K-12 educators (Vasconcelos & Kim, under review). The guidelines entail (1) integrating coding simulations into scientific inquiry tasks, (2) adapting coding tasks around specific scientific modeling steps, (3) creating and developing one's own simulations, (4) designing instructional scaffolding strategies to support mindful engagement in both coding and scientific modeling tasks, and (5) implementing assessment methods that target development of coding skills and scientific model progression. An in-depth discussion of guidelines and their theoretical foundations is beyond the scope of the present study. We discuss the CS-Model instructional module and online tool as follows.

The *CS-Model instructional module* includes asynchronous online and face-to-face activities in which teachers participate in scientific modeling activities that involve coding a simulation and conducting experiments in a science lab. Specifically, teachers express their models of water filtration systems by coding simulations that use different filtration materials, as well as conduct physical experiments (Guidelines 1 and 3) in a science lab. Combined use of science simulations with inquiry activities is recommended and effective as found in science learning literature (e.g., D'Angelo et al., 2014; Renken et al., 2016; Smetana & Bell, 2012). The sequence of activities in the CS-Model module was designed and tailored around an adaptation

of Schwarz et al.'s (2009) scientific model progression (Guideline 2), in which teachers (1) anchor phenomenon, (2) construct model, (3) test model, (4) evaluate model, (5) revise model, and (6) use model to predict or explain. Throughout these steps, reflection prompts are provided as scaffolds that help teachers connect coding concepts with specific aspects of the filtration experiment in the simulation (Guideline 4). For instance, teachers verbally answer questions such as “Which blocks represent independent and dependent variables in the water filtration system and how did you code such blocks?” Scaffolds are critical to structure and problematize teachers’ on-task thought processes (Belland, 2014, 2017; Wood, Bruner, & Ross, 1976). Last but not least, the CS-ModeL module features assessments on teachers’ scientific model development and coding skills (Guideline 5), such as screen recordings of simulation coding, audio recordings of peer conversations, and finalized simulations. A detailed description of CS-ModeL weekly activities is presented in the study procedures section.

The *CS-ModeL online tool* (Figure 3.2) was designed to scaffold teachers’ learning to design lessons in which they integrate simulation coding into scientific modeling instruction (Guidelines 1 and 3). The CS-ModeL tool features seven steps that were adapted from Schwarz et al.'s (2009) framework for scientific modeling tasks (Guideline 2). Hints are provided within each step of CS-ModeL to guide teachers on how to design activities that involve simulation coding and scientific modeling tasks. Steps and hints serve as scaffolding mechanisms to structure and problematize (Reiser, 2002, 2004) the lesson design process (Guideline 4). They provide a clear structure for scientific modeling activities and highlight key elements that teachers need to address respectively. Although most CS-ModeL steps are equivalent to Schwarz et al.'s (2009), their accompanying hints were rewritten to include coding tasks in addition to scientific modeling tasks. It is important to highlight that CS-ModeL features *assessment*, an

Scientific modeling steps

Anchor phenomenon

- Introduce driving questions and phenomena for a particular concept. Use a phenomenon that may necessitate using a model to figure it out.
- Explain the concept of scientific models
- Introduce coding as a strategy to create visual representations of scientific models.

Construct model

- Facilitate student reflection about agents, relationships, and interactions in the model.
- Prompt students to use coding to create an initial model expressing an idea or hypothesis.
- Encourage students to debug coding errors.
- Prompt students to discuss the purpose and nature of models.
- Prompt students to explain the model and how they used coding to represent it

Empirically test model

- Design activities in which students investigate the phenomena predicted and explained by the model.
- Design opportunities for peer feedback on model and code.

Evaluate model

- Provide students with opportunities to return to the model and compare with empirical findings and peer feedback.
- Encourage students to identify and discuss criteria for evaluation and revision
- Facilitate reflection on how to improve the model
- Facilitate reflection on how to re-write the code to update the model

Revise model

- Engage students in comparing competing models
- Promote opportunities for constructing/coding a consensus model
- Offer students opportunities to revise change the model and the code to fit new evidence.

Use model to predict or explain

- Design activities in which students apply model and coding representation to predict and explain other phenomena.

Assessment

- Create formative assessment strategies to monitor ongoing student learning/performance in scientific model progression and coding skills, and provide them with feedback
- Create summative assessment strategies to evaluate student learning/performance in scientific model progression and coding skills at the end of the lesson.

Figure 3.2. A screen capture of the CS-Model online tool.

additional step that was not present in Schwarz et al.'s (2009) framework. *Assessment* guides teachers in designing formative and summative methods to assess students' scientific model development and coding skills (Guideline 5).

Research Purpose and Questions

The purpose of this study was to investigate science teachers' (1) epistemic discourse during simulation coding, (2) perception of simulation coding for teaching scientific modeling, and (3) lessons integrating simulation coding into scientific modeling. These research questions guided the study:

RQ1: How do participants engage in epistemic discourse during simulation coding?

RQ2: How do participants perceive simulation coding for scientific modeling instruction?

RQ3: How do participants integrate simulation coding activities into scientific modeling lessons?

Methods

Research Design

This was a single case study. This research design is suitable when participants' experiences are not clearly distinguishable from the context of occurrence (Gagnon, 2010; McMillan & Schumacher, 1997; Yin, 2014). Participants' epistemic discourse, perceptions of simulation coding, and lessons were inherently related to the science teacher education course within which CS-Model was administered. This study contained three embedded units (Baxter & Jack, 2008; Yin, 2014), and each was formed by two participants who worked as a team during face-to-face CS-Model activities. Analysis of each embedded unit as well as the overarching single case generated in-depth description of participants' experiences. Case studies entail in-depth investigation and extensive data collection (Creswell, 2013; Stake, 1995; Yin,

2014). The present research design includes multiple qualitative data sources (Table 3.1) to address each research question, and they are used for triangulation and convergence of findings.

Table 3.1

Data Sources per Research Question

Research Questions	Data Sources	Dana Analysis
RQ1: How do participants engage in epistemic discourse during simulation coding?	- Computer screen recording - Interview transcript	Open coding followed by qualitative framework analysis
RQ2: How do participants perceive simulation coding for scientific modeling instruction?	- Computer screen recording - Video recording - Interview transcript	Open coding followed by qualitative thematic analysis
RQ3: How do participants integrate simulation coding activities into scientific modeling lessons?	- Lessons - Interview transcript	Qualitative framework analysis

Setting and Participants

This study was conducted in a teacher education course offered at a public southeastern university in the United States. The course focuses on methods of science teaching in secondary grades, and it is offered to preservice and in-service teachers pursuing an undergraduate or graduate degree in science education or a science teacher certificate. The course was co-taught by two instructors and entailed weekly face-to-face and asynchronous online activities. Seven students were attending the course, and all of them accepted the invitation to participate in this study. One was excluded from data analysis because the participant missed one face-to-face class meeting when CS-ModeL was implemented. A total of six participants remained (Table 3.2). Despite the small number of participants, this study included diverse student profiles. Four participants were preservice teachers, one was an in-service teacher, and one was a non-degree student pursuing a teacher certificate. Four participants were pursuing a master's degree and one was pursuing a bachelor's degree. On average, participants were 27 years old, and their age

range was 21-38. All but one were female and White. One was male and Asian. Only the male participant had prior programming experience (Python). Pseudonyms assigned to participants are Anne and Maria (team 1), Melinda and Marcus (team 2), and Erica and Kathy (team 3).

Table 3.2

Participant Information

Team	Participant	Teaching Status	Degree	Major
1	Anne	In-service	Master's	Science education
	Marcia	Preservice	Master's	Science education
2	Melinda	Teacher certificate	Non-degree	Science education
	Marcus	Preservice	Undergraduate	Physics and astronomy
3	Erica	Preservice	Master's	Science education
	Kathy	Preservice	Master's	Science education

Study Timeline and Procedures

This study was 4 weeks long and included asynchronous online and face-to-face activities (Figure 3.3). In week 1, informed consent and demographics data were obtained from all individual participants during a face-to-face class meeting. After recruitment, the researcher delivered a 20-minute presentation and guided a whole-class discussion on scientific models, scientific modeling, and coding for K-12 educators. These activities were important to activate participants' prior knowledge and to encourage them to think about coding as a potential teaching tool. In week 2, participants engaged in asynchronous online activities that involved reading an interview on the importance of coding in schools (Merrill, 2017) and reviewing a scientific modeling lesson that used block-based coding (Growing Up Thinking Scientifically (GUTS), n.d.). In week 3, they participated in scientific modeling activities that involved water filter experiments and simulation coding as well as a whole-class discussion on teaching scientific modeling with coding during a 2.5-hour face-to-face class meeting. As homework,

participants used the CS-Model online tool to design a scientific modeling lesson that included simulation coding. In week 4, participants joined individual semi-structured face-to-face or phone interviews. Participation in this study is estimated to have been 5 hours total.

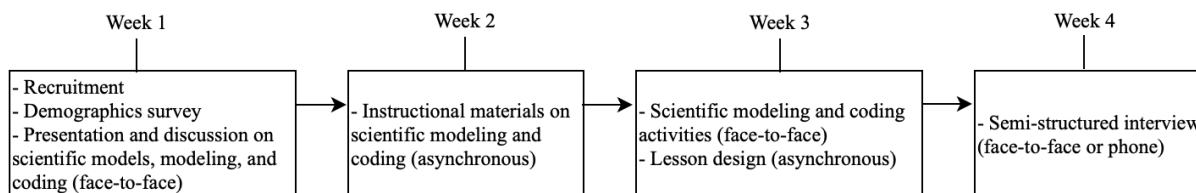


Figure 3.3. Overview of study timeline and procedures.

CS-Model activities in week 3.

Face-to-face activities were divided into three segments within which participants conducted scientific inquiry in a science lab, coded analogous simulations, and joined a whole-class discussion on their perceptions of teaching with coding.

Scientific inquiry.

Scientific inquiry activities were adapted from Kim and Oliver's (2018) scientific modeling lesson on water filter experiments. First, participants were presented with a problem that they had to solve through self-driven inquiry. The problem involved constructing an effective water filter to clear river water at a campground using these available materials: coffee filters, rubber bands, plastic water bottles, cotton, gravel, and activated charcoal in different forms (ground, grains, and pellets). A “polluted” water mix containing tap water, liquid blue food coloring, and a drizzle of olive oil was provided for experiments. The course instructors, the researcher, and participants discussed relevant independent and dependent variables to examine in experiments. Participants collectively decided to examine how filtration materials affect water quality (using color as a proxy), water filtration time, and water input-output ratio.

Next, participants observed the researcher demonstrate five water filter experiments using individual filtration materials. During this demonstration, participants responded to reflection prompts provided by the researcher before, during, and after each experiment. For example, what do you expect the most effective filtration material to be? (before), can you explain what you see regarding water filtration time? (during), and why was so much water retained in the filter? (after). During experiments, participants discussed, explained, and learned about filtration-related concepts such as absorption, adsorption, and porosity. For instance, participants pointed out that (a) cotton yields high input-output ratio of water because cotton is highly absorbent and (b) the smaller the activated charcoal, the slower the water filtration speed and the clearer the water. Subsequently, participants formed teams (pairs) and independently conducted experiments using multiple layers of filtration materials. During experiments, participants *constructed a model* by sharing their understanding of each material's efficiency, *used the model to predict and explain* by predicting and explaining results, *tested the model* by observing or conducting the experiment, and *evaluated the model* by assessing their predictions. During experiments with multiple materials, participants also iteratively *revised the model* until the water filtration result was satisfactory. Lastly, each team presented to class their most effective water filter design as well as their findings.

Simulation coding.

After being introduced to block-based coding in Scratch, participants coded three analogous water filter simulations (Figure 3.4). The researcher created simulation components and code but left blank parameters for when I receive, broadcast, and wait_secs blocks. Such blocks embody the programming commands of independent variables, dependent variables, and delay, respectively. By coding the simulation, participants express relationships between

materials (independent variable) and its resulting water quality (color) and filtration time (dependent variables). In the third simulation, participants coded the additional block *repeat*, which represents the concept of *loops* and reflects water input-output ratio. While coding, teams engaged in epistemic discourse as they discussed how to externalize their models and hypotheses onto the simulation using block-based coding. The three simulations are presented as follows³.

Simulation 1: Participants completed block parameters by selecting (1) cotton, gravel, or activated charcoal for when I receive; (2) dirty, clear, or brackish water for broadcast; and (3) entering numbers in wait__secs to estimate water filtration time. Figure 3.5 shows an example of the code for the cotton sprite (top), the water sprite (bottom), and the simulation storyboard. The code shows that cotton yields brackish water and relatively slow water filtration given the 1-second interval for simulation frames.

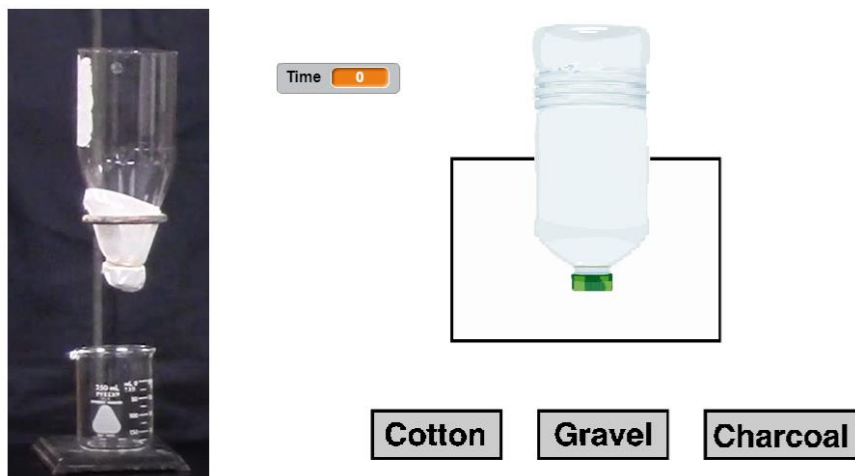


Figure 3.4. Physical water filter experiment and analogous simulation in Scratch.

³ Code and screenshots extracted from participants' computer screen recordings. The code and storyboard are used for illustration only and do not necessarily represent the result of the experiment.

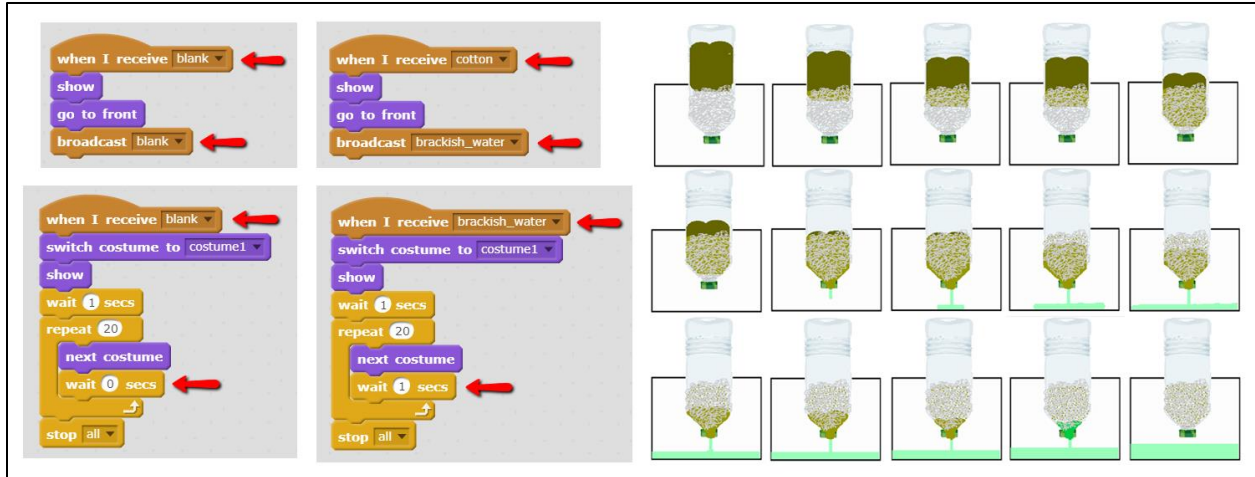


Figure 3.5. Code and storyboard from simulation with cotton.

Simulation 2: Participants completed block parameters by (1) selecting 1/4", 1/2", or 1" charcoal grain sizes for when I receive; (2) choosing dirty, clear, or brackish water for broadcast; and (3) entering a number in wait__secs to estimate water filtration time. Activated charcoal sizes were equivalent to ground, grains, and pellets, respectively. Figure 3.6 shows an example of code for the 1/4" activated charcoal sprite (top), water sprite (bottom), and the simulation storyboard. The code shows that 1/4" (ground) charcoal yields clear water and a very slow filtration time given the 3-second interval for simulation frames.

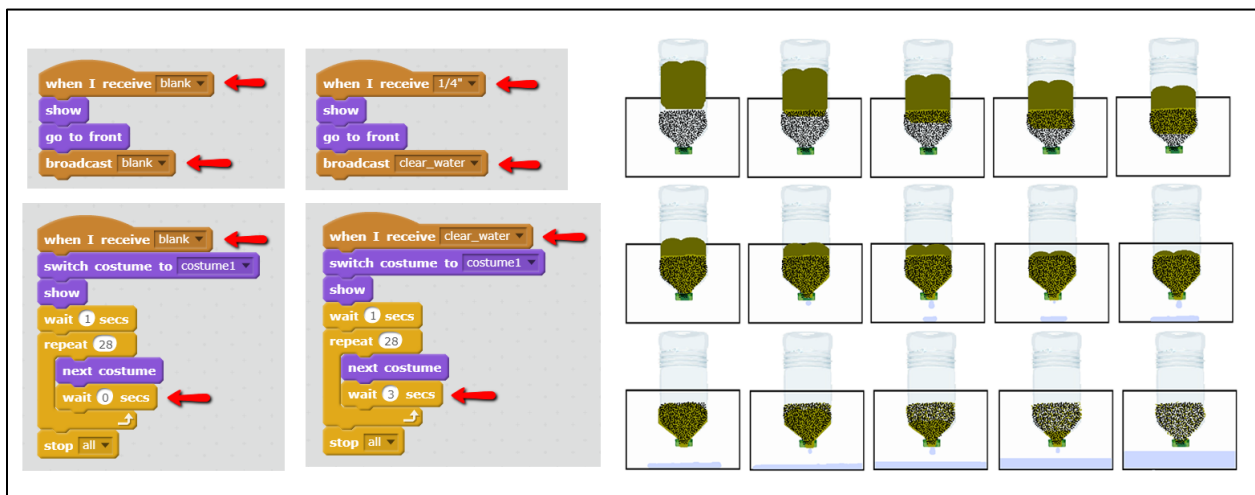


Figure 3.6. Code and storyboard from simulation with 1/4" activated charcoal.

Simulation 3: This was the only simulation in which participants (a) created layers of filtration media in the filter and (b) were instructed to represent the dependent variable water input-output ratio by coding the block repeat. Additionally, this simulation did not showcase 1-1 variable relationships as participants had to construct different combinations. Figure 3.7 shows an example of the code for a simulation with a layer of ground charcoal in between two layers of cotton. Participants completed block parameters by (1) selecting clear, dirty, or brackish water in when I receive; (2) choosing clear, dirty, or brackish water in broadcast; (3) entering a number for wait__secs to estimate water filtration time; and (4) entering a number for repeat to determine how many times filtration frames are executed. The block repeat represents input-output ratio of water. Based on the code, cotton-ground charcoal-cotton yields brackish water, a relatively slow water filtration given the 2-second interval for simulation frames, and high input-output ratio (water retention) as only 15 out of 26 frames were executed.

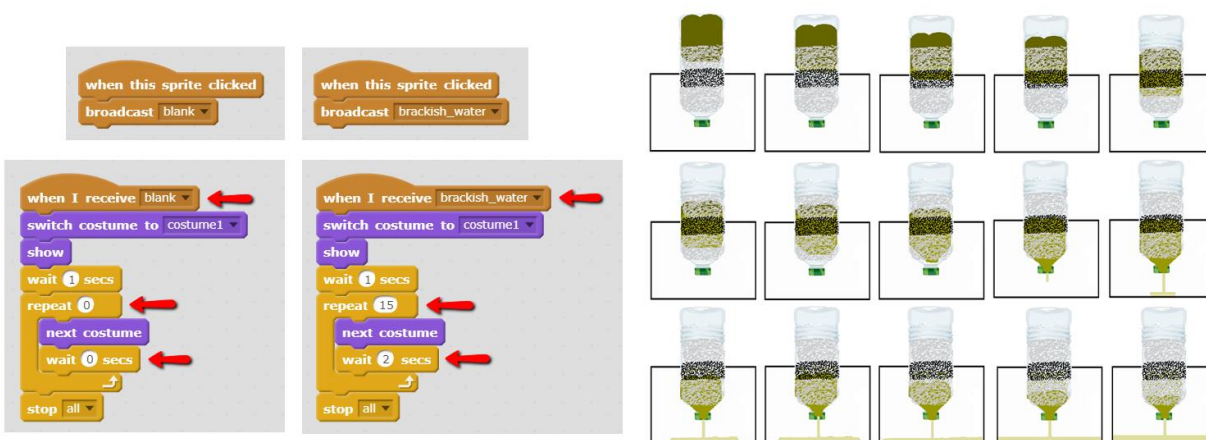


Figure 3.7. Code and storyboard from simulation with cotton-ground charcoal-cotton.

Whole-class discussion.

Subsequent to coding activities, the researcher and course instructors led a whole-class discussion on coding as a teaching tool within and beyond scientific modeling instruction. Teams shared their experiences and discussed affordances and challenges of teaching with coding.

Data Sources

Several data sources were collected. Teams' computer screens and verbal interactions were video recorded during simulation coding (RQ1 and RQ2) using Screencast-O-matic. Each team's video was 36 minutes long, and all videos add up to 107 minutes. The class discussion on coding for teaching scientific modeling was video recorded (RQ2). The video was 10 minutes long. All participants but Marcus designed a scientific modeling lesson (RQ3). Four participants (Anne, Marcia, Marcus, and Melinda) accepted joining semi-structured interviews (RQ1, RQ2, and RQ3 – Table 3.3), which were 16 minutes long on average and added up to 65 minutes. Interview questions 5, 9, 10, 11, and 12 were borrowed or adapted from Kim, Yuan, Vasconcelos, Shin, and Hill (2017, 2018).

Table 3.3

Overview of Interview Protocol

Research Question	Interview Questions
RQ1: How do participants engage in epistemic discourse during simulation coding?	1. Tell me about your experience coding the water filter simulation. 2. How did you use coding to materialize aspects of the water filter experiment? 3. How was it like working with a partner? 4. Did you make any coding errors?
RQ2: How do participants perceive simulation coding for scientific modeling instruction?	5. What do you think about coding as a teaching tool? 6. What do you think of coding as a skill for K-12 students? 7. How would you use coding to teach scientific modeling?
RQ3: How do participants integrate simulation coding activities into scientific modeling lessons?	8. How did you come up with the lesson idea? 9. Tell me about the coding activities in your lesson. 10. What are your students expected to learn from this lesson? 11. If you could go back to your lesson and change it, what would you do?

Data Analysis

Prior to data analysis, data sources were sanitized and transcribed. A phenomenological stance was adopted to elucidate participants' personal experiences with and perceptions of events (Grbich, 2013; Smith, 2015). Specifically, we examined data sources to generate in-depth accounts of participants' epistemic dialogue during simulation coding, perception of simulation coding for their teaching, and designed scientific modeling lessons.

Data analysis entailed reading transcripts multiple times and jotting down notes that included evidence from data sets along with preliminary interpretations (Saldaña, 2016). Subsequently, the researcher randomly selected one embedded unit, administered open coding techniques to create an initial coding scheme for each research question, and conducted multiple cycles of data analysis and code scheme refinement (Fereday & Muir-Cochrane, 2006; Saldaña, 2016) in NVivo 11. Regarding RQ1, sentences that represented meaningful epistemic practices were individually coded unless more of them were needed to convey a meaningful message. After several cycles of data analysis, it was noteworthy that the coding scheme on participants' epistemic discourse (RQ1) significantly overlapped with Asterhan and Schwarz's (2009). Then, the researcher merged both coding schemes to create an adapted version (Table 3.4) that helps address the research question; after this he performed qualitative framework analysis. The researcher's major advisor reviewed the analysis of transcript excerpts, both reached intersubjective consensus (Kvale & Svend, 2015; Saldaña, 2016), and then the researcher completed the analysis.

Table 3.4

Coding Scheme Nodes, Description, and Example Excerpts

Node	Description	Example Excerpts
Claim	An explanation or proposition about the water filtration experiment or pertinent concepts (e.g., absorption and adsorption properties of filtration materials).	Anne: Charcoal gave us clear water.
Request claim	Request partner to explain a concept, solve a problem, or take a stance regarding one's own explanation.	Erica: The dirtiest water goes with which one?
Agree	Confirm a teammate's claim or explanation is correct without providing additional rationale or justification.	Marcus: Gravel gave us dirty water. Melinda: Yes.
Support	Strengthen one's own claim or explanation by providing additional information, rationale, or justification.	Melinda: Let's say 2 [seconds] because that [charcoal] was the slowest, and cotton was the second slowest. Melinda: It took a long time, the finely ground one, it took so long.
Elaborate	Add new information that strengthens a teammate's claim or explanation so that they collaboratively construct an idea or argument.	Erica: When I receive charcoal pellets... Kathy: Broadcast dirty [water].
Request information	Request teammate to clarify or provide information on a specific claim or explanation.	Marcus: What's this one? [asking teammate how gravel yields water quality]
Repeat idea	Repeat claim or idea that has been previously uttered by a teammate or themselves without adding new information.	Marcia: We connected it so that for the clear water it's the charcoal. Anne: It's the charcoal.
Recall	Remember information, facts, or evidence from water filter experiments that occurred prior to simulation coding activities	Kathy: Charcoal took a while.
Oppose	Disagree with a teammate's claim or explanation without providing a rationale for disagreement.	Erica: The dirtiest water goes with which one? Kathy: Cotton, probably. Erica: Well, brackish.
Challenge	Contest a teammate's claim or explanation by providing rationale or describing a circumstance under which a teammate's claim is not valid.	Melinda: I guess we don't need any [material] that was [yielded] clear [water]. Marcus: Charcoal, the finely ground was clear.

Concession	Admit that a teammate's opposition to one's claim is valid even though the teammate does not provide rationale.	Marcia: Brackish was gravel. Anne: No, dirty was gravel. Brackish was cotton. Marcia: Okay. Okay.
Rebuttal	Weaken a teammate's challenge toward one's own claim by providing further evidence that one's claim is valid.	Not found in data sets.

Qualitative thematic analysis (Boyatzis, 1998; Braun & Clarke, 2006; Fereday & Muir-Cochrane, 2006) was administered on embedded units (RQ1) as well as the overarching case (RQ1 and RQ2). Thematic analysis is a “method for identifying, analyzing, and reporting patterns (themes) within data” (Braun & Clarke, 2006, p. 6). Results from embedded unit data analysis shed light onto participant's experiences within each team. Next, overarching themes were developed (Ayres, Kavanaugh, & Knafl, 2003; Creswell, 2013) to explain and compare teams' overall experience. Each theme is justified by converging evidence from different data sources (Fereday & Muir-Cochrane, 2006; Tracy, 2010). Developing themes was an iterative process that involved moving back and forth from data sources to theme write-up (Braun & Clarke, 2006; Clarke & Braun, 2013).

Open coding techniques were used to assess participants' perceptions of coding (RQ2). A rubric on lesson design created by (Kim et al., 2015) was modified to address scientific modeling teaching (Table 3.5) and used to evaluate how participants integrated simulation coding into scientific modeling (RQ3). Listed criteria are (a) subject inclusion; (b) topic inclusion; (c) NGSS learning standard inclusion; (d) learning objective inclusion; (e) activity description; (f) alignment between learning standards, objectives, and activity description; (g) simulation coding; (h) authenticity in scientific modeling tasks; and (i) integrated simulation coding into scientific modeling. Each criterion was assessed as 0 (lesson does not meet criterion) or 1 (lesson meets criterion).

Table 3.5

Rubric for Lesson Analysis

Criterion	Description
1. Subject inclusion	Lesson includes subject.
2. Topic inclusion	Lesson includes topic
3. NGSS learning standard inclusion	At least one NGSS learning standard is included
4. Learning objective inclusion	At least one related learning objective is included
5. Activity description	Description of activities is in-depth or vague
6. Alignment: learning objectives, standards, and activity description	Learning objectives and standards are aligned with activity description
7. Inclusion of simulation coding	Simulation coding is included in the lesson
8. Authenticity in scientific modeling tasks	Scientific modeling tasks are authentic, i.e., engage students in solving complex problems that require application of practices performed by professionals in the real world such as collecting and analyzing data
9. Integrated scientific inquiry and simulation coding	Lesson combines scientific inquiry and simulation coding activities

Results and Discussion**RQ1 on Epistemic Discourse****Anne and Marcia (team 1): Low self-efficacy.**

Anne and Marcia's epistemic discourse revealed low self-efficacy regarding use of computers and coding as evidenced by utterances such as "I'm a computer idiot" (Anne), or "I am apparently just too dumb for this one" (Marcia), which expressed their insecurities. While coding, they were afraid of breaking the simulation or accidentally deleting its parts. The team felt nervous every time they encountered an error, or the simulation did not work as expected:

Anne: You broke it. You did everything right. What happened?

Marcia: What the heck? I'm blaming the costume on this.

[Participant repeatedly clicks several buttons but simulation does not work.]

Anne: We don't know what we did, but we did something wrong.

Anne and Marcia asked the researcher for validation several times even when their hypotheses on the water filtration concepts and simulation coding ideas were correct. The researcher offered prompts such as “Which independent-dependent relationships do you want to simulate?”, but participants wanted approval prior to coding and testing the simulation. As a result, they displayed significantly fewer instances of epistemic discourse compared to other teams. It appears that Anne and Marcia’s low self-efficacy towards coding influenced the amount and quality of peer-to-peer interactions. During the interview, both participants mentioned that it took them a long time to be comfortable with the activity, but they were able to successfully complete it. Anne explained that working with a partner entailed “discussing and figuring out what really happened” during the experiment while Marcia described it as thinking “about the different parts and how everything works separately before you put it together.” The excerpt below shows the team collaborating:

Marcia: So clear was the quarter [charcoal size], right?

Anne: The smallest one, yeah.

Marcia: And it took forever [for water] to get through. So like 3 [seconds] [enters 3 seconds to wait_secs block]?

Anne: Three is good.

Marcia: Sure.

Anne: We’re learning. We’re doing really good on this one.

Melinda and Marcus (team 2): Peer tutoring.

At the beginning of the coding activity, Melinda showed negative attitudes towards coding, which Marcus addressed at different times by showing that block-based coding is easy, and reinforcing that Melinda is capable of doing it. Given his previous experience with

programming, Marcus adopted a tutor role wherein he interrupted the dialogue several times to explain coding concepts (e.g., conditionals) and Scratch vocabulary (e.g., costumes, sprites) to Melinda. Once Melinda acknowledged that she understood a concept, they would automatically switch back to discussing water filtration concepts and how to code the simulation, as shown below:

Melinda: We need to slow it [filtration] down more. Right?

Marcus: Yes, 1 second.

Melinda: What's a costume?

Marcus: Things that change with the animation [shows sprite costumes with mouse].

Melinda: Oh okay.

Melinda: So we made it clear [water] with cotton?

Melinda was not intimidated by the difference in coding expertise, and she engaged in productive discussions with Marcus about water filtration design. During the interview, Melinda said that they were often “on the same page” and that it was helpful that Marcus knew coding beforehand; she described this as having a “tutoring session.” Both participants valued the experience of pair coding as they collaboratively “rebuilt the experiment through coding, and it makes the connection [knowledge of water filtration] a little stronger” (Marcus).

Erica and Kathy (team 3): Computer operator.

Erica and Kathy's epistemic discourse evolved depending on who operated the computer to code the simulation (Table 3.6). When Erica was coding, she took a leadership role in reiterating and reflecting about water filtration experiments and variables while Kathy showed a more passive role in mostly agreeing or elaborating on her peer's statements. Once participants switched places so that Kathy would operate the computer, Kathy and Erica equally contributed

to the discussion on science concepts and simulation coding. Instead of simply agreeing with her partner, Kathy's participation was more meaningful as she was able to actively co-construct knowledge.

Table 3.6

Compared Epistemic Discourse in Erica and Kathy's Team

Erica Operates Computer	Kathy Operates Computer
Erica: Okay so click on the charcoal.	Erica: So 1 inch. I'd say click on that.
Kathy: Yeah.	Kathy: Of yeah this is the one. When I receive
Erica: So it's [water] just going through.	1 inch.
Kathy: Yeah.	Erica: Broadcast dirty.
Erica: And we'll do less than 1 second for the other ones.	Kathy Half inch, hide. A quarter inch hides too.
Kathy: Yeah, that's good.	

Theme 1: Lingering misconceptions.

During pair simulation coding, teams shared and compared their models of how each water filter design would perform. Although teams had vivid memories of the physical water filter experiments, participants still had lingering misconceptions, which they corrected through epistemic discourse with their peers. The excerpt below shows Kathy's claim that using cotton in the water filter yielded one of the fastest filtration times compared to other materials. Erica challenged that idea, saying that cotton absorbed water and only released it after a while when cotton was entirely soaked. Kathy then conceded and proposed to increase delay between filtration frames to accurately represent filtration time.

Erica: So do you think that's too long [water filtration time in simulation]?

Kathy: Cotton was one of the fastest ones.

Erica: No, it [water] soaked into the cotton and then came...

Kathy: Yes, let's try 10 seconds and see how that does.

During the interview, Marcia explained that “working with a partner was helpful because she had ideas that were slightly different than mine and thought about things that I hadn't thought about.” Similarly, Marcus highlighted that working with Melinda was beneficial because he used his coding expertise to help her while Melinda contributed with knowledge of water filter experiments. In Marcus’s words, “some of the things that she pointed out, I wouldn't have at first gotten it, (...) and I was able to help her, she was able to help me.”

Through pair coding, participants engaged in epistemic discourse, collaborative work, and creation of a consensus model, i.e., a mutual understanding of the science phenomenon (Kim & Oliver, 2018; Samarapungavan et al., 2015) in order to code the simulation. These findings align with De Vries, Lund, and Baker's (2002), which highlight the benefit of computer-mediated epistemic dialogue for development of a more refined conceptual understanding of science phenomena. Similarly, Sengupta et al. (2013) showed students’ learning gains from one-on-one scaffolds while coding simulations on ecology and kinematics concepts. Collaborative work is effective and should be integrated into scientific modeling instruction as it fosters joint reflection, shared learning objectives (Kanno, Furuta, & Kitahara, 2010), and epistemic discourse. Consequently, collaborative work leads to correction of lingering misconceptions and enhanced conceptual understanding of target phenomena.

Theme 2: Support for error debugging.

All teams struggled to debug code errors when simulations did not work as expected. Examples of code errors include leaving block parameters blank, not establishing variable relationships correctly, and entering the wrong parameters for specific blocks. Though a detailed account of error debugging is beyond the scope of the present study, it was noticeable that participants mostly employed inattentive strategies such as clicking around, briefly reviewing the

code, and asking the researcher for help. Marcus explained that his team fixed errors using “trial and error, putting in different times, [and] seeing what the result was,” and Anne said her team’s strategy was “asking for help, we called everyone over there at least once.” After encountering an error, teams did not form and test hypotheses and/or review the code to identify the cause of the error. This echoes Kim et al.’s (2017, 2018) (a) findings that preservice teachers struggle to mindfully debug errors during robot programming and (b) recommendation for additional training and scaffolds on debugging.

Lack of debugging skills influenced teams’ epistemic dialogue as participants had to interrupt conversations about key aspects of water filtration systems to discuss errors or ask for help. Most often, teams did not resume the conversation from where they stopped. As Anne explained, “trying to figure out how to work the program was distracting because you had to spend time figuring that out before you could then start figuring out how to break apart the experiment and [...] translate that into the program.” Future practice and research on learning to code should include training and scaffolds to support teachers’ code error debugging such as a list of common errors and debugging strategies, question prompts to document the debugging trail, and observation of experts coding and debugging (Kim et al., 2018). Such strategies would facilitate debugging without interfering in teachers’ productive struggle (Hiebert & Grouws, 2007; Warshauer, 2015) while learning to code.

Theme 3: Low conflict argumentation.

Analysis of teams’ epistemic discourse showed that an overwhelming majority of coded sentences entailed epistemic practices that represent agreement with a teammate’s statement or co-construction of knowledge. Specifically, teams’ epistemic dialogue mostly included making a claim, requesting a claim, agreeing with a teammate’s claim, elaborating on a teammate’s claim,

and recalling specific aspects or outcomes of physical water filter experiments. All teams displayed low levels of conflict and argumentation through epistemic practices such as opposing or challenging a teammate's idea, conceding to a teammate's explanation, or providing a counterargument (rebuttal).

Low conflict argumentation across teams is partly justified by the absence of scaffolds to structure and promote epistemic argumentation. Teams' dialogue was unstructured and constantly alternated between discussion about coding and about water filtration processes. De Vries et al. (2002) list several possible reasons that lead to communication barriers such as willingness to engage in conflict argumentation, preparation to manage "interpersonal social conflict" (p. 99), or an understanding of how to engage in argumentation. Further investigation on effective ways to promote teachers' epistemic dialogue during coding is needed so that preservice and in-service teachers can engage their future students in productive argumentation (Kaya, 2013), as well as identify and address students' misconceptions through epistemic dialogue (Abi-El-Mona & Abd-El-Khalick, 2006).

RQ2 on Perception about Coding

Theme 1: Teachers need assistance to teach with coding from early grades.

Analysis of classroom discussion and interviews revealed that all participants emphasized that K-12 students should learn to code from early grades. As Marcia explained, coding is "the direction to which our society is going," it is "something students might do as a future job," and it is "helpful in STEM, blending science and technology." However, participants also voiced concerns about teaching with coding. First, teachers "need considerable training to be able to teach it" (Melinda). Second, teachers need dedicated time to practice coding to confidently integrate it into their instructional settings and support students' learning to code. Last, non-

computer science teachers need support from the district, school, and fellow computer science teachers within their schools to successfully integrate coding into their lessons. For instance, schools can encourage teachers to “connect different classes, like computer science and science” (Erica and Marcia), so that teachers with different backgrounds can co-offer meaningful and interdisciplinary STEM instruction.

Participants’ claims are not unsupported. Calls have been made to offer professional learning on how to integrate coding and other computer science skills into K-12 education to preservice and in-service teachers across subjects and grades (Gal-Ezer & Stephenson, 2010; Google & Gallup, 2015; Obama, 2016). As a result, resources for teachers are growing in number, such as the K-12 computer science framework (K-12 Computer Science Framework Steering Committee, 2016), free online courses on computer science concepts (Code.org, n.d.; Google, n.d.), and exemplary lessons that use coding (Code.org, n.d.; Computer Science Teachers Association (CSTA) & International Society for Technology in Education (ISTE), 2011; GUTS, n.d.). But in addition to learning resources, teachers need professional learning and dedicated time for learning to code. In future research, a portfolio of lessons targeting topics from various subdomains of science instruction (e.g., life sciences, physics, chemistry) could be included. Lessons should be accompanied by simulations that teachers could use for practicing coding, a step-by-step simulation design guide, and ideas for supporting students’ learning to code (e.g., reflection prompts to promote epistemic discourse). Partnerships with computer scientists or industry professionals (e.g., Granor, DeLyser, & Wang, 2016; Papini, DeLyser, Granor, & Wang, 2017) could also benefit K-12 educators in terms of exchanging content and pedagogical knowledge on teaching with coding.

Theme 2: Coding science simulations offers various benefits.

When questioned about their coding experience, participants identified three main benefits of simulation coding. First, completing scientific inquiry activities leading to coding activities “helps connect and facilitates transferring it [knowledge] over to the coding project” (Marcia). Marcia and Marcus explained that coding simulations is a safe, low-cost approach if the school “doesn’t have necessary equipment” or if the experiment is dangerous for first-hand exposure (e.g., investigate how weight affects acceleration using free-falling objects). Other participants highlighted that scientific inquiry and coding supplement each other and reinforce learning of target concepts. Second, participants believe that simulation coding promotes critical thinking as one needs to “figure out the experiment, break that apart, and then mirror that into the coding” (Marcia). Melinda elaborated on that idea by saying that coding “requires problem solving, sequencing things, and putting things together, so it’s a lot of learning skills. I think you get a lot more out of it than ‘I know how to code.’” Last, coding allows modeling of microscopic and macroscopic phenomena. When asked about examples of topics they would teach with coding, every participant mentioned unobservable phenomena. For instance, Anne would teach osmosis and Marcus would teach acceleration on Earth vs. on the Moon.

RQ3 on Simulation Coding in Scientific Modeling Lessons

Lesson features: All lessons included a subject, topic, at least one NGSS learning standard, and a learning objective. Although not required, Marcia attempted to include a computer science learning objective — “to become familiar with coding” — though it does not communicate a learner’s measurable performance (Anderson, Krathwohl, & Bloom, 2001).

Activity description: Marcia and Anne provided in-depth descriptions of scientific modeling activities while Melinda, Kathy, and Erica’s lessons lacked information. For example,

Erica refers to supporting students in revising their models as “give students time to apply changes.”

Alignment: most lesson activities aligned with learning standards and objectives. For instance, Erica’s students were expected to develop models to explain geoscience processes (e.g., earthquakes, volcanoes, surface weathering, and deposition). To do such, her students code simulations of plate tectonics, which she expects to be helpful in explaining and predicting volcano eruptions. Marcia’s was the only lesson with poor alignment. Her students were to “develop a model to describe the cycling of Earth’s materials” though she proposed hosting a guessing game wherein students identify minerals based on key properties.

Inclusion of simulation coding: All participants but Anne included simulation coding activities in their lesson. For instance, students were to code simulations of respiratory, circulatory, and digestive systems in Melinda’s lesson to explain how system components function. Marcia’s lesson included in-depth description of strategies to support code error debugging, which are “reviewing the code, documenting actions, and creating hypotheses.” This shows she understands the importance of conducting mindful, hypotheses-driven debugging and code revision (Kim et al., 2017, 2018). In her description of how students will construct a model, she explained that they will “look up properties of minerals” to create a mystery mineral guessing game using coding. Conversely, Anne designed activities wherein students simply visualize a simulation to explain cell osmosis.

Authenticity in scientific modeling tasks: The concept of authenticity is herein regarded as representative of practices and challenges experienced in real-world problem solving by STEM professionals. All but Anne’s lesson failed to (a) present a complex problem that students would have to solve, (b) account for students’ hypotheses or prior knowledge, (c) describe

student-driven tasks, or (d) list pedagogical strategies to promote personalized learning. In Kathy's lesson, for example, students need to tie together pictures of animals and plants using a string to create a food web with producers and consumers. In Marcia's lesson, students played a guessing game to identify minerals based on specific features such as hardness and streaks. Conversely, Anne designed a lesson that involved scientific inquiry. Students were to investigate how concentration of salt affects mass and diameter of shell-less eggs and then use empirical data to explain osmosis and osmotic movement. Anne's lesson also featured group work and peer feedback, students' evaluation of their own hypotheses, and production of a lab report detailing experiment design, data collection procedures, analysis methods, and conclusions.

Such findings are not surprising. First, Anne is the only in-service teacher and has more teaching experience than others. No other participant had previous teaching experience and all were taking their first science teaching methods course. And second, these findings are supported by previous studies, which indicate preservice teachers' limited understanding of scientific inquiry and scientific modeling (Schwartz et al., 2004; Schwarz, 2009; Windschitl et al., 2008) and highlight the need for training on scientific inquiry lesson design (Lederman, Schwartz, Abd-El-Khalick, & Bell, 2001; Yoon, Joung, & Kim, 2012). Participants were instructed to include simulation coding in their designed lessons. One may conjecture that participants without authentic scientific inquiry in their lesson may have had to lessen complexity in science learning activities to the required inclusion of simulation coding activities. Anne's lesson, without simulation coding, exhibited complexity with authentic scientific inquiry. It is not possible to draw conclusion in this regard because this study did not examine participants' lesson design process. Further investigation is needed.

Table 3.7

Lesson Data Analysis

Criterion	Anne	Marcia	Melinda	Marcus	Erica	Kathy
1. Subject inclusion	✓	✓	✓	✓	✓	✓
2. Topic inclusion	✓	✓	✓	✓	✓	✓
3. NGSS learning standard inclusion	✓	✓	✓	✓	✓	✓
4. Learning objective inclusion	✓	✓	✓	✓	✓	✓
5. Activity description	✓	✓				
6. Alignment: learning objectives, standards, and activity description	✓		✓	✓	✓	✓
7. Inclusion of simulation coding		✓	✓	✓	✓	✓
8. Authenticity in scientific modeling tasks	✓					
9. Integrated scientific inquiry and simulation coding						

Integrated scientific inquiry and simulation coding: No participant designed lessons that meaningfully integrated simulation coding into authentic scientific inquiry. In fact, lessons either focused on coding *or* scientific modeling, but not both. For example, Erica’s lesson mostly listed what students are to do with coding blocks (e.g., use code to construct simulation, revise code) while the description of scientific modeling tasks was superficial. In her lesson, she wrote that students would construct a model of plate tectonics and would use “pre-made sprites for each of the interaction types and allow students to build the code themselves.” Moreover, simulation coding is the only instructional strategy used in this lesson, which has been discouraged in the science learning literature given mixed results in empirical studies (e.g., D’Angelo et al., 2014; Renken et al., 2016; Smetana & Bell, 2012).

Melinda, Marcia, and Kathy wrote their lessons focusing on scientific modeling tasks and ended up with superficial descriptions for coding activities. For instance, Melinda wrote that students will evaluate their models of body systems by “working in groups to discuss the 3 human body system codes they put together.” Her lesson did not offer other details about such

tasks such as strategies to support students in evaluating specific elements of their models. Anne, on the other hand, did not include simulation coding in her lesson at all.

Conclusions and Directions for Future Research

This qualitative single case study examined five preservice and one in-service science teachers' epistemic discourse while coding simulations of water filtration systems, perceptions of coding for teaching scientific modeling, and the scientific modeling lessons they designed using simulation coding.

Analysis of participants' epistemic discourse revealed that participants discussed key aspects and results of water filtration experiments (e.g., porosity of materials and their influence on water quality), shared models and hypotheses with teammates, and brainstormed how to materialize their models onto the simulation using block-based coding. As a result, participants corrected lingering misconceptions that they still possessed after conducting physical experiments. Findings also pointed out that participants' lack of debugging skills detracted from their ability to maintain engagement in epistemic discourse. There were several instances when participants felt overwhelmed by the fact that simulations did not work as expected. As they tried to debug code errors, it was noticeable that they adopted inattentive debugging strategies such as random trial and error or asking for help. Another interesting finding on participants' epistemic discourse is that most of their utterances embodied agreement and co-construction of knowledge, and there were few instances of conflict and disagreement. Scaffolds to guide teachers in sharing conflicting perspectives, managing conflict, and engaging in productive argumentation leading to a consensus model can be featured in future studies. Scaffolds could also be designed to support error debugging through hypothesis generation, documentation of performed actions, and identification of cause of error.

An investigation of teachers' perceptions of coding for teaching scientific modeling revealed that they acknowledge the importance of integrating coding and other computer science concepts into K-12 education although doing it without support from districts, schools, and computer science colleagues is unfeasible. Teachers also believe that coding should be taught from early childhood, which they assume requires extensive teacher training.

Regarding participants' designed lessons, most participants failed to (a) design authentic scientific inquiry and (b) provide meaningful descriptions of activities that target both coding and scientific modeling. Additional support strategies such as a diagram or a concept map could be integrated into future studies with CS-Model so that teachers can connect target science concepts, code blocks, computer science concepts, target coding skills to be learned by students, tasks that students are expected to perform, and rationale on how such activities promote (a) epistemic agency and (b) help students externalize and further develop their scientific models.

Study Limitations

This study is a stepping stone for researchers and practitioners striving to integrate coding into STEM teaching and beyond, especially if it involves scientific modeling. Interpretation of results should be cautious given study limitations. First, the study had a small number of participants, most of which were White females. Future studies should target a larger and more diverse population. Second, participants had limited face-to-face time for CS-Model activities given the hybrid format of the teacher education course. Consequently, participants had minimal training on Scratch coding prior to CS-Model. Additional exposure to and practice with coding activities prior to CS-Model activities are recommended. Fourth, teachers' use of coding in scientific modeling instruction was assessed through their lessons rather than actual teaching. Future studies that aim to replicate CS-Model could extend it and observe preservice teachers

teaching their designed lessons. Last, participants' limited lesson design experience partially explains lesson quality. Future iterations of CS-Model could be administered to preservice teachers who are further into their certification program and have had previous experience designing lessons and/or teaching.

References

- Abi-El-Mona, I., & Abd-El-Khalick, F. (2006). Argumentative discourse in a high school chemistry classroom. *Chool Science and Mathematics*, *106*(8), 349–361.
- Adamson, S. L., Banks, D., Burtch, M., Cox, F. I. I. I., Judson, E., Turley, J. B., ... Lawson, A. E. (2003). Reformed undergraduate instruction and its subsequent impact on secondary school teaching practice and student achievement. *Journal of Research in Science Teaching*, *40*(10), 939–957. <https://doi.org/10.1002/tea.10117>
- Akerson, V. L., Townsend, J. S., Donnelly, L. A., Hanson, D. L., Tira, P., & White, O. (2009). Scientific modeling for inquiring teachers network (SMIT’N): The influence on elementary teachers’ views of nature of science, inquiry, and modeling. *Journal of Science Teacher Education*, *20*(1), 21–40. <http://dx.doi.org/10.1007/s10972-008-9116-5>
- Anderson, L. W., Krathwohl, D. R., & Bloom, B. S. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom’s taxonomy of educational objectives*. London, UK: Longman.
- Asterhan, C. S. C. (2013). Epistemic and interpersonal dimensions of peer argumentation: Conceptualization and quantitative assessment. In M. Baker, J. Andriessen, & S. Järvelä (Eds.), *Affective learning together: Social and emotional dimensions of collaborative learning* (pp. 251–271). New York, NY: Routledge/Taylor & Francis Group.
- Asterhan, C. S. C., & Schwarz, B. B. (2009). Argumentation and explanation in conceptual change: Indications from protocol analyses of peer-to-peer dialog. *Cognitive Science*, *33*(3), 374–400. <https://doi.org/10.1111/j.1551-6709.2009.01017.x>

- Ayres, L., Kavanaugh, K., & Knafl, K. A. (2003). Within-case and across-case approaches to qualitative data analysis. *Qualitative Health Research, 13*(6), 871–883.
<http://dx.doi.org/10.1177/1049732303013006008>
- Baxter, P., & Jack, S. (2008). Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report, 13*(4), 544–559.
- Belland, B. R. (2014). Scaffolding: Definition, current debates, and future directions. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (4th ed, pp. 505–518). New York, NY: Springer.
- Belland, B. R. (2017). *Instructional scaffolding in STEM education*. Cham: Springer International Publishing. <https://doi.org/10.1007/978-3-319-02565-0>
- Berland, L. K., Schwarz, C. V., Krist, C., Kenyon, L., Lo, A. S., & Reiser, B. J. (2016). Epistemologies in practice: Making scientific practices meaningful for students. *Journal of Research in Science Teaching, 53*(7), 1082–1112. <https://doi.org/10.1002/tea.21257>
- Bowen, B., & Deluca, W. (2015). Comparing traditional versus alternative sequencing of instruction when using simulation modeling. *Journal of STEM Education: Innovations and Research, 16*(1), 5.
- Boyatzis, R. E. (1998). *Transforming qualitative information: Thematic analysis and code development*. Thousand Oaks, CA: Sage Publications.
- Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. <https://doi.org/10.1191/1478088706qp063oa>

- Buckley, B. C. (2012). Model-based learning. In N. M. Seel (Ed.), *Encyclopedia of the sciences of learning* (pp. 2300–2303). Boston, MA: Springer. Retrieved from http://www.springerlink.com/index/10.1007/978-1-4419-1428-6_589
- Cheng, M., Lin, J., Chang, Y., Li, H., Wu, T., & Lin, D. (2014). Developing explanatory models of magnetic phenomena through model-based inquiry. *Journal of Baltic Science Education, 13*(3), 351–360.
- Chinn, A. C., & Malhotra, B. A. (2002). Epistemologically authentic reasoning in schools: A theoretical framework for evaluating inquiry tasks. *Science Education, 86*(2), 175–218.
- Clarke, V., & Braun, V. (2013). Teaching thematic analysis: Overcoming challenges and developing strategies for effective learning. *The Psychologist, 26*(2), 120–123.
- Code.org. (n.d.). *CS fundamentals unplugged*. Retrieved from <https://code.org/curriculum/unplugged>
- Computer Science Teachers Association, & International Society for Technology in Education. (2011). *Computational thinking for all* (2nd ed.). Retrieved from <https://www.iste.org/explore/articleDetail?articleid=152&category=Solutions&article=Computational-thinking-for-all>
- Craik, K. J. W. (1943). *The nature of explanation*. Cambridge: Cambridge University Press.
- Creswell, J. W. (2013). *Qualitative inquiry and research design: Choosing among five approaches* (3rd ed.). Los Angeles, CA: Sage Publications.
- D'Angelo, C., Rutstein, D., Harris, C., Bernard, R., Borokhovski, E., & Haertel, G. (2014). *Simulations for STEM learning: Systematic review and meta-analysis*. Menlo Park, CA: SRI International.

- Dass, K., Head, M. L., & Rushton, G. T. (2015). Building an understanding of how model-based inquiry is implemented in the high school chemistry classroom. *Journal of Chemical Education*, 92(8), 1306–1314. <http://doi.dx.org/10.1021/acs.jchemed.5b00191>
- de Jong, T., & van Joolingen, W. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, 68(1), 179–201.
- De Vries, E., Lund, K., & Baker, M. (2002). Computer-mediated epistemic dialogue: Explanation and argumentation as vehicles for understanding scientific notions. *Journal of the Learning Sciences*, 11(1), 63–103. https://doi.org/10.1207/S15327809JLS1101_3
- Driel, J. H. V., & Verloop, N. (1999). Teachers' knowledge of models and modelling in science. *International Journal of Science Education*, 21(11), 1141–1153.
- Duschl, R. (2008). Science education in three-part harmony: Balancing conceptual, epistemic, and social learning goals. *Review of Research in Education*, 32(1), 268–291. <https://doi.org/10.3102/0091732X07309371>
- Elgin, C. Z. (2013). Epistemic agency. *Theory and Research in Education*, 11(2), 135–152.
- Engel, P. (2013). Is epistemic agency possible? *Philosophical Issues*, 23(1), 158–178. <https://doi.org/10.1111/phis.12008>
- Fereday, J., & Muir-Cochrane, E. (2006). Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. *International Journal of Qualitative Methods*, 5(1), 80–92.
- Gagnon, Y. (2010). *The case study as research method: A practical handbook*. Québec: Les Presses de l'Université du Québec.
- Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation is critical. *ACM Inroads*, 1(1), 61–66.

- Giere, R. N. (1988). *Explaining science: A cognitive approach*. University of Chicago Press.
- Gilbert, J. K., & Justi, R. (2016). *Modelling-based teaching in science education* (Vol. 9). Switzerland: Springer. <https://doi.org/10.1007/978-3-319-29039-3>
- Google. (n.d.). *Computational thinking for educators [online course]*. Retrieved from https://computationalthinkingcourse.withgoogle.com/course?use_last_location=true
- Google, & Gallup. (2015a). *Images of computer science: Perceptions among students, parents and educators in the U.S.* Retrieved from <http://g.co/cseduresearch>
- Google, & Gallup. (2015b). *Searching for computer science: Access and barriers in U.S. K-12 education*. Retrieved from <http://g.co/cseduresearch>
- Gouvea, J., & Passmore, C. (2017). ‘Models of’ versus ‘models for’: Toward an agent-based conception of modeling in the science classroom. *Science & Education*, 26(1–2), 49–63. <https://doi.org/10.1007/s11191-017-9884-4>
- Granor, N., DeLyser, L. A., & Wang, K. (2016). TEALS: Teacher professional development using industry volunteers. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (pp. 60–65). Memphis, TN: ACM Press. <https://doi.org/10.1145/2839509.2844589>
- Grbich, C. (2013). *Qualitative data analysis: An introduction* (2nd ed). Thousand Oaks, CA: Sage Publications.
- Growing Up Thinking Scientifically. (n.d.). *Water as a shared resource*. Retrieved from https://code.org/curriculum/science/files/CS_in_Science_Module_2.pdf
- Harel, I., & Papert, S. (1991). *Constructionism: Research reports and essays, 1985-1990*. Norwood, NJ: Ablex Publishing.

- Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education*, 22(9), 1011–1026.
<http://dx.doi.org/10.1080/095006900416884>
- Hiebert, J., & Grouws, D. A. (2007). The effects of classroom mathematics teaching on students' learning. In F. K. Lester (Ed.), *Second Handbook of Research on Mathematics Teaching and Learning* (pp. 371–404). Charlotte, NC: Information Age Publishing.
- Hokayem, H., & Schwarz, C. (2014). Engaging fifth graders in scientific modeling to learn about evaporation and condensation. *International Journal of Science and Mathematics Education*, 12(1), 49–72. <https://doi.org/10.1007/s10763-012-9395-3>
- Holbert, N., & Wilensky, U. (2018). Designing educational video games to be objects-to-think-with. *Journal of the Learning Sciences*, 1–41.
<https://doi.org/10.1080/10508406.2018.1487302>
- Johnson-Laird, P. N. (1989). Mental models. In M. I. Posner (Ed.), *Foundations of cognitive science* (pp. 469–499). Cambridge, MA: The MIT Press.
- Justi, R. S., & Gilbert, J. K. (2002). Science teachers' knowledge about and attitudes towards the use of models and modelling in learning science. *International Journal of Science Education*, 24(12), 1273–1292. <https://dx.doi.org/10.1080/09500690210163198>
- K-12 Computer Science Framework Steering Committee. (2016). *K–12 computer science framework*. Retrieved from <http://www.k12cs.org>
- Kafai, Y. (2012). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 35–46). New York, NY: Cambridge University Press.
- Kanno, T., Furuta, K., & Kitahara, Y. (2010). A model of team cognition based on mutual beliefs. *Theoretical Issues in Ergonomics Science*, 14(1), 38–52.

- Kaya, E. (2013). Argumentation practices in classroom: Pre-service teachers' conceptual understanding of chemical equilibrium. *International Journal of Science Education*, 35(7), 1139–1158. <https://doi.org/10.1080/09500693.2013.770935>
- Kenyon, L., Davis, E. A., & Hug, B. (2011). Design approaches to support preservice teachers in scientific modeling. *Journal of Science Teacher Education*, 22(1), 1–21. <https://doi.org/10.1007/s10972-010-9225-9>
- Kim, C., Kim, D., Yuan, J., Hill, R. B., Doshi, P., & Thai, C. N. (2015). Robotics to promote elementary education pre-service teachers' STEM engagement, learning, and teaching. *Computers & Education*, 91, 14–31. <https://doi.org/10.1016/j.compedu.2015.08.005>
- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2017). Prospective elementary teachers' debugging during block-based visual programming. Presented at the American Educational Research Association (AERA) Annual Meeting, San Antonio, TX.
- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, 46(5), 767–787. <https://doi.org/10.1007/s11251-018-9453-5>
- Kim, E., Oliver, J. S., & Jackson, D. F. (2016). Connecting the imperatives of STEM, NGSS, deep learning and assessment: A conceptual paper. Presented at the National Association for Research in Science Teaching, Baltimore, MD.
- Kim, Y., & Oliver, J. S. (2018). Supporting preservice teachers' use of modeling: Building a water purifier. *Innovations in Science Teacher Education*, 3(1), 1–14.
- Knuuttila, T. (2005a). *Models as epistemic artefacts: Toward a non-representationalist account of scientific representation*. Helsinki, Finland: University of Helsinki.

- Knuuttila, T. (2011). Modelling and representing: An artefactual approach to model-based representation. *Studies in History and Philosophy of Science*, 42(2), 262–271.
<https://doi.org/10.1016/j.shpsa.2010.11.034>
- Krajcik, J., & Merritt, J. (2012). Engaging students in scientific practices: What does constructing and revising models look like in the science classroom? *The Science Teacher*, 79(3), 38–41.
- Krell, M., & Krüger, D. (2016). Testing models: A key aspect to promote teaching activities related to models and modelling in biology lessons? *Journal of Biological Education*, 50(2), 160–173. <https://doi.org/10.1080/00219266.2015.1028570>
- Kvale, S., & Svend, B. (2015). *InterViews: learning the craft of qualitative research interviewing* (3rd ed). Los Angeles, CA: Sage Publications.
- Lederman, N. G., Schwartz, R. S., Abd-El-Khalick, F., & Bell, R. L. (2001). Preservice teachers' understanding and teaching of the nature of science: An intervention study. *The Canadian Journal of Science, Mathematics, and Technology Education*, 1(2), 135–160.
- Lehrer, R., & Schauble, L. (2006). Cultivating model-based reasoning in science education. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 371–388). New York, NY: Cambridge University Press.
- Li, F. W. B., & Watson, C. (2011). Game-based concept visualization for learning programming. In *Proceedings of the Third International ACM Workshop on Multimedia Technologies for Distance Learning* (pp. 37–42). Scottsdale, AZ: ACM.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. <https://doi.org/10.1016/j.chb.2014.09.012>

- Maloney, J. H., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, *10*(4), 1–15. <https://doi.org/10.1145/1868358.1868363>
- McMillan, J. H., & Schumacher, S. (1997). *Research in education: A conceptual introduction* (4th ed). New York, NY: Longman.
- Merrill, S. (2017). The future of coding in schools. Retrieved from <https://www.edutopia.org/article/future-coding-schools>
- Namdar, B., & Shen, J. (2015). Modeling-oriented assessment in K-12 science education: A synthesis of research from 1980 to 2013 and new directions. *International Journal of Science Education*, *37*(7), 993–1023. <https://doi.org/10.1080/09500693.2015.1012185>
- National Academy of Engineering and National Research Council. (2014). *STEM integration in K-12 education: Status, prospects, and an agenda for research*. Washington, DC: The National Academies Press. Retrieved from <https://doi.org/10.17226/18612>
- National Research Council. (2008). *Taking science to school: Learning and teaching science in grades K–8*. Washington, DC: National Academy Press.
- National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking*. Washington, DC: National Academies Press.
- National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC: The National Academies Press.
- Nelson, M. M., & Davis, E. A. (2012). Preservice elementary teachers' evaluations of elementary students' scientific models: An aspect of pedagogical content knowledge for scientific modeling. *International Journal of Science Education*, *34*(12), 1931–1959. <http://dx.doi.org/10.1080/09500693.2011.594103>

- NGSS Lead States. (2013). *Next Generation Science Standards: For states, by states*. Washington, DC: National Academies Press.
- Obama, B. (2016). *Computer science for all*. Retrieved from <https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all>
- Osborne, J. (2014). Teaching Scientific Practices: Meeting the Challenge of Change. *Journal of Science Teacher Education*, 25(2), 177–196. <https://doi.org/10.1007/s10972-014-9384-1>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Papini, A., DeLyser, L. A., Granor, N., & Wang, K. (2017). Preparing and supporting industry professionals as volunteer high school computer science co-instructors. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 441–446). Seattle, WA: ACM Press. <https://doi.org/10.1145/3017680.3017743>
- Passmore, C., Gouvea, J. S., & Giere, R. (2014). Models in science and in learning science: Focusing scientific practice on sense-making. In M. R. Matthews (Ed.), *International handbook of research in history, philosophy and science teaching* (pp. 1171–1202). Dordrecht, Netherlands: Springer. https://doi.org/10.1007/978-94-007-7654-8_36
- Passmore, C., Schwarz, C. V., & Mankowski, J. (2016). Developing and using models. In C. V. Schwarz, C. Passmore, & B. J. Reiser (Eds.), *Helping students make sense of the world using next generation science and engineering practices* (pp. 109–134). Arlington, VA: NSTA Press. <https://doi.org/10.2505/9781938946042>
- Price, T. W., & Barnes, T. (2015). Comparing textual and block interfaces in a novice programming environment. In *Proceedings of the Eleventh Annual International*

- Computing Education Research* (pp. 91–99). Omaha, NE: ACM Press.
<https://doi.org/10.1145/2787622.2787712>
- Project Growing Up Thinking Scientifically (GUTS). (n.d.). Retrieved from
<http://www.projectguts.org/resources>
- Reinisch, B., & Krüger, D. (2018). Preservice biology teachers' conceptions about the tentative nature of theories and models in biology. *Research in Science Education*, 48(1), 71–103.
<https://doi.org/10.1007/s11165-016-9559-1>
- Reiser, B. J. (2002). Why scaffolding should sometimes make tasks more difficult for learners. In *Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community* (pp. 255–264). International Society of the Learning Sciences. Retrieved from <http://dl.acm.org/citation.cfm?id=1658652>
- Reiser, B. J. (2004). Scaffolding complex learning: The mechanisms of structuring and problematizing student work. *The Journal of the Learning Sciences*, 13(3), 273–304.
- Renken, M., Peffer, M., Otrell-Cass, K., Girault, I., & Chiocciariello, A. (2016). *Simulations as scaffolds in science education*. Cham: Springer.
- Rutten, N., van Joolingen, W. R., & van der Veen, J. T. (2012). The learning effects of computer simulations in science education. *Computers & Education*, 58(1), 136–153.
<https://doi.org/10.1016/j.compedu.2011.07.017>
- Saldaña, J. (2016). *The coding manual for qualitative researchers* (2nd ed). Los Angeles, CA: SAGE.
- Samarapungavan, A., Tippins, D., & Bryan, L. (2015). A modeling-based inquiry framework for early childhood science learning. In K. C. Trundle & M. Saçkes (Eds.), *Research in Early*

Childhood Science Education (pp. 259–277). Dordrecht, Netherlands: Springer.

https://doi.org/10.1007/978-94-017-9505-0_12

Schwartz, R. S., Lederman, N. G., & Crawford, B. A. (2004). Developing views of nature of science in an authentic context: An explicit approach to bridging the gap between nature of science and scientific inquiry. *Science Education*, 88(4), 610–645.

<https://doi.org/10.1002/sce.10128>

Schwarz, C. (2009). Developing preservice elementary teachers' knowledge and practices through modeling-centered scientific inquiry. *Science Education*, 93(4), 720–744.

<https://doi.org/10.1002/sce.20324>

Schwarz, C., Passmore, C., & Reiser, B. J. (2017). *Helping students make sense of the world using next generation science and engineering practices*. Arlington, VA: NSTA Press.

Schwarz, C. V., & Gwekwerere, Y. N. (2007). Using a guided inquiry and modeling instructional framework (EIMA) to support pre-service K-8 science teaching. *Science Education*, 91(1), 158–186.

Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., ... Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners. *Journal of Research in Science Teaching*, 46(6), 632–654. <https://doi.org/10.1002/tea.20311>

<https://doi.org/10.1002/tea.20311>

Schwarz, C. V., & White, B. Y. (2005). Metamodeling knowledge: Developing students' understanding of scientific modeling. *Cognition and Instruction*, 23(2), 165–205.

http://dx.doi.org/10.1207/s1532690xci2302_1

- Seel, N. M. (2017). Model-based learning: A synthesis of theory and research. *Educational Technology Research and Development*, 65(4), 931–966. <https://doi.org/10.1007/s11423-016-9507-9>
- 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>
- Shen, J., Lei, J., Chang, H., & Namdar, B. (2014). Technology-enhanced, modeling-based instruction (TMBI) in science education. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of research on educational communications and technology* (pp. 529–540). New York, NY: Springer New York. https://doi.org/10.1007/978-1-4614-3185-5_41
- Smetana, L. K., & Bell, R. L. (2012). Computer simulations to support science instruction and learning: A critical review of the literature. *International Journal of Science Education*, 34(9), 1337–1370. <https://doi.org/10.1080/09500693.2011.605182>
- Smith, J. A. (2015). Interpretative phenomenological analysis. In J. A. Smith & M. Osborne (Eds.), *Qualitative psychology: A practical guide to research methods* (3rd ed). Birkbeck College, UK: Sage Publications.
- Sneider, C., Stephenson, C., Schafer, B., & Flick, L. (2014). Exploring the science Framework and NGSS: Computational thinking in the science classroom. *Science Scope*, 38(3), 10–15.
- Stake, R. E. (1995). *The art of case study research*. Thousand Oaks, CA: Sage Publications.

- Stammen, A., Malone, K., & Irving, K. (2018). Effects of modeling instruction professional development on biology teachers' scientific reasoning skills. *Education Sciences*, 8(3), 1–19. <https://doi.org/10.3390/educsci8030119>
- Stroupe, D. (2014). Examining classroom science practice communities: How teachers and students negotiate epistemic agency and learn science-as-practice. *Science Education*, 98(3), 487–516. <https://doi.org/10.1002/sce.21112>
- Suárez, M. (2003). Scientific representation: Against similarity and isomorphism. *International Studies in the Philosophy of Science*, 17(3), 225–244.
- Tracy, S. J. (2010). Qualitative quality: Eight “big-tent” criteria for excellent qualitative research. *Qualitative Inquiry*, 16(10), 837–851.
<https://doi.org/10.1177/1077800410383121>
- Vasconcelos, L., & Kim, C. (under review). Coding in scientific modeling lessons (CS-Model). *Under Review in Educational Technology Research & Development*.
- Warshauer, H. K. (2015). Strategies to support productive struggle. *Mathematics Teaching in the Middle School*, 20(7), 390–393. <https://doi.org/10.5951/mathteachmidscho.20.7.0390>
- Weintrop, D. (2015). Blocks, text, and the space between: The role of representations in novice programming environments. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 301–302). Atlanta, GA: IEEE.
- Weintrop, D., & Wilensky, U. (2015). To block or not to block, that is the question: Students' perceptions of blocks-based programming. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 199–208). ACM Press.
<https://doi.org/10.1145/2771839.2771860>

- Weiss, I. R., & Pasley, J. D. (2006). *Scaling up instructional improvement through teacher professional development: Insights from the local systemic change initiative*. CPRE Policy Briefs. Retrieved from https://repository.upenn.edu/cpre_policybriefs/32
- Wilkerson-Jerde, M. H., Gravel, B. H., & Macrander, C. A. (2015). Exploring shifts in middle school learners' modeling activity while generating drawings, animations, and computational simulations of molecular diffusion. *Journal of Science Education and Technology*, 24(2), 396–415.
- Windschitl, M., Thompson, J., & Braaten, M. (2008). Beyond the scientific method: Model-based inquiry as a new paradigm of preference for school science investigations. *Science Education*, 92(5), 941–967. <http://doi.dx.org/10.1002/sce.20259>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100. <https://doi.org/10.1111/j.1469-7610.1976.tb00381.x>
- Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education*, 26(4), 235–254. <https://doi.org/10.1080/08993408.2016.1257418>
- Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM*, 60(4), 55–62.
- Yin, R. K. (2014). *Case study research: Design and methods*. Los Angeles, CA: Sage Publications.

- Yoon, H., Joung, Y. J., & Kim, M. (2012). The challenges of science inquiry teaching for pre-service teachers in elementary classrooms: Difficulties on and under the scene. *Research in Science Education*, 42(3), 589–608. <https://doi.org/10.1007/s11165-011-9212-y>
- Zhang, M., Parker, J., Koehler, M. J., & Eberhardt, J. (2015). Understanding inservice science teachers' needs for professional development. *Journal of Science Teacher Education*, 26(5), 471–496. <https://doi.org/10.1007/s10972-015-9433-4>

CHAPTER 4

CODING SCIENTIFIC MODELS: PRESERVICE TEACHERS' EPISTEMOLOGICAL
UNDERSTANDING, CODING SKILLS, AND LESSON DESIGN⁴

⁴ Vasconcelos, L., & Kim, C. To be submitted to *Instructional Science*.

Abstract

This study implemented the redesigned Coding in Scientific Modeling Lessons (CS-Model), an instructional module and online tool that aimed to support preservice science teachers' use of coding in scientific modeling and in lessons respectively. Participants externalized their models of water filtration systems by coding simulations, constructed and tested water filters, evaluated empirical results against their original models, and revised simulation code to reflect their revised models. Participants also designed lessons to support scientific modeling with coding. This mixed methods study investigated if and how participation in CS-Model affected participants' epistemological understanding of scientific models and modeling along with their understanding of computer science concepts. The study also investigated how participants used coding in their scientific modeling lessons. Results showed that many participants improved their epistemological understanding of models and modeling, as well as their conceptual understanding of computer science concepts after CS-Model. Most participants successfully integrated coding and scientific modeling tasks into their lesson. Participants designed lessons in which coding is used either as a research tool or as an exploration tool, and most lessons targeted computer science practices, but not concepts. Participants' epistemological understanding of models and modeling was not reflected in lesson design. Study limitations and directions for future research are discussed.

Keywords: scientific models, scientific modeling, coding, epistemological understanding, conceptual understanding, lesson design

Introduction

One of the tenets in secondary science education is that teachers need to design scientific modeling instruction in which students generate and develop their own scientific models (Namdar & Shen, 2015; National Research Council (NRC), 2012; Passmore, Schwarz, & Mankowski, 2016). However, students struggle in scientific modeling because they often lack skills in manipulating complex information (Lowe, 2004), constructing conceptual models of target phenomena (Shen, Lei, Chang, & Namdar, 2014), and/or monitoring their own scientific model progression (Hernández, Couso, & Pintó, 2015; Rea-Ramirez, Clement, & Núñez-Oviedo, 2008; White & Frederiksen, 1998).

Using block-based coding in scientific modeling instruction supports use and development of scientific models (Vasconcelos & Kim, 2019). Block-based coding is an appealing strategy to teach programming (Aivaloglou & Hermans, 2016; Price & Barnes, 2015; Weintrop, 2015). In the context of scientific modeling instruction, block-based coding serves as a self-expression medium (Mannila et al., 2014) that enables one to recall, reflect on, and externalize their own models (Lehrer & Schauble, 2000; Papert, 1980) to create animated artifacts such as virtual experiments, tutorials, and games.

Empirical studies that integrate block-based coding into scientific modeling instruction are scarce. The present study addressed this gap by implementing a Coding in Scientific Modeling Lessons (CS-Model) instructional module and online tool in a teacher education course for secondary preservice science teachers. The ultimate goal of this study is to support teachers in designing lessons that meaningfully integrate block-based coding into contexts of science learning, such as scientific modeling instruction.

Relevant Literature

Scientific Models

Scientific models are conceptual tools that embody features of a complex and often unobservable entity (Cheng et al., 2014; Chiou & Anderson, 2010; Chu, Deuermeyer, & Quek, 2017; Seel, 2017), which can be a phenomenon or a system in the natural world. Scientific models are designed to fulfill one's epistemic goals such as making predictions, understanding, explaining, simulating, visualizing, and generating knowledge about a complex phenomenon (Buckley, 2012; Knuuttila, 2009, 2011; Seel, 2017). For instance, the atom model is commonly used by scientists to simplify and investigate interactions among neutrons, protons, and electrons. Scientific models can be physical or conceptual (Buckley, 2000; Gilbert & Boulter, 2000), such as a volcano mockup or the mathematical model for acceleration. Models often include linguistic, symbolic, and/or mathematical components (Harrison & Treagust, 2000; Samarapungavan, Tippins, & Bryan, 2015), such as a written description or the formulae for Newton's motion law.

Scientific models are dynamic entities that one can create or re-create to highlight features of a referent (Grosslight, Unger, Jay, & Smith, 1991; Schwarz et al., 2009) that are relevant for their own epistemic goals (Gouvea & Passmore, 2017; Knuuttila, 2011). For example, maps serve as pictorial models that one designs to investigate similarities or differences across regions such as topography, climate, political relationships, economic trends, and much more. For instance, Bloch, Buchanan, Katz, and Quealy (2018) created an interactive map that helps people investigate results from the 2016 US presidential election at different levels of granularity. It is important to note that maps, as any other model, have limited representational

capabilities as they cannot perfectly represent all features, interactions, relationships, and elements of the referent (Gilbert, 1991; Harrison & Treagust, 2000; Krajcik & Merritt, 2012).

Scientific Modeling

Next Generation Science Standards (NGSS) advise K-12 educators to design school science instruction in ways that mirror authentic professional activity in STEM fields (NRC, 2000, 2012). One of the key ideas in NGSS standards is that students need to develop scientific modeling skills (Kim & Oliver, 2018; NGSS Lead States, 2013). Scientists draw on existing theories and models to construct, test, evaluate, and revise scientific models that help advance their thinking about a target phenomenon (Cheng et al., 2014; Krajcik & Merritt, 2012; Nersessian, 2008; Windschitl, Thompson, & Braaten, 2008a). More specifically, scientists create hypotheses based on an existing model, conduct experiments to test such hypotheses, select tools to identify patterns and interpret datasets, evaluate experiment results against hypotheses and models, craft evidence-based explanations, and draw conclusions to accept or refute hypotheses and models (NRC, 1996; Osborne, 2014). For example, scientists rely on existing knowledge about appetite suppressant substances to test their hypotheses on the effectiveness of commercial food supplements on appetite control and weight loss, as well as potential side effects. Such investigations lead scientists to accept, refine, or reject models that explain which substances promote healthy weight loss, which guides future research and practice.

Constructing Science Simulations with Block-based Coding

Block-based coding is a visual programming language (VLP) (Papadakis, Kalogiannakis, Zaranis, & Orfanakis, 2016; Price & Barnes, 2015) that entails creating a linear sequence of blocks that incorporate programming commands (e.g., conditionals) to control animated behavior on the screen (Basu, Gray, Kelleher, Sheldon, & Turbak, 2017; Weintrop, 2015). Block-based

coding can be instrumental for science learning as it enables one to construct science simulations to model complex phenomena. Simulations are animated models that one designs by applying computational skills (e.g., coding) and knowledge of a target complex referent (e.g. natural selection) to depict a phenomenon, manipulate key variables, and generate knowledge. For instance, coding a simulation can help one better understand, visualize, and/or predict how predators and natural resources such as water and food affect growth of a rabbit population over time.

Coding scientific models.

Using block-based coding to construct artifacts such as simulations is a promising constructivist approach (Blikstein & Wilensky, 2009; Harel & Papert, 1991; Holbert & Wilensky, 2018; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013) for supporting scientific modeling instruction. To code a science simulation, one activates their own models of a science phenomenon, identifies features of interest to be investigated, and associates such features with computer science concepts (e.g., variables, loops) that can materialize them (Sengupta et al., 2013; Wilensky & Resnick, 1999; Xiang & Passmore, 2015). Then one conducts iterative cycles of code creation, testing, evaluation, and revision until the simulation is congruent with one's own scientific model, theories, and/or empirical evidence. The process of designing, manipulating, and visualizing simulations of science phenomena with block-based coding leads to scientific model refinement (Vasconcelos & Kim, 2019).

Epistemology in coding scientific models and epistemological understanding.

Epistemology is an area of philosophy that studies the nature, origin, and scope of knowledge (Boyd et al, 1991; Kelly, McDonald, and Wickman, 2012). Specifically, epistemology focuses on what, how, by whom, and under what circumstances learning occurs. In

the context of science education, learners' epistemologies consist of how they conceptualize knowledge of science phenomena and of science. Learners' conceptualizations affect their learning processes and engagement in scientific practices (Kelly et al, 2012; Hofer, 2001). It is critical to help preservice teachers develop a sophisticated epistemological understanding of models and modeling so they can design scientific modeling instruction that embodies learning experiences grounded on authentic scientific practices.

Coding a science simulation involves generating an artifact that is visually and epistemologically aligned (Holbert & Wilensky, 2018) with a real-world referent. Visual alignment consists of creating a simulation that embodies a certain degree of accuracy and sophistication in representing a referent. Epistemological alignment entails making authentic use of code-based simulations as investigative tools to accomplish epistemic goals (Gouvea & Passmore, 2017; Knuuttila, 2011; Osborne, 2014). Block-based coding serves not only as a tool but also as a workspace wherein one can engage in reflection about scientific concepts (Kim, Oliver, & Jackson, 2016) as well as their own epistemological understanding of what it means to engage in authentic scientific practices (Hokayem & Schwarz, 2014). Coding scientific modeling entails expression of one's models of science phenomena as well as understanding of computer science commands. This approach is expected to help learners develop conceptual understanding of science topics (e.g., osmosis) and computer science concepts (e.g., loops). Conceptual understanding entails not only developing an accurate conceptualization of target constructs, but also developing skills to successfully apply constructs in different contexts (Roth, 1990; Zacharia, 2007; Konicek-Moran & Keeley, 2015), such as using loops in different simulations.

Coding scientific models is an approach that addresses preservice teachers as epistemic agents, that is, they pursue personally-relevant epistemic goals to generate knowledge about a

target phenomenon by engaging in relevant and intentional scientific practices such as constructing, testing, evaluating, and revising models (Berland et al., 2016; Duschl, 2008; Kinnuttila, 2011; Gouvea & Passmore, 2017). This approach is expected to help preservice science teachers design instruction that also empowers their future students as epistemic agents rather than passive learners. Such professional learning experiences involving coding and simulations are worthy of investigation to examine how they affect preservice teachers who have limited understanding of scientific models and modeling (e.g., Krell & Krüger, 2016; Schwarz et al., 2009; Windschitl et al., 2008; Zacharia, 2003) and limited experience teaching with coding (Gal-Ezer & Stephenson, 2010; Google & Gallup, 2015; Maiorana et al., 2017).

Purpose and Research Questions

The purpose of this study was twofold: to implement the redesigned CS-Model and to investigate preservice secondary science teachers' experiences with coding and scientific modeling. Specifically, this study investigated if and how participation in CS-Model affected preservice teachers' (1) epistemological understanding of scientific models and modeling, and (2) conceptual understanding of coding concepts. This study also examined how they use block-based coding to support scientific modeling in lessons. The following research questions guided the study:

RQ1: Do preservice teachers' epistemological understanding of scientific models and modeling change after CS-Model, and how?

RQ2: Do preservice teachers' understanding of computer science concepts change after CS-Model, and how?

RQ3: How do preservice teachers use coding to support scientific modeling in lessons?

Methods

Study Design

This was a predominantly qualitative mixed methods study (Greene, 2007; Hesse-Biber, 2010; Plano Clark & Creswell, 2011; Teddlie & Tashakkori, 2006) that examined preservice teachers' experiences related to the implementation of CS-Model in a teacher education course.

Table 4.1. presents data sources and analysis methods per research question.

Table 4.1

Research Questions, Data Sources, and Analysis Methods

Research Questions	Data Sources	Analysis Methods
RQ1 on epistemological understanding of models and modeling	- Pre- and post-interview transcripts	- Wilcoxon Signed-Rank test - Qualitative framework analysis
RQ2 on understanding of computer science concepts	-Pre- and post-interview transcripts	- Wilcoxon Signed-Rank test - Qualitative framework analysis
RQ3 on use of coding in scientific modeling lessons	-Lessons - Post-interview transcripts	- Qualitative framework analysis - Open coding - Qualitative thematic analysis

Setting and Participants

Participants were preservice teachers recruited from a teacher education course at a public university in southeastern United States. The course focuses on technologies for middle and secondary science teaching. Informed consent was collected from all 19 participants (Table 4.2), and among them there were 11 females and eight males. They were bachelor's (12) or master's (7) students in science education with an emphasis on earth/space science, biology, chemistry, or physics. Participants were White (13), Black (2), Asian (2), or Latino (1). One did not disclose their race. Seven participants had experience with programming and four with block-based programming. Participants were on average 22.68 years old (SD = 2.60). Anne, Erica, and Marcus participated in the CS-Model pilot (Chapter 3). Names are pseudonyms.

Table 4.2

Participants' Information and Data Completion

Participants	Race	Degree	Area of Emphasis	Programmed Before	Used Block-based Coding	Data Completion		
						Pre-interview	Lesson	Post-interview
Jayden	White	Bachelor's	Earth/space science	O	O	O	O	O
William	White	Bachelor's	Biology	X	X	O	O	O
Mary	White	Master's	Biology	O	X	O	O	O
Sam	Asian	Bachelor's	Biology	X	X	O	O	O
Simon	White	Master's	Biology	X	X	O	X	O
Regina	White	Bachelor's	Plant biology	O	O	O	O	O
Carla	White	Bachelor's	Biology and communication studies	X	O	O	O	O
Carl	White	Master's	Chemistry	X	X	O	O	X
Fiona	Black	Bachelor's	Biology and public health	X	X	O	O	O
John	Not disclosed	Bachelor's	Physics	O	O	O	O	O
Rocco	Latino	Bachelor's	Earth/space science	O	X	X	O	X
Calvin	White	Master's	Biology	X	X	X	O	O
Esther	White	Bachelor's	Biology	X	X	X	O	X
Juliet	Black	Bachelor's	Biology	X	X	X	O	X
Chloe	White	Bachelor's	Earth/space science	X	X	X	O	X
Rafaela	White	Bachelor's	Biology	X	X	O	O	X
Anne*	White	Master's	Biology and Chemistry	X	O	O	X	X
Erica*	White	Master's	Biology	X	O	O	O	O
Marcus*	Asian	Master's	Physics	O	O	O	O	X

Notes. *Participated in the CS-Model pilot study. Pseudonyms are the same.

“O” means yes, and “X” means no.

Study Timeline and Procedures

This was a 4-week study that involved in-person and asynchronous online activities (Table 4.3). *In the first week*, the researcher delivered a 10-minute presentation (Appendix A) on block-based coding in Scratch and mentioned that Scratch would be used in a module jointly implemented by the researcher and course instructor. Information on models and modeling was not covered during the presentation to avoid influencing participants' responses during subsequent pre-interviews.

Table 4.3

Overview of Study Timeline and Procedures

Week	Modality	Activities
1	In-person	- Presentation on Scratch coding, recruitment, and demographics survey - Pre-interviews
2	In-person	- Coding workshop: Presentation on scientific models and modeling, demonstration of Scratch coding and mindful debugging, coding activities, and whole-class discussion
	Asynchronous	- Review instructional materials: Interview on coding in schools, exemplary lesson, and code-based science simulation in Scratch - Reflection assignment
3	In-person	- Anchoring problem: design water filter to clear polluted water - Construct models of effective water filters, create and record hypotheses, and code water filter simulations based on hypotheses - Test water filter designs by conducting physical experiments and recording results - Evaluate experiment results against hypotheses - Revise water filter models and simulations - Discuss perceptions of teaching scientific modeling with coding
	Asynchronous	- Write essay describing how block-based coding was used to simulate science concepts involved in water filtration activities - Use CS-ModeL tool to design lesson wherein block-based coding is used to support scientific modeling
4	In-person	- Post-interviews

In that same week, the researcher conducted face-to-face, semi-structured, and artifact-based pre-interviews (Table 4.4) that addressed RQ1 and RQ2. Regarding the first research question on epistemological understanding of models and modeling, nine questions were

borrowed and one question was adapted from Schwarz and White's (2005) protocol. One question was created by the researcher. The interview was structured so the researcher asked an open-ended question, attentively listened to the content in participant's response, and used techniques to elicit reflection (Roulston, 2010) such as asking follow-up questions (e.g., why?), and asking for concrete examples (e.g., can you give me an example of why a scientist would change a model?) (Appendix B). Interview questions that addressed RQ2 were designed by the researcher and administered along with a Scratch animation (Appendix C), which served as an artifact to elicit participants' prior knowledge of the computer science concepts of loops, conditionals, and delays.

Table 4.4

Representative Pre- and Post-interview Questions

Research Question	Interview Questions
RQ1 on epistemological understanding of models and modeling	<ol style="list-style-type: none"> 1. "What is a model?" (Schwarz & White, 2005, p. 182) 2. "What are models for?" (Schwarz & White, 2005, p. 182) 3. Which modeling activities should students be exposed to and what is their value?
RQ2 on understanding of computer science concepts	<ol style="list-style-type: none"> 4. This is a repeat. What do you think this is for? 5. This Scratch simulation is coded so that the cat moves from point A to point C. If you had to make the cat stop at point B, which is halfway between these two points, how would you change the code? 6. If you had to make the cat go slower from A to C, how would you change the code? 7. If you had to make the cat say meow once the dog is clicked, how would you change the code?
RQ3 on use of coding in scientific modeling lessons	<ol style="list-style-type: none"> 8. Tell me about your experience designing the lesson 9. Which science concepts did you target on your lesson? 10. Which coding concepts did you target on your lesson? 11. How do the activities you designed represent your understanding of scientific modeling?

In the second week, the researcher led a 1.5-hour workshop on coding for scientific modeling (Appendix D). The researcher (a) delivered a 10-minute presentation on scientific

models and modeling, (b) proposed using block-based coding to simulate science phenomena and support scientific modeling, (c) showed Scratch features (e.g., searching simulations, remixing code), (d) demonstrated how to assemble blocks to create an animation, (e) showed examples of simulations on topics pertinent to participants' future teaching areas, and (f) demonstrated how to mindfully debug code errors (e.g., create, document, and test hypotheses). This workshop was designed based on results from the pilot implementation of CS-ModeL (Vasconcelos & Kim, under review, 2019), which suggested that preservice teachers should learn to code before use of coding for scientific modeling purposes. The debugging demonstration was proposed by Kim, Yuan, Vasconcelos, Shin, and Hill (2018). Next, participants completed two Scratch coding activities (Appendix E) wherein they used blocks that embody loops, conditionals, and delays. The second coding activity was downloaded from the Scratch Wiki⁵. The researcher and two research assistants with previous experience in Scratch coding offered support during the workshop. At the end, the researcher and course instructor led a whole-class discussion on ways that coding can deepen science learning.

As homework in the second week, participants (a) read an interview on the future of coding in schools (Merrill, 2017), (b) reviewed at least one lesson unit that uses coding to support scientific modeling and targets topics related to their future teaching areas (Project Growing Up Thinking Scientifically (GUTS), n.d.) (Appendix F), (c) analyzed a science simulation available in Scratch, and (d) completed a reflection assignment (Appendix G) that prompted them to express their perceptions of block-based coding simulations and develop ideas for teaching with coding.

⁵<https://static1.squarespace.com/static/54a1ab67e4b092556fa8c9e1/t/579f7d1915d5db2343bdb870/1470070045973/Broadcasting+in+Scratch.pdf>

In the third week, participants engaged in scientific modeling and coding activities involving water filtration experiments during a 3-hour in-person class meeting (Appendix H). Scientific modeling activities were adapted from Kim and Oliver (2018) though their work did not include coding activities. First, the researcher conducted a 5-minute debriefing about their reflection activity (homework) from the previous week. Second, participants were introduced to an (a) anchoring problem that involved lack of clean water at a camping site, (b) problem criteria/constraints that included a list of available materials (plastic bottles, coffee filters, rubber bands, cotton balls, gravel, and activated charcoal powder, grounds, and pellets), and (c) a rule that entailed designing a water filtration system to clear impurities out of water. Third, participants joined a whole-class discussion led by the researcher to define independent and dependent variables, identify variables relevant to the problem, and craft research questions (e.g., how filtration materials affect water color, filtration time, and water input-output ratio?). Fourth, participants documented their hypotheses on water filtration designs in pairs using the hypotheses sheet (Appendix I). The sheet includes designs featuring individual materials and space for self-made designs to combine different materials. Fifth, participants coded water filter simulations in Scratch to externalize and visualize their models and hypotheses, as well as manipulate variables. Simulations has been designed and partially coded by the researcher so that participants only had to complete blank block parameters that represented key variables in the water filtration experiments (Table 4.5). While coding, participants also used a debugging sheet (Appendix J), which contained a list of common coding errors, strategies to mindfully identify and fix coding errors, and reflection prompts to document the debugging process (e.g., error, hypothesis, executed solution, and cause of problem) (Kim et al., 2018). Coding activities took place in a computer lab.

Table 4.5

Blocks, Computer Science and Science Concepts, and Simulation Coding

Coding Blocks	Computer Science Concepts	Science Concepts	Coded Simulation
When I receive	Independent variables	Filtration material(s)	Select filtration material (cotton, gravel, or charcoal)
Broadcast	Dependent variables	Water quality	Select water quality (dirty, brackish, or clear)
Wait_secs	Delay	Water filtration time	Enter number to control delay between animation frames and overall filtration time
Repeat	Loops	Input-output ratio	Enter number to control water sprite going downwards through the filter and how much water leaves or stays in the filter

Sixth, participants observed the researcher conduct experiments using individual filtration materials in a science lab. During this activity, the researcher verbally prompted participants to make predictions, observe systematic manipulation of materials, evaluate results against their hypotheses, and provide evidence-based explanations to discuss science concepts such as pore-size efficiency, absorption, and adsorption. Participants completed the experimental data sheet (Appendix K) to record water filtration time in seconds, and water input-output in milliliters. They also took photos of water color before and after experiments. The experiment data sheet was designed so that participants would not rely only on memory to compare experiment results with original hypotheses (Vasconcelos & Kim, 2019). Seventh, teams conducted three self-designed physical experiments to test their hypotheses about water filtration designs using combined materials. They also recorded results on the experimental data sheet. The researcher, course instructor, and two research assistants provided support when needed. Eighth, teams presented their most effective water filtration design to class and reflected on how results confirmed or refuted original hypotheses. Ninth, participants returned to the computer lab and

revised code for their computer simulations based on their revised models of effective filtration systems. Tenth, the researcher and course instructor led a debriefing session so participants (a) shared their experiences with simulation coding and physical experiments, and (b) discussed foreseen affordances and challenges of teaching scientific modeling with block-based coding.

As homework (Appendix L), participants wrote an essay on how blocks were used to materialize key variables of water filtration systems. Participants were instructed to use terminology learned in class regarding science filtration concepts and computer science concepts. In addition, participants designed a lesson within which they had to use block-based coding to support scientific modeling. Participants individually designed the lesson using the CS-Model tool. Assignment instructions prompted participants to target a topic relevant to their future teaching areas. Participants designed the lesson based on an existing or hypothetical Scratch science simulation.

In the fourth week, participants joined individual, in-person, semi-structured, and artifact-based post-interviews. This interview addressed all research questions (Table 4.4). Participants' lessons and essays were used to promote recall of cognitive processes (De Smet, Van Keer, De Wever, & Valcke, 2010; Lyle, 2003).

Data Analysis

Procedures to warrant data analysis rigor.

Several procedures were adopted to enhance data analysis rigor. First, the researcher adopted computer tools to assist with data analysis. Data was transcribed with an online speech to text transcription and editing software. Transcriptions were reviewed line by line against audio files to check accuracy. Quantitative data was analyzed in SPSS (version 25) and qualitative data in NVivo (version 25). Second, the researcher maintained a journal (Creswell & Miller, 2000;

Tracy, 2010) to record data analysis procedures, rationale for why procedures were suitable to answer research questions, and preliminary interpretation of data (Appendix M). Third, the researcher's major adviser reviewed the researcher's journal and data analysis files in SPSS and NVivo. A fellow graduate student who has extensive experience with statistics reviewed the researcher's SPSS output and helped the researcher reflect on suitability, strengths, and weaknesses of nonparametric statistical tests. Fourth, results are presented with in-depth descriptions and illustrated with participants' words verbatim (Tracy, 2010; Zucker, 2009).

RQ1: epistemological understanding of models and modeling.

To assess participants' understanding of scientific models and modeling, an adaptation of Upmeyer zu Belzen and Krüger's (2010) framework⁶ of model knowledge (Table 4.6) was used. The category "testing models" from the original framework was removed for two reasons: (1) It overlaps with other categories and (2) it was not addressed in the interview protocol.

Quantitizing methods were administered by assigning numerical scores to non-numerical data (Sandelowski, Voils, & Knafl, 2009), which consisted of participants' responses to pre- and post-interviews. The researcher read participants' responses to interview questions several times and assigned scores 1 (level 1), 2 (level 2), or 3 (Level 3) based on their knowledge of (a) nature of models, (b) why multiple models exist, (c) purpose of models, and (d) how models change.

Subsequently, the researcher retrieved coded excerpts in NVivo and used constant comparison methods (Glaser & Strauss, 1967; Strauss & Corbin, 1998) to examine if the scores and level of model knowledge assigned to participants were suitable and accurate. A few corrections were made. Examples of participants' coded responses are presented in Table 4.6. The resulting data set was ordinal, categorical, and involved two related samples (Field, 2013). Although the

⁶ Original framework in German. English version used by Krell, Reinisch, and Krüger (2014).

Table 4.6

Adaptation of Upmeier zu Belzen and Krüger's (2010) Framework of Model Knowledge and Coded Examples

	Level 1		Level 2		Level 3	
	Description	Example	Description	Example	Description	Example
Nature of models	Replication or description of a referent	“A scientific model is something to show how something works” (Fiona).	Simplified representation of a referent	“A model would be a representation of an instance or a system” (John).	Conceptual or theoretical artifact based on a referent	“A model is a concept that you have in your head of how something works or how a phenomenon occurs in science, and it's something that you kind of create out of that to express that idea, that concept” (Regina).
Multiple models	Different formats	“A model of the water cycle can be seen on a piece of paper where it has the arrows and then I'm pretty sure somebody can make a 3D model of the water actually evaporating. So yeah many different models could be used for one phenomenon” (Sam).	Focus on different features of a referent	“You can have a model for how many times it [lightning] is going to strike at a given year. But you could also have a model for how much ozone it is going to displace and that's tomorrow models looking at the same thing. Now those aren't the same models looking at the same thing” (William).	Various hypotheses about a referent	“I think you can construct models differently depending on what you need, your hypotheses, or what you're trying to accomplish with that model” (Sam).

Purpose of models	Describe a referent	“Models are for visualizations of all of your data.” (Mary)	Explain a referent	“Models are to help clarify how something works and teach how something works as well as to be able to explore the mechanics of different things.” (Erica)	Predict about a referent	“I feel like scientists use models in many ways, whether it's to help them predict something that they want to be able to show the public or show the community scientific community or whether it's a, a result of their studies with like graphs and things like that.” (Carla)
Changing models	Correct flaws in the model itself	“If they discovered that one part of it is [model] wrong or it is not representative of what would happen in real life” (Fiona).	Accommodate new ideas or evidence	“If they discovered new data or knew if there was a significant amount of research that supported something that they would need to alter it” (Carla).	Refuted hypotheses about a referent	“If there is something that they [scientists] observe or something they noticed that goes against what their model would have predicted then they have to alter their model to fit that.” (Regina)

normality assumption was not violated, the sample size was small. Thus, the non-parametric Wilcoxon Signed-Rank test was administered to investigate if there were differences in participants' model knowledge before and after CS-Model (Gaddis & Gaddis, 1990; McCrum-Gardner, 2008; Siegel, 1957).

Next, sequential and explanatory qualitative analysis methods were administered to shed light onto quantitative results (Greene, 2007; Plano Clark & Creswell, 2011; Teddlie & Tashakkori, 2006). Qualitative analysis focused only on participants who experienced change (positive or negative) in their assigned scores. Thematic analysis (Boyatzis, 1998; Clarke & Braun, 2013; Fereday & Muir-Cochrane, 2006) was administered to describe changes in participants' epistemological understanding of scientific models and modeling. A sample of qualitative data analysis is presented in Appendix N.

RQ2: understanding of computer science concepts.

The rubric in Table 4.7 was used to assess participants' understanding of coding concepts based on their responses to pre- and post-interviews. During the interview, participants were asked to explain what the blocks *repeat*, *wait_secs*, *broadcast*, and *when I receive* are used for, as well as solve coding challenges using those blocks (Appendix B). Data analysis focused on three criteria including: (a) proper use of coding-related terminology such as naming blocks and concepts (e.g., *wait_secs* – delay), (b) correct explanation of what each block is used for, and (c) use of blocks to solve coding challenges. Quantitizing methods (Sandelowski et al., 2009) were used to score participants' interview responses. Assigned scores for each category were 0 (inexistent), 1 (intermediate), or 2 (proficient). The rubric was designed based on the premise that comprehensive assessment of coding skills includes proper use of terminology, ability to define computer science concepts, and practical application of concepts for problem solving

(Brennan & Resnick, 2012; Grover, Cooper, & Pea, 2014; Grover & Pea, 2013; Qian & Lehman, 2017). Constant comparison strategies (Glaser & Strauss, 1967; Strauss & Corbin, 1998) were used in NVivo to assign scores based on identified excerpts within interview transcripts. The Wilcoxon Signed-Rank test was conducted to examine if there were differences between participants' conceptual understanding of conditionals, delays, and loops before and after participating in CS-ModeL.

Table 4.7

Rubric to Assess Conceptual Understanding of Coding Concepts

Criterion	Inexistent (0)	Intermediate (1)	Proficient (2)
Use of terminology	Participant does not use coding-related terminology (e.g., name blocks and concepts)	Participant uses some coding-related terminology by either mentioning code blocks or concepts but not both	Participant uses coding-related terminology by referring to both blocks and concepts
Definition of block function	Participant does not know what blocks are used for or provides an incorrect explanation	Participant provides a partially correct explanation of what blocks are used for	Participant correctly explains what blocks are used for
Use block to solve coding challenge	Participant cannot use blocks to solve a coding challenge	Participant provides a partially correct solution for a coding challenge using blocks	Participant successfully solves a coding challenge using blocks

Subsequently, qualitative data analysis was administered to compare and describe how participants' understanding of coding concepts changed based on their assigned scores before and after CS-ModeL. Consequently, participants who were assigned the same score before and after CS-ModeL were not included in the qualitative analysis. An inductive approach and open coding techniques (Blair, 2015; Holton, 2010) were used. A sample of qualitative data analysis is presented in Appendix O.

RQ3: use of coding in scientific modeling lessons.

To assess how participants designed block-based coding tasks for their scientific modeling lessons, qualitative framework analysis was administered. This entailed organizing codes created by the researcher “that can be used to manage and organize [*sic*] the data (...) to summarize/reduce the data in a way that can support answering the research questions” (Gale, Heath, Cameron, Rashid, & Redwood, 2013, p. 1). The researcher designed a rubric to assess if lessons plans feature (a) student-driven scientific inquiry, (b) aligned science learning standards and tasks, (c) aligned computer science standards and tasks, (d) well-connected science and coding tasks, and (e) scientific modeling practices based on Schwarz et al.'s (2009) framework.

Subsequently, open coding and thematic analysis were administered to participants’ post-interview transcripts and lesson plans. Data analysis aimed to shed light onto (a) specific ways that coding was used to support scientific modeling and (b) how participants’ epistemological understanding of modeling influenced their lesson design. A sample of qualitative data analysis is presented in Appendix P.

Results

RQ1: Epistemological Understanding of Scientific Models and Modeling

Quantitative results.

Ten participants’ responses to pre- and post-interview questions were analyzed. Participants’ assigned level of model knowledge scores were entered in Table 4.8, and arrows show change in participants’ scores from pre- to post-interviews. Arrows pointing right represent increase in scores, and arrows pointing left represent regression in scores. Absence of an arrow represents same score assigned at both time points. Based on the scoring and the classification scheme used, arrows in Table 4.8 and the number of participants assigned to more advanced

levels (Figure 4.1) show that the majority of participants' scores increased. Statistical tests were administered to quantitative data. The Shapiro-Wilk test, which is suitable for small sample sizes, showed that all variables are normally distributed. The nonparametric Wilcoxon Signed-Rank test was conducted given the small sample size.

Table 4.8

Changes in Participants' Understanding of Models and Modeling

Participant	Nature of models			Multiple models			Purpose of models			Changing models		
	1	2	3	1	2	3	1	2	3	1	2	3
William		● → ●			●			●				●
Sam		●		● → ●		●	● → ●		●	● ← ●		
Regina		●	●	● → ●		●	● → ●		●		● → ●	●
Carla	● → ●			● → ●		●	● → ●		●		● → ●	●
Fiona	● → ●			● ← ●		●		● → ●		●		
John		●		● → ●		●	● → ●		●			●
Erica		● → ●		● → ●		●		● → ●		● → ●		●
Jayden		●		● → ●		●	● → ●		●	● ← ●		●
Mary		● → ●		●		●	● → ●		●	● → ●		●
Simon		● → ●			●			● ← ●		● → ●		●

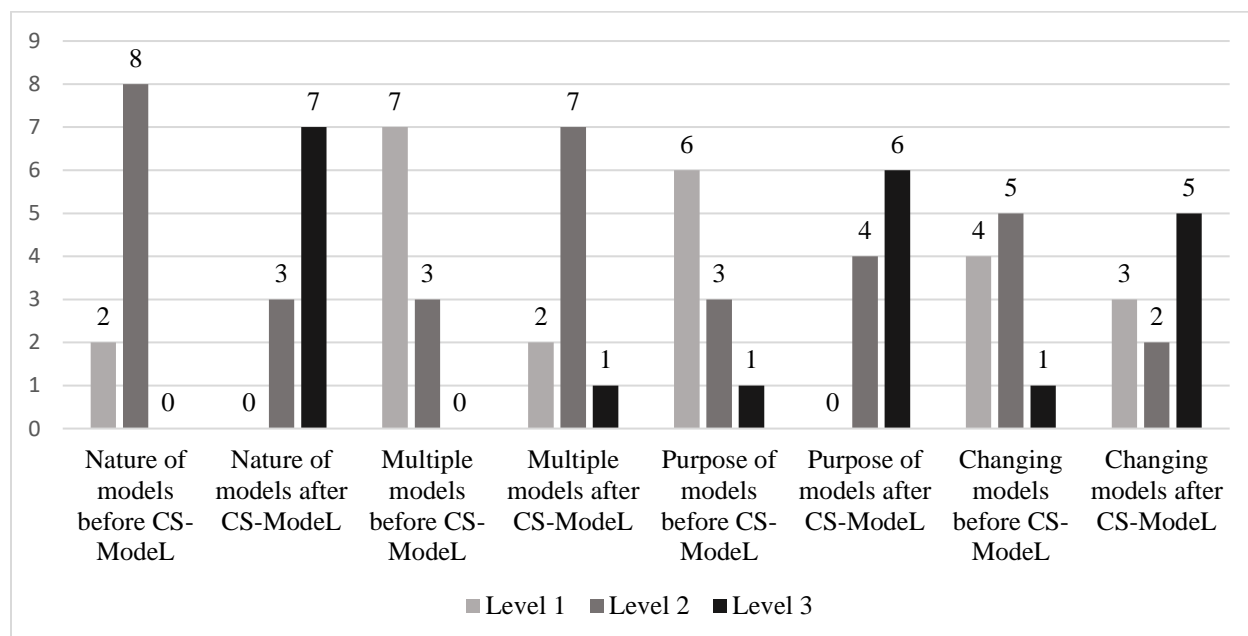


Figure 4.6. Participants before and after CS-ModelL per model knowledge scale.

Statistical results showed that participants' scores after CS-Model were statistically significantly higher than before CS-Model in nature of models ($Z = -2.460, p = 0.014, r = -0.78$) and purpose of models ($Z = -2.373, p = 0.018, r = -0.75$). Differences in participants' scores were not statistically significant for multiple models ($Z = -1.897, p = 0.058, r = -0.60$) and changing models ($Z = -1.406, p = 0.160, r = -0.44$). Quantitative results are presented in Table 4.9. Large effect sizes were found for nature of models, purpose of models, and multiple models. A moderate effect size was found for changing models.

Table 4.9

Results from Wilcoxon Signed-Rank Tests

Variable	Pre-interview		Post-interview		Z	p	r
	M	SD	M	SD			
Nature of models	2.00	.422	3.00	.483	-2.460	.014	-.78
Multiple models	1.00	.483	2.00	.568	-1.897	.058	-.60
Purpose of models	1.00	.707	3.00	.516	-2.373	.018	-.75
Changing models	2.00	.675	2.50	.919	-1.406	.160	-.44

Note. Scores range from 1-3.

Qualitative results.

Theme 1: Nature of models: From visual representations to conceptual tools.

Before CS-Model, all but two participants achieved level 2 (Table 4.8) as they defined models as a simplified and idealized representation of a complex referent. During pre-interviews, Jayden explained that a model is “a representation of an instance or a system. Could just be a simple sketch modeling how something functions, or it could be a physical, 3D model, or maybe even a miniature scale of an actual system.” The two participants in level 1, Carla and Fiona, defined models as ways to describe, duplicate, and “show how something works” (Fiona). All participants referred to models as accurate analogues to a referent. Carla, Sam, and William

relied on mathematics to define models as a “mathematical way to explain something that occurs in the natural world” (William).

After CS-ModeL, all but three participants reached level 3 of model knowledge as they defined models as conceptual or theoretical constructions of a referent. For example, Regina said that a model is an expression of “a concept that you have in your head of how something works or how a phenomenon occurs in science.” Additionally, several participants expressed an understanding of agency and how it influences scientific modeling, such as in Carla’s definition: “A model is something that a student uses to either construct their ideas and show like what they think is going to happen to something.” Similarly, Mary referred to models as artifacts that embody one’s “hypothetical thoughts.” Three participants’ (Sam, John, and Jayden) scores did not change after CS-ModeL. Results displayed in Table 4.8 and Figure 4.1 indicate that many participants achieved the highest level of knowledge about the nature of models.

Theme 2: Multiple models: From different forms to specific foci and goals.

During pre-interviews, all but three participants explained that multiple models for the same phenomenon can be created in different formats such as a computer simulation or a drawing, and as such, participants were categorized as level 1. Sam’s rationale was “a model of the water cycle can be seen on a piece of paper where it has the arrows and (...) a 3D model of the water actually evaporating.” Three participants (William, Fiona, and Simon) mentioned that different models of the same phenomenon focus on specific aspects of a referent, and these participants were categorized as level 2. As Fiona explained, “you could have different variables be changed across the models.” Jayden was not able to explain why multiple models exist because he was not sure.

During post-interviews, only Sam reached level 3, which entails creating different models based on different hypotheses about a referent. In Sam's opinion, "you can construct models differently depending on what you need or what you're trying to accomplish in terms of hypotheses for that model." Sam's response shows that she also considered the agent's epistemic goal as a criterion to inform model creation. Seven participants were categorized as level 2. William's and Simon's score did not change. The other five participants (Regina, Carla, John, Erica, and Jayden) articulated that different models are created based on an agent's needs to focus on key features of the referent and/or epistemic goals. As Regina explained,

if you're trying to show how the moon goes around the earth, (...) you could try to show the phases of the moon. There is a variety of ways to kind of depict that depending on what you're interested in knowing about the phases.

Fiona was the only participant whose score decreased after CS-Model, and she was assigned to level 1. During the pre-interview, Fiona mentioned that models of a phenomenon vary based on different foci or variables of interest. In the post-interview, her response focused solely on how models can be materialized as physical, pictorial, or virtual tools. Fiona also expressed a misconception that different models to "serve the same purpose."

Results displayed in Table 4.8 and Figure 4.1 suggest that several participants improved their understanding of why multiple models of the same phenomenon exist. At first, they emphasized variety in model formats. Later, most participants mentioned that different models feature different elements of a referent and/or fulfill specific goals. All but one participant failed to see models as tools to investigate different hypotheses about a phenomenon.

Theme 3: Purpose of models: From description to prediction.

Before CS-Model, Simon was the only participant assigned to the advanced level 3 of model knowledge regarding the purpose of models. As he described, models are “for observations and making predictions, (...) creating a tangible representation of what you are trying to study and observe.” No other participant referred to the purpose of models as generating predictions about a referent. Three participants (William, Erica, and Fiona) were assigned to level 2 given their perceptions of models for explanatory purposes, such as in “models are to help clarify how something works and teach how something works” (Erica). All other participants were classified as level 1 because they articulated that models are simply for describing or demonstrating a target phenomenon. For instance, Jayden said that models are “demonstrations to describe a system.”

After CS-Model, eight participants experienced an increase in score, and six of them reached level 3. For example, Carla considered models as generative tools that “students use to construct their ideas and show what they think is going to happen to something.” Similarly, Erica asserted that models are to “gain more understanding about how that system or concept works.” Although not always explicit, it was possible to infer that most participants addressed models as tools that embody predictions about a referent as evidenced in Carla’s response above. William and Simon adopted a systems thinking perspective to define models as complex entities formed by interrelated smaller components. As William stated, a model is “a way to systematically break down pieces of a whole to describe it (...) and breaking it down into pieces in order to see how each individual part works in order to understand the whole.” William’s score did not change. Simon experienced a regression in his score after CS-Model. He explained that a model is “a system that we can use to demonstrate a phenomenon”, and this response represents level 2.

Results suggest that most participants' scores increased, and six of them reached the advanced level of model knowledge. Data presented in Table 4.8 and Figure 4.1 show that many participants improved their understanding of the purpose of models. During pre-interviews, participants' responses mostly addressed models as tools for describing and explaining a referent. During post-interviews, several of them emphasized models as generative tools that materialize predictions about a referent.

Theme 4: Changing models based on new evidence.

During pre-interviews, William was the only participant who asserted that scientists change models if empirical data is found to contradict hypotheses or predictions, and as such he was assigned to level 3 of model knowledge. William said during the pre-interview:

Let's talk about climate change. Forty years ago, they had a certain model about how much carbon parts per million was going to be in the atmosphere. (...) We did not predict the population was going to be growing this fast. (...) So we are going to have to adapt that model not to fit these last 40 years that we did not think it was going to grow that quickly. So sometimes you can't predict right and you have to adapt [the model].

Four participants (Fiona, Erica, Mary, and Simon) were assigned to level 1 as they pointed out that models evolve if flaws are found in a model. Fiona said that one needs to change a model if "one part of it is wrong or it is not representative of what would happen in real life." The remaining five participants were categorized as level 2 as they explained that models need to be revised due to unexpected evidence. For example, these participants said that models change if you find something that "you did not properly account" (Jayden) or if there is "evidence that their current understanding is flawed in some way" (Regina).

During post-interviews, five participants were categorized as level 3. Carla explained that a scientist would need to change a model if “hypothesis is incorrect (...) or if you would want to elaborate and design a model with the new hypothesis that builds on your last one.” In this case, the participant considered continuity in the scientific enterprise as a scientist builds upon existing models, tests hypotheses, and refines models. Two participants (John and Mary) were assigned to level 2 as they considered that models change “every time you learn new data” (Mary). Three participants (Fiona, Sam, and Jayden) were categorized as level 1, as they believed that models change due to “an error in a previous model” (Sam). Three participants’ (William, Fiona, and John) scores did not change. Jayden and Sam experienced a regression in their score.

An important finding from data analysis was that Simon, Sam, and Mary partially attributed changes in models to the evolving nature of science and scientific inquiry in post-interviews. Simon described science as “a process”, Sam as “tentative”, and Mary said that it “changes a lot.” This shows their understanding of the scientific enterprise as an iterative process and of models as imperfect constructs.

Results presented in Table 4.8 and Figure 4.1 suggest that half of participants’ scores increased and only four participants reached an advanced epistemological understanding about why models change. Before CS-Model, these participants thought that models change due to flaws in the model itself or unaccounted factors. After CS-Model, participants articulated that models change due to refuted hypotheses and to the evolving nature of science and scientific inquiry. Results need to be cautiously considered given that other participants either did not experience change in scores or experienced a regression in scores.

RQ2: Conceptual Understanding of Conditionals, Delays, and Loops

Quantitative results.

A total of 10 participants attended both pre- and post-interviews. Three participants (Jayden, Mary, and Simon) could not complete coding challenges in Scratch due to a university-wide loss of wireless Internet during their post-interviews. These participants' use of terminology and definitions of coding concepts were assessed, but not their application of computer science concepts to solve a challenge. They were not included in statistical analyses involving application of code. Changes in participants' understanding of conditionals, delays, and loops are presented in Tables 4.10, 4.11, and 4.12 respectively. Arrows pointing right show score increase, and arrows pointing left show score decrease. Absence of arrows represent no change in participants' scores. Based on the scoring and classification scheme, arrows show that most participants' scores increased.

Table 4.10

Changes in Participants' Understanding of Conditionals

Participants	Conditionals								
	Terminology			Definition			Application		
	0	1	2	0	1	2	0	1	2
William ⁺	●	→	●	●	→	●	●	→	●
Sam ⁺	●	→	●	●	→	●	●	→	●
Regina			●			●	●	→	●
Carla			●		●	→	●	→	●
Fiona ⁺	●	→	●	●	→	●	●		
John ⁺			●		●	→	●	→	●
Erica [*]		●	→	●		●	→	●	→
Jayden			●		●	→	●		N/A
Mary	●	→	●		●				N/A
Simon		●	→	●	→	●			N/A

Table 4.11

Changes in Participants' Understanding of Delays

Participants	Delays								
	Terminology			Definition			Application		
	0	1	2	0	1	2	0	1	2
William ⁺		● → ●			● → ●				●
Sam ⁺		● → ●		● → ●					●
Regina		●			● → ●				●
Carla			●			●		● → ●	
Fiona ⁺	● → ●			● → ●			● → ●		●
John ⁺		● → ●			● → ●				●
Erica [*]			●		● ← ●			● → ●	●
Jayden			●			●		N/A	
Mary		● → ●			● ← ●			N/A	
Simon		● → ●			● → ●			N/A	

Table 4.12

Changes in Participants' Understanding of Loops

Participants	Loops								
	Terminology			Definition			Application		
	0	1	2	0	1	2	0	1	2
William ⁺	● → ●			● → ●			● → ●		●
Sam ⁺		●			● → ●		● → ●		●
Regina			●		● → ●		● → ●		●
Carla			●			●	● → ●		●
Fiona ⁺	● → ●				● → ●		● → ●		●
John ⁺			●			●			●
Erica [*]	● → ●				● → ●		● → ●		●
Jayden			●		● → ●			N/A	
Mary			●			●		N/A	
Simon			●			●		N/A	

Statistical tests were conducted. The Shapiro-Wilk test showed that all variables were normally distributed, but the nonparametric Wilcoxon Signed-Rank test was suitable given the small sample size. Results indicated that differences in participants' understanding of computer science concepts before and after CS-Model were statistically significant regarding use of conditionals terminology ($Z = -2.271$, $p = .023$, $r = -0.86$), definition of conditionals ($Z = -2.585$,

$p = .010$, $r = -0.98$), application of conditionals ($Z = -2.724$, $p = .006$, $r = -0.88$), use of delay terminology ($Z = -2.449$, $p = .014$, $r = -0.93$), definition of loops ($Z = -2.333$, $p = .020$, $r = -0.88$), and application of loops ($Z = -2.810$, $p = .005$, $r = -0.93$). No statistically significant differences were found in definition of delays ($Z = -1.508$, $p = .132$, $r = -0.57$), application of delays ($Z = -1.633$, $p = .102$, $r = -0.62$), and use of loop terminology ($Z = -1.633$, $p = .102$, $r = -0.62$). Statistical results are presented in Table 4.13.

Table 4.13

Quantitative Results on Participants' Understanding of Computer Science Concepts

Variable	Subscale	Pre-interview		Post-interview		Z	p	r
		M	SD	M	SD			
Conditionals	Terminology	1.00	.943	2.00	.422	-2.271	.023	-.86
	Definition	1.00	.765	2.00	.316	-2.585	.010	-.98
	Application	.00	.488	2.00	.378	-2.333	.020	-.88
Delays	Terminology	1.00	.632	2.00	.422	-2.449	.014	-.93
	Definition	1.00	.789	2.00	.483	-1.508	.132	-.57
	Application	2.00	.787	2.00	.000	-1.633	.102	-.62
Loops	Terminology	2.00	.843	2.00	.316	-1.633	.102	-.62
	Definition	1.00	.675	2.00	.000	-2.333	.020	-.88
	Application	1.00	.378	2.00	.000	-2.449	.014	-.93

Qualitative results.

Analysis of participants' responses to pre- and post-interviews revealed that participants held several misconceptions (see Table 4.14) about the computer science concepts conditionals, delays, and loops. An overview of which participants corrected misconceptions after CS-Model is presented in Table 4.15. Qualitative results are discussed per concept as follows.

Conditionals – input (when I receive block).

One misconception was identified during pre-interviews across two participants (William and Mary), who defined the input block (when I receive) as a message trigger. For instance, they said that the block entails “giving a command to the figure” (William) and “the [code] sequence

is going to happen after that one action” (Mary). Both participants defined when I receive as an output rather than a block that receives an input. Sam and Fiona were not sure how to define the block. Two participants (Sam and Fiona) were not sure how to explain the use of conditionals (when I receive block). The other six participants properly articulated a definition such as “when you receive a command of some sort, then you will perform a certain action” (Carla).

During post-interviews, no participant expressed misconceptions about conditionals (input). Participants explained that the block when I receive creates input-output relationships between sprites and controls certain actions in a simulation. For instance, Sam said “it is like the input (...), you’re saying what the input is for you to output whatever you’re going to put under that [block].” Similarly, Simon said that “it is a communication between two sprites. When something happens to one sprite, it activates the broadcast code which is programmed with a certain thing.” Results suggest that most participants improved scores and reached the advanced level regarding their understanding of this computer science concept, as presented in Table 4.10. Additionally, no participant expressed misconceptions after CS-Model as shown in Table 4.15.

Conditionals – output (broadcast block).

Only William expressed a misconception about the output block of conditional statements in pre-interviews. He explained that the broadcast block executes the simulation by “portraying it [simulation] on the screen.” Five participants (Sam, Carla, Fiona, Erica, and Simon) were not sure and could not provide an answer. Only four participants (Mary, Regina, John, and Jayden) successfully defined what conditionals are for in pre-interviews by saying that broadcast embodies a “specific kind of output” (Regina).

Analysis of post-interviews suggest that participants did not have misconceptions and all of them were able to provide an explanation about this concept (Table 4.15). Table 4.10 also

shows that most participants' scores increased, which indicates an enhanced understanding of outputs for conditional statements. For instance, William said "it is signaling. You can think of it like a conversation, like one character is going to do an action or say something based on another character broadcasting that."

Delays (wait seconds block).

Analysis of participants' responses to pre-interview questions revealed three misconceptions across six participants. John and Simon believed that delays lead to a pause between two sequential blocks. Sam said it yields a pause between repeats (loops). William, Regina, and Fiona expressed that the block pauses the entire simulation. For example, Fiona explained that it "makes it [simulation] hesitate." Although it is possible to use the block to perform all these functions, those definitions are incomplete and therefore inaccurately describe what the block is used for. Four participants (Carla, Erica, Jayden, and Mary) successfully described the purpose of using delays during pre-interviews.

Results of analysis with post-interviews suggest that four participants (Fiona, John, Mary, and Simon) still held lingering misconceptions about delays as they believed that delays consist of a pause between loops. Fiona and Simon similarly said that a delay "is the hesitation before you go to the next repeat" (Fiona) or "the amount of time in seconds between repetitions as coded by the repeat block" (Simon). The other six participants were able to successfully define the concept as shown in Table 4.15. Participants' scores (Table 4.11) indicate improvement among most of them after CS-Model. This suggests that the majority of participants improved their understanding of delays as a command that causes the code to "wait for however many seconds" (Carla), and it "tells a sprite to pause before going to the next command" (Regina).

Table 4.14

Identified Misconceptions per Computer Science Concept

Blocks	Identified Misconceptions
Replay	Infinite loop
	Replays entire simulation
Wait seconds	Pause between two sequential blocks
	Pause between repeats
	Pause entire simulation
When I receive	Message trigger
Broadcast	Message receiver
	Play simulation on the screen

Table 4.15

Participants' Conceptual Understanding in Pre- and Post-interviews

Participant	Pre-interviews				Post-interviews			
	Loops	Delay	Conditionals (input)	Conditional (output)	Loops	Delay	Conditionals (input)	Conditionals (output)
William	M	M	M	M	✓	✓	✓	✓
Sam	M	M	Not sure	Not sure	✓	✓	✓	✓
Regina	M	M	✓	✓	✓	✓	✓	✓
Carla	✓	✓	✓	Not sure	✓	✓	✓	✓
Fiona	M	M	Not sure	Not sure	✓	M	✓	✓
John	✓	M	✓	✓	✓	M	✓	✓
Erica	✓	✓	✓	Not sure	✓	✓	✓	✓
Jayden	✓	✓	✓	✓	✓	✓	✓	✓
Mary	✓	✓	M	M	M	M	✓	✓
Simon	M	M	M	Not sure	✓	M	✓	✓

Note. M stands for misconception, i.e., identified gap in participants' definitions. "✓" means that participants accurately explained the concept. "Not sure" means participants did not know how to explain the concept.

Loops (replay block).

Two misconceptions about loops were identified across four participants during pre-interviews. Regina defined this computer science concept as an infinite loop that will “play around and around” while others (William, Sam, and Fiona) assumed that the block replays the entire code sequence by “playing the animation again” (William). Five participants (Carla, John, Erica, Jayden, and Mary) correctly provided an explanation of loops as well as their purpose during pre-interviews. Jayden proposed that it consists of “repeating a series of code (...) for x amount of times”, and Erica emphasized that “it repeats whatever series of things you put inside there (...) so it just loops it.

Participants who had misconceptions about loops during pre-interviews did not express the same misconceptions during post-interviews. Mary was the only participant who had a misconception during post-interviews, as she explained that loops “repeat the whole [code] segment all over again.” Table 4.12 shows that most participants improved scores and achieved the advanced level of understanding about loops. As Sam explained, “it repeats whatever you put in the little hamburger” referring to the blocks that are snapped together inside the repeat block. Similarly, Carla added that “whatever is inside those two bars, it would repeat those however many times before moving on if there was more [code] underneath it in the sequence.”

RQ3: Use of Coding in Scientific Modeling Lessons

A total of 18 participants designed a lesson using coding activities to support scientific modeling. Only Simon and Anne did not complete this assignment. Table 4.16 outlines results.

Results of rubric analysis.

Student-driven scientific inquiry. Results suggest that most participants’ lessons involved authentic student-led inquiry, which is a critical component of scientific modeling

instruction. In Sam's lesson, students would investigate how various factors, such as brightness and light color, affect photosynthesis. Students would use block-based coding to "carry out an experiment of his or her choice" (Sam), test variables, collect and analyze data from different outputs such as graphs and tables, and refine their models of photosynthesis by reaching a consensus model with classmates. On the other hand, five participants designed lessons that entailed teacher-led inquiry. In Mary's lesson, students would use block-based coding to simply modify an existing simulation on organisms within an ecosystem. Modifying a simulation as a decontextualized activity does not represent authentic scientific modeling.

Alignment between NGSS standards and scientific modeling activities. All but two participants designed lessons whose scientific modeling tasks were well-aligned with the selected NGSS standards. For example, Carla's learning standards entailed developing and using models to describe asexual reproduction. Her students would create and test their own hypotheses of how "genetic information passes to offspring during reproduction" using block-based coding. Two participants' lessons did not feature such alignment. For example, Rocco's learning standard involved developing and using a model of the earth-sun-moon system though activities in his lesson described students changing angles of the earth in a simulation to explain the four seasons.

Alignment between computer science learning standards and coding activities.

Thirteen participants designed lessons whose coding activities were well aligned with the proposed computer science standards. For instance, Regina's standards involved creating, testing, and refining computational artifacts. In her lesson, students would need to form and express hypotheses about "how food intake and exercise will affect glucose levels" using block-based coding simulations. Additionally, students would need to remix and debug simulations that

contain errors to show how a variety of factors influence homeostasis. Four participants were not successful in aligning computer science standards with coding activities. For instance, Jayden's learning standard focused on promoting collaboration around computing technologies though his lesson did not include activities wherein students collaborate or cooperate using computational artifacts or concepts.

Connection between computer science and science tasks. Twelve participants designed lessons wherein science and computer science tasks were well connected. For instance, Regina's lesson aimed to engage students in using feedback loops as a programming concept to materialize the feedback mechanisms involved in cell homeostasis. Loops represented how factors such as food intake and exercise affect homeostasis over time. On the other hand, five participants' lessons provided disconnected scientific modeling and coding tasks. For instance, Fiona's students were to create models of mitosis and meiosis in Scratch, but the lesson neither listed computer science concepts nor stated how students would use block-based coding.

Anchor phenomenon. Thirteen participants successfully designed activities wherein students would anchor a science phenomenon to a problem that needs to be solved through investigation, inquiry, and model development. For instance, Chloe's multi-day lesson on pollution started by offering students driving questions such as "How does carbon cycle through the environment?" and proposing block-based coding as a tool to visualize and investigate that phenomenon. Students would then develop a quantitative model on how much time it takes for carbon molecules to complete the cycle. Four participants failed to anchor target phenomena. For example, Jayden's lesson provided a rationale for why coding can help students investigate "geologic processes on a viable time scale" but it did not create connections to real-world problems or events.

Construct model. All participants designed tasks wherein students use block-based coding to construct models of target science phenomena. In Erica’s lesson, students would code a simulation to externalize and construct a model of “interactions between parts of a cell that produces the molecules necessary for cell function.” Similarly, in Rafaela’s lesson, students would “use coding to create a simulation where they include their hypothesis in the coding” about cell functions.

Test model. Eleven participants designed tasks wherein students need to test their hypotheses and models of a phenomenon through physical or virtual experiments. For example, Juliet’s students would test their hypotheses of how genetic disorders are passed to a minimum of eight generations and compare their results with peers’. Six participants did not include model testing in their scientific modeling tasks. William’s plan was to engage students in “writing down on paper a broad model for natural selection” rather than conducting empirical or virtual tests.

Evaluate model. Fourteen participants designed lessons wherein students compare multiple models, analyze empirical results, and evaluate the models. For instance, John’s students would create simulations of bounce tests to model momentum. They would review peers’ models created with block-based coding, analyze results from physical tests with different types of balls and surfaces, and assess how model accuracy can be improved to represent concepts such as “mass, energy, bounciness, and velocity.” However, three participants failed to address this model evaluation. In Mary’s lesson, students were asked to “research materials explaining organism interactions” within an ecosystem. Conducting research is not equivalent to evaluating a model against empirical data and theories.

Table 4.16

Analysis of Lessons

Participant	Student-driven Scientific Inquiry	Alignment Between Standards and Tasks		Connection Between Computer Science and Science Tasks	Scientific Modeling Tasks							
		NGSS	Computer Science		Anchor Phenomenon	Construct Model	Test Model	Evaluate Model	Revise Model	Use	Assess	
										Model to Predict/ Explain	Coding Model	
William	X	O	O	O	O	O	X	O	X	O	X	O
Sam	O	O	O	O	O	O	O	O	O	O	O	O
Regina	O	O	O	O	O	O	O	O	O	O	X	O
Carla	O	O	O	O	O	O	O	O	O	O	O	O
Fiona	X	X	O	X	X	O	X	O	O	X	O	O
John	O	O	O	X	O	O	O	O	O	O	O	O
Erica*	O	O	O	O	O	O	O	O	O	O	O	O
Jayden	X	O	X	X	X	O	X	X	O	O	X	O
Mary	X	O	X	O	O	O	X	X	O	X	X	O
Esther	O	O	O	O	O	O	O	O	O	O	X	O
Carl	O	O	X	O	X	O	X	O	O	O	O	O
Chloe	O	O	O	X	O	O	O	O	X	O	O	O
Calvin	O	O	O	O	O	O	O	O	O	O	O	O
Juliet	O	O	O	O	O	O	O	O	O	X	O	O
Marcus*	O	O	O	O	O	O	O	O	O	O	O	O
Rafaela	O	O	O	O	O	O	O	O	O	O	O	O
Rocco	X	X	X	X	X	O	X	X	X	O	X	O

Note. * Preservice teachers who participated in the CS-ModeL pilot.

“O” means participant successfully addressed item in lesson.

“X” means participant did not address item in lesson.

Revise model. Fourteen participants designed tasks wherein students revised their models to improve results, accuracy, or efficiency. In Calvin’s lesson, students would work in groups of four to analyze each other’s models created with block-based coding, identify problems and areas for improvement, and revise their models of how DNA structures determine the structure of proteins. Three participants, however, did not include model revision in their scientific modeling tasks. For instance, Chloe misunderstood the concept of debugging for model revision and described it as if students could revise the model by identifying and fixing coding errors.

Use model to predict or explain. Fourteen participants designed tasks that entailed student-led use of models to explain and/or predict a phenomenon. For instance, Marcus’s students were to use their models of Newton’s Third Law to explain collisions during a soccer game. In other words, they would need to code collisions so that there is an equal and opposite reaction for every action. Three participants did not address this scientific modeling practice. One of them was Juliet, whose description involved students playing “each other’s games to test their models and their own knowledge” of pedigree disorder.

Assess development of coding skills. Eleven participants proposed assessment strategies that focused on student’s development of coding skills, such as “writing a short essay reflecting on their coding skills” (Rafaela), and a coding scenario with embedded errors that students “will then attempt to fix” (Marcus). Six participants did not propose assessments on coding as they focused solely on science concepts.

Assess scientific model development. All participants proposed assessments for scientific model development. For instance, William planned to show students “different models about natural selection and have students write up conclusions.” In another example, Carl

proposed using discussions and one-on-one teacher-student interactions as formative assessments on students' scientific models, as well as using their final simulation in which they "code a chemical reaction based on a given set of reactants" as a summative assessment.

A total of six participants (Sam, Carla, Erica, Calvin, Marcus, and Rafaela) addressed all items in the rubric (Table 4.16). Five participants (Regina, John, Esther, Chloe, and Juliet) failed to address only one or two of the items in the rubric.

Results of thematic analysis.

Theme 1: Coding used as a research tool vs. as an exploration tool.

Eleven participants (Table 4.17) used coding as a research tool to support scientific modeling by designing activities that entailed hypothesis testing, expressing ideas, confirming or rejecting predictions, and explaining phenomena. As Carl wrote in his lesson,

Using a given set of chemicals, students will first hypothesize what are reactants and what are products in the reactions that make cars and rockets go [move]. (...) Students will use coding to visually play out their hypotheses by selecting a combination of reactants to produce a product.

Use of coding as a research tool empowers students as epistemic agents by allowing them to actively generate knowledge and pursue epistemic goals, externalize prior knowledge and possibly misconceptions, collaborate with peers on virtual and/or physical experiments, analyze and record data, and reach a refined model. In this approach, block-based coding also serves as a medium to help students identify gaps in their own knowledge. As Sam explained, "mistakes are encouraged as students can learn from them. At the end [of lesson], students will be given space to reflect on his/her experiment and if (...) it was successful to support his/her hypothesis."

Table 4.17

Use of Coding to Support Scientific Modeling

Participant	Theme 1		Theme 2				Concepts
	Research Tool	Exploration Tool	Abstraction and Modularizing	Being Iterative	Reusing and Remixing	Testing and Rebugging	
William	X	O	O	O	O	X	O
Sam	O	X	O	X	O	O	X
Regina	O	X	X	O	O	X	O
Carla	O	X	O	O	X	X	X
Fiona	X	O	X	X	X	O	X
John	O	X	X	X	X	X	X
Erica	X	O	O	O	O	X	X
Jayden	X	O	O	O	X	X	X
Mary	O	X	X	O	X	O	X
Esther	X	O	O	X	X	X	X
Carl	O	X	X	O	X	X	X
Chloe	O	X	X	O	X	O	X
Calvin	O	X	X	O	X	X	O
Juliet	O	X	X	O	O	O	X
Marcus	O	X	X	O	O	O	X
Rafaela	O	X	O	O	X	O	X
Rocco	X	O	O	O	X	X	X

Note. “O” means participants addressed item in lesson.

“X” means participants did not address item in lesson.

Results suggest that the other six participants integrated coding into their lessons to help students explore, visualize, and better understand a given phenomenon. Lessons entailed teacher-led activities wherein students would code a simulation to demonstrate a process. For example, Rocco’s students were to use an existing simulation to explore the angles of the Earth compared to the sun and how such angles affect the four seasons. Similarly, William’s lesson shows that his students would “get comfortable with a simulation” by running it several times and then they would “write down on paper” their models of natural selection. Such examples feature use of coding as an exploration tool so students can manipulate and explore simulations. Students are not addressed as epistemic agents because lessons neither involve self-driven inquiry nor

engagement in scientific modeling practices to generate knowledge about the target phenomenon.

Theme 2: Most lessons feature computer science practices, not concepts.

Participants were instructed to address computer science learning standards and create related learning objectives. Analysis of lessons showed that most participants addressed computer science practices rather than coding concepts (Table 4.17). Concepts are specific constructs to be learned (e.g., conditionals, loops), and practices describe how learning occurs (e.g., abstraction, debugging) (Brennan & Resnick, 2012). Only three participants explicitly listed concepts and described what students would need to do with code blocks that embodied those concepts. For example, William mentioned that students would use the “clone button to show different generations of offspring” and analyze natural selection for bunnies. Additionally, conditionals would be needed in that lesson to create cause-and-effect relationships in order to show how the presence of wolves affects a population of bunnies.

All participants' lessons addressed at least one of Brennan and Resnick's (2012) practices, which are (a) abstracting and modularizing, (b) being iterative, (c) reusing and remixing, and (d) testing and debugging. Six participants designed tasks wherein coding facilitated abstraction and modularizing, such as in “explain how and why some beaches are sandy and some rocky (...) and run simulations to defend your conclusions” (Jayden). In this case, students were to code and model the effect of several factors on mineral erosion and apply knowledge to generate explanations about coastal environments using block-based coding.

Thirteen participants designed tasks wherein students need to iteratively refine and improve code-based simulations. For instance, William explained that “students edit and change their model to fit their own creative thinking ideas to improve the simulation. By editing it

themselves, they understand more about how the working parts come together to make a coherent hole.” In this case, students iteratively refined the simulation code and output, which is expected to accurately represent the target science phenomenon.

Six participants integrated reusing and remixing into their lessons so their students can explore, test, remix, and reuse peers’ simulations. Such practice is also a strategy to promote exposure to multiple models, prompt model evaluation and revision, and reach a consensus model. For instance, Juliet’s lesson shows that students would use code to design games on pedigree and genetic disorders. Students would “play their peers’ game, (...) share feedback (...), and then reflect on their own pedigree and peers’ pedigree” model.

Seven participants addressed testing and debugging in their lessons. For example, Marcus described it as “when students run into bugs, they will explain the bug and how it is affecting the simulation, and then attempt to fix the error.” Interestingly, Rafaela was the only participant who planned to purposefully include an error in the simulation so that students could correct such an error.

Theme 3: Inconsistencies between participants’ epistemologies and lesson design.

A comparison of RQ1 and RQ3 results was administered to verify if participants’ epistemological understanding was reflected in their design of code-enhanced scientific modeling lessons. This analysis was administered to participants who provided a complete data set. Results suggest that there were inconsistencies between participants’ epistemological understanding of models and modeling and how they designed code-enhanced scientific modeling lessons (Table 4.18). For instance, participants who held an advanced epistemological understanding about the purposes of models (level 3) would be expected to use block-based coding simulations as research tools to help participants test hypotheses and models. Along these

lines, one would think that participants who considered models as explanatory tools (level 2) would use block-based coding simulations as tools to simply explore and visualize a phenomenon. However, data presented in Table 4.18 shows that only six participants followed that pattern. Sam, Regina, Carla, and Mary (level 3 in purpose of models) used block-based coding simulations for research purposes in their lessons, and Jayden and William (level 2 in purpose of models) designed lessons wherein students use coding to explore science concepts.

Table 4.18

Participants' Epistemological Understanding and Lesson Design

Participant	Epistemology	Lesson		Epistemology	Lesson	Epistemology	Lesson
	Purpose of Models	Research	Explore	Multiple Models	See Peers' Models	Changing Models	Peer Feedback
William	2	X	O	2	O	3	X
Sam	3	O	X	3	O	1	O
Regina	3	O	X	2	X	3	X
Carla	3	O	X	2	O	3	O
Fiona	3	X	O	1	O	1	X
John	2	O	X	2	X	2	X
Erica	3	X	O	2	X	3	O
Jayden	2	X	O	2	X	1	X
Mary	3	O	X	1	O	2	O

Note. Scores in bold show that participants designed a lesson aligned with their epistemological understanding. “O” means yes and “X” means no.

Inconsistencies were even more apparent regarding participants' epistemological understanding of the existence of multiple models. It would be expected that those who considered that different models exist due to different foci (level 2) would design activities wherein students are prompted to examine multiple perspectives in models. It would also be expected that participants who believe that different models are used to investigate different hypotheses about a phenomenon (level 3) would create activities so students redesign models to investigate various hypotheses. However, only William and Carla (level 2) followed that pattern (Table 4.18). Sam, Fiona, and Mary exposed students to multiple models though their designed

activities were not representative of their epistemological understanding. The activities proposed by these five participants entailed reviewing peers' models to evaluate a different perspective and provide feedback. The remaining four participants (Regina, John, Erica, and Jayden) were categorized as level 2 in their epistemological understanding of multiple models. However, they did not include activities that involved multiple models.

Inconsistencies were also found regarding participants' design of model revision activities. It was expected that participants at level 3 of model change would design activities wherein falsified hypotheses or predictions drive model revision. Likewise, participants who were in level 2 would be expected to design activities wherein students design models based on new evidence or unaccounted information. It was noticeable that only four lessons (those by Sam, Carla, Erica, and Mary) addressed model revision, and all of them focused on model revision based on peer feedback. This practice illustrates level 2 of model knowledge, which considers model revision as a practice that is driven by new insights rather than falsified hypotheses. Only Mary's lesson followed the pattern described above. Other participants (William, Regina, Fiona, John, and Jayden) did not feature model revision activities in their lessons. Results displayed in Table 4.18 show that many participants did not design code-enhanced scientific modeling lessons that reflected their epistemological understanding of models and modeling.

Discussion

Epistemological Understanding of Models and Modeling

Prior to CS-ModeL, most participants perceived models as visual depictions or idealized representations (levels 1 and 2 respectively) that serve as learning tools and fulfill purposes of describing and explaining a phenomenon (levels 1 and 2 respectively). Previous studies on the

nature and purpose of models conducted with biology teachers (Krell & Krüger, 2016; Reinisch & Krüger, 2018), preservice science teachers (Akerson et al., 2009; Krell, Upmeier zu Belzen, & Krüger, 2012), and in-service science teachers (Van Driel & Verloop, 1999) found similar results. Several participants in this study highlighted the use of models as educational artifacts that supplement science learning experiences, which is in line with Schwarz and Gwekwerere's (2007) results on preservice science teachers' perceptions of models as tools to assess students' conceptual understanding.

Quantitative analysis showed statistically significant differences and a large effect size on participants' epistemological understanding about the nature and purpose of models. Qualitative analysis showed that many participants experienced increased scores as they provided more refined explanations about the nature and purpose of models after participating in CS-Model. Specifically, many participants defined models as conceptual constructs that embody select aspects of a referent and that serve purposes of prediction and hypothesis testing. These results echo claims in the literature that engaging preservice teachers in authentic scientific modeling experiences is beneficial to help them develop a more sophisticated understanding of scientific models and modeling (e.g., Wilkerson, Andrews, Shaban, Laina, & Gravel, 2016; Windschitl, Thompson, & Braaten, 2008b) and scientific reasoning (Stammen, Malone, & Irving, 2018). In CS-Model, participants were addressed as epistemic agents (Berland et al., 2016; Schwarz, Meyer, & Sharma, 2007) as they coded simulations and conducted experiments to fulfill an epistemic goal. Thus, simulation coding is used as an authentic epistemic and epistemological process (Renken, Peffer, Otrell-Cass, Girault, & Chiocciariello, 2016).

Differences in participants' scores regarding their epistemological understanding of why multiple models exist and why models change were not statistically significant. Results of

qualitative analysis suggest that many participants articulated more refined explanations for those two variables after CS-Model, which indicates improvement in their epistemological understanding. During pre-interviews, participants believed that multiple models of a phenomenon exist because a model can take on different forms, such as a simulation or a drawing (level 1). During post-interviews, several participants expressed that multiple models are created based on one's needs and goals. It is important to highlight that only one participant reached the advanced level 3 of model knowledge, and all others failed to account for epistemic purposes and hypotheses as motifs for creating multiple models. Similarly, Krell and Krüger (2016) found that biology teachers lacked a sophisticated epistemological understanding of alternative/competing models. In the present study, CS-Model engaged participants in designing and testing multiple water filtration systems. However, participants were not offered opportunities for scaffolded reflection (e.g., whole-class discussion) on why multiple models are created and which purposes different models serve. Perhaps manipulating multiple models without opportunities for reflection about the underlying reasons why it occurs is not enough to make an impact on preservice teachers' epistemological understanding. In fact, scaffolded reflection is considered beneficial to facilitate understanding of scientific modeling (Grosslight et al, 1991; Windschitl et al, 2008b), and it can be included in future research using CS-Model.

Qualitative analysis of participants' responses to pre-interview questions about why models change showed that participants started off the study with a perception that revisions occur if one finds flaws in the model or encounters unexpected evidence (level 1 and level 2 respectively). During post-interviews, several participants expressed that hypotheses or predictions that are falsified by empirical data is what drives model change. Interestingly, a few participants mentioned that models change due to the tentative nature of science. This replicates

findings in studies conducted with preservice teachers, which examined how participation in authentic scientific inquiry affects preservice teachers' conceptions of the nature of science and scientific inquiry (e.g., Lederman, Schwartz, Abd-El-Khalick, & Bell, 2001; Schwartz, Lederman, & Crawford, 2004). Similar findings are also highlighted in Reinisch and Krüger's (2018) study. Participants attending CS-Model were prompted to redesign water filtration systems and test them, but they were not offered scaffolds to support reflection on why models change and how the process occurs. Further research can include on-task scaffolds to assist participants in developing an epistemological understanding of scientific modeling practices as they perform them.

It is important to cautiously consider results about Erica's and Anne's epistemological understanding given that they had previously attended the CS-Model pilot study. Before CS-Model, Anne's epistemological understanding was categorized as intermediate (level 2) for nature of models and changing models, and as a basic understanding (level 1) of multiple models and purpose of models. Anne did not participate in post-interviews, so she could not be included in statistical analyses. Before CS-Model, Erica's performance was similar to other participants. After CS-Model, Erica achieved the highest level of model knowledge in nature of models, purpose of models, and changing models. Developing preservice science teachers' epistemological understanding requires time and extended training. It is possible that the short duration of the CS-Model pilot study did not make a significant impact on Erica's performance during pre-interviews. However, she may have experienced a cumulative effect after participating in the main study. This may have influenced the high scores assigned to her post-interview responses.

Qualitative analysis examined participants who experienced change in their assigned levels of epistemological understanding after CS-Model. It was noteworthy that only two participants (Mary and Fiona) remained in level 1 for multiple models (Mary) and changing models (Fiona). Moreover, three participants experienced a regression in their assigned scores after CS-Model. Simon's score decreased from level 3 to level 2 in purpose of models. Sam and Jayden's score decreased from level 2 to level 1 in changing models. It is possible that the CS-Model module was not long enough to meaningfully influence participants' epistemological understanding. Moreover, CS-Model featured limited face-to-face time and several asynchronous online activities. Another possibility is that relying solely on interviews to elicit and assess participants' epistemologies may not have been an effective strategy. This invites future investigations that feature an extended CS-Model timeline, as well as multiple assessment methods on participants' epistemological understanding, including assessment of on-task discourse (Wilkerson et al., 2016) during activities with physical experiments and simulation coding.

Understanding of Computer Science Concepts

Before CS-Model, participants believed that loops (repeat block) would yield an infinite loop or replay the entire simulation. These are misconceptions about the number and scope of loops, and how they affect the simulation output. These have been identified as common misconceptions among novice programmers in Qian and Lehman's (2017) literature review. Grover and Basu (2017) found similar misconceptions among middle school students, who struggled to determine which part of the code should be looped. Quantitative analysis showed statistically significant differences in participants' definitions and application of loops after CS-Model. Qualitative analysis showed that all but one participant corrected misconceptions and

provided accurate explanations about loops after CS-ModeL. In a study with K-12 students, Mladenović, Boljat, and Žanko (2018) found that Scratch users overperformed their counterparts who used Logo and Python by correcting misconceptions about loops.

Regarding the delay (wait seconds block), several participants showed confusion during pre-interviews regarding whether it would pause loops or execute two adjacent blocks or the entire simulation. Participants could not determine the extent to which a delay would affect the simulation. Statistically significant differences were only found in participants' use of proper terminology. Differences in participants' definitions and application were not statistically significant. Qualitative results showed that four participants still held misconceptions after CS-ModeL. Four participants presented misconceptions about this block after CS-ModeL. Misconceptions among novice programmers are common regarding loops (Grover & Basu, 2017; Qian & Lehman, 2017), delays (Booth & Stumpf, 2013), and conditionals (Qian & Lehman, 2017; Swidan, Hermans, & Smit, 2018).

Regarding conditionals (when I receive and broadcast blocks), statistical analysis revealed statistically significant results after CS-ModeL. Qualitative analysis showed that most participants either had misconceptions or could not provide an explanation during pre-interviews. Most participants were confused about which block was a message receiver (input) or message receiver (output). Several of them were not sure how to explain input-output relationships between the two blocks and how the blocks would affect the simulation. These knowledge gaps are identified in Qian and Lehman (2017) and in Swidan et al. (2018) as common challenges involving learning to code conditional statements. Along these lines, Kwon (2017) found that preservice teachers failed to understand and use variables to create if-then statements in text-based programming environments. Grover and Basu's (2017) study found that middle schoolers

could not successfully define input and output variables to code if-then statements in Scratch. These authors argued that conditionals are complex programming statements that need to be addressed to promote conceptual understanding and practice.

It is possible that participants' enhanced understanding and successful application of computer science concepts is positively influenced by the inclusion of a coding workshop wherein participants learned and practice concepts/blocks prior to using them for scientific modeling purposes. Perhaps participants' experience with the coding workshop reduced participants' cognitive load during scientific modeling activities, as recommended in Vasconcelos and Kim (2019). CS-Model also features pair coding activities. During post-interviews, all participants explained that working with a teammate was helpful as they supported each other by explaining concepts, jointly debugging errors, and deciding how to code the simulation. Correcting misconceptions about computer science concepts and their function is critical for learning to code (Swidan et al., 2018), and further investigation on scaffolding strategies for enhanced understanding is needed. Future research may also examine participants' discourse during simulation coding to analyze if they express misconceptions and how those are corrected through pair coding.

Results about Anne's and Erica's performance should be considered with caution given their previous experience with CS-Model. Before CS-Model, Anne was at the intermediate level (1) of understanding for conditionals and delays, but she was assigned to basic level (0) regarding understanding of loops. Anne did not attend post-interviews, so it was neither possible to include her in statistical analyses nor draw conclusions about changes in her understanding of computer science concepts. Before CS-Model, Erica was at the intermediate level (1) of conceptual understanding of conditionals and loops, as well as application of delays. After CS-

Model, she reached the advanced level (2) in all categories but definition of delays. Erica's performance after CS-Model was similar to most participants who were not in the pilot study. From one perspective, Erica's performance was probably influenced by a cumulative effect caused by her extended participation in CS-Model. From another perspective, several other participants also reached advanced levels. It is possible that the coding workshop offered early in the CS-Model module was beneficial to leverage participants' knowledge of conditionals, delays, and loops.

Use of Coding in Scientific Modeling Lessons

Most participants designed lessons that featured student-led and authentic scientific inquiry activities, which were well-aligned with the selected NGSS standards. Participants' use of authentic inquiry could be due to their immersive scientific modeling experience with physical and virtual experiments. Teachers tend to replicate the methods that they are taught with once they take on instructional responsibilities (Momsen, Long, Wyse, & Ebert-May, 2010). This claim is supported by the literature, which shows that preservice teachers who are exposed to practice-oriented science teaching show enhanced epistemological understanding of science, scientific methods (Adamson et al., 2003; Schwarz, 2009) and increased use of such methods in their lessons (Koenig, Schen, & Bao, 2012).

Most lessons featured well-aligned computer science concepts with proposed standards selected from the K-12 Computer Science Framework (2016). In addition, most participants designed well-connected computer science and science tasks, and they addressed almost all scientific modeling steps listed in the rubric. Perhaps offering preservice teachers exemplary scientific modeling lessons (Kenyon, Davis, & Hug, 2011) and simulations targeting topics in their area of teaching (e.g., life sciences, biology) was beneficial to help them understand and

envision teaching with block-based coding. All participants designed activities that entailed model construction, but several participants failed to address other scientific modeling tasks such as model testing or revision. Similarly, Cotterman and Kenyon (2009) found that preservice teachers successfully addressed model construction but not model evaluation and revision in their lessons.

Thematic analysis revealed that participants integrated coding into lessons either as a research tool to test hypotheses and models or as an exploration tool to simply manipulate variables. Although one would expect that participants' epistemologies would be reflected in how they use block-based coding to support scientific modeling in lessons, such relationship was not found. Along these lines, Schwarz, Meyer, and Sharma's (2007) study found that preservice teachers showed a more refined understanding of technology-enhanced science teaching after implementation of a scientific inquiry module using computer-based simulations. However, they struggled to propose lesson ideas involving scientific modeling, scientific inquiry, and use of simulations for scientific modeling. Likewise, Windschitl et al. (2008b) found that secondary preservice science teachers' understanding of models was "roughly predictive of the degree of sophistication they employed in using model-based instruction" (p. 363) in their lessons.

Preservice and in-service science teachers' epistemological understanding is often not aligned with their actual lesson design or implementation. Inconsistency between participants' epistemologies and their practice of modeling (e.g., Crawford & Cullin, 2004; Schwarz, 2009), as well as with their use of simulations for modeling purposes (Wilkerson et al., 2016) was identified in previous studies. Moreover, studies have found that groups of preservice teachers who received training on how to reflect about their pedagogies, use of technologies, and how to connect both to transform learners (Valanides & Angeli, 2005, 2008) designed higher quality

computer-enhanced lessons compared to control groups. Future research on CS-Model can integrate scaffolds for participants to reflect on and design scientific modeling lessons using coding that embody their epistemological understanding of models and modeling.

Thematic analysis also revealed that most participants targeted computer science practices, and few targeted specific concepts in lessons. This is partly due to participants' limited experience with coding. Another possibility is that several scientific modeling practices (e.g., model revision) are closely related to computer science practices (e.g., being iterative). An extended discussion on the overlap between scientific modeling and computer science practices is offered in Sengupta et al. (2013). Future redesign of the CS-Model tool can include scaffolding instructions so that preservice teachers address both computer science concepts and practices (Barr & Stephenson, 2011; Brennan & Resnick, 2012) in their lessons in order to specifically describe tasks that students are expected to perform and concepts that they are expected to learn.

It is important to note that Marcus and Erica had previously used the CS-Model tool to design a lesson during the pilot study. Erica and Marcus were two of the few participants who addressed every item from the rubric in their lessons. Although these participants did not receive feedback on lessons designed during the pilot study, it is probable that their previous experience influenced their lesson design. Similar to other participants, Erica's epistemological understanding of models and modeling was not reflected in her lesson. She was assigned to the advanced level 3 for purpose of models though her lesson featured use of block-based coding simulation as a tool to explore and play with variables rather than to promote authentic inquiry. It was not possible to make such comparisons about Marcus because he did not attend the post-interview.

Conclusions and Future Research

National and state learning standards urge K-12 teachers to design authentic STEM learning experiences, such as using and developing models to produce knowledge about a phenomenon (NRC, 2012; NGSS Lead States, 2013). Teachers are asked to serve as the point of connection between scientific knowledge and students. To accomplish this task, preservice teachers need to experience authentic scientific inquiry themselves in teacher education programs (Kenyon et al., 2011; Schwarz et al., 2009; Webster-Wright, 2009) so they can develop content knowledge and pedagogical content knowledge (Vasconcelos & Kim, 2019; Weiss & Pasley, 2006).

Calls have also been made to integrate computer science practices, such as coding, into K-12 learning environments that can benefit from it (Paul, 2016; Wing, 2006; Yadav, Stephenson, & Hong, 2017). In this paper, we propose that using block-based coding to create science simulations and model science phenomena is a promising approach to support scientific model development while also learning computer science. This approach is informed by constructionist learning (Harel & Papert, 1991; Kafai, 2012; Papert, 1980) as it aims to help one express and refine their own models through construction, testing, evaluation, and revision of simulations. This interdisciplinary approach is expected to promote science and computer science learning.

The present study was designed to support preservice teachers in learning to code simulations of science phenomena and designing lessons in which simulation coding is used to support scientific modeling. The redesigned CS-ModeL was implemented in a teacher education course. This mixed methods study investigated if and how preservice science teachers' epistemological understanding of models and modeling in addition to their understanding of

computer science concepts changed after CS-Model. The study also examined how preservice teachers used coding in scientific modeling lessons. Quantitative and qualitative results suggest that many participants' epistemological understanding improved regarding the nature of models, purpose of models, multiple models, and changing models. Statistically significant differences were found in the two first variables. Future research implementing CS-Model can include scaffolded activities in which preservice teachers (a) examine, dissect, and discuss competing models of the same phenomenon, as well as reflect on why multiple models exist, and (b) discuss the reasons and the process of changing and developing models to generate knowledge (Chu et al., 2017; Gouvea & Passmore, 2017; Stammen et al., 2018).

Results suggest that many participants corrected misconceptions and provided more refined explanations about computer science concepts, though not all variables were statistically significant. A few participants still held misconceptions after the study. Epistemic and conceptual scaffolds (Belland, 2017; Sandoval & Reiser, 2004) can be provided in future research so participants discuss how computer science concepts materialize variables of interest in science simulations. Alternative methods for assessing conceptual understanding should be used in addition to interviews such as statistically validated surveys or tests (Field, 2013; Knapp & Mueller, 2010; Leung, 2015) on computer science knowledge, as well as case scenarios (Grover et al., 2014) that involve applying concepts, identifying, and fixing faulty code.

Study results also revealed that most lessons featured well-integrated computer science and science tasks, and that most participants addressed computer science practices, not concepts. Future research can provide scaffolds so teachers can explicitly connect computer science concepts with science concepts, and computer science practices with scientific modeling practices. All lessons featured model construction, though several participants failed to address

all scientific modeling steps. Future research may include peer feedback on lesson design and lesson implementation (Carrier, 2011; Herrenkohl, Tasker, & White, 2011; Kazempour & Amirshokoochi, 2014) so participants can further develop lessons and experience what it entails to implement coding-enhanced scientific modeling lessons respectively. Results also suggest that participants' epistemological understanding of models and modeling was not reflected in their lessons. Further investigation is invited to analyze how training on how to materialize one's epistemological understanding in lesson design and implementation affects their lessons.

CS-Model is a promising approach to provide professional learning on scientific modeling and teaching with coding to preservice science teachers. Teacher educators can adopt CS-Model in their teacher preparation courses. CS-Model can be adapted to address various topics and train teachers from different areas in science education to teach scientific modeling with block-based coding and science simulations.

Study Limitations

Results of this study need to be carefully considered given its limitations. First, a small number of participants attended both pre- and post-interviews, which led to a small sample size for quantitative analysis. Second, the study did not have a control group, which would have enabled comparisons between participants who attended or did not attend CS-Model. Third, participants' knowledge of computer science concepts and epistemological understanding of models and modeling were assessed using their responses to interviews rather than a validated test or survey. Fourth, two participants (Erica and Marcus) attended the pilot implementation of CS-Model. Coincidentally, they also attended the course where the main dissertation study was conducted. These participants probably experienced a cumulative effect compared to the others who attended CS-Model for the first time. Their results should be considered with caution.

Fifth, CS-Model included several course assignments that were not included in data analysis because they did not help answer research questions. Sixth, participants had limited face-to-face exposure to CS-Model activities due to constraints in the teacher education course. Finally, analysis for RQ3 included designed lessons but did not capture the lesson design process.

References

- Adamson, S. L., Banks, D., Burtch, M., Cox, F. I. I. I., Judson, E., Turley, J. B., ... Lawson, A. E. (2003). Reformed undergraduate instruction and its subsequent impact on secondary school teaching practice and student achievement. *Journal of Research in Science Teaching*, 40(10), 939–957. <https://doi.org/10.1002/tea.10117>
- Aivaloglou, E., & Hermans, F. (2016). How kids code and how we know: An exploratory study on the Scratch repository. *Proceedings of the 2016 ACM Conference on International Computing Education Research*, 53–61. <https://doi.org/10.1145/2960310.2960325>
- Akerson, V. L., Townsend, J. S., Donnelly, L. A., Hanson, D. L., Tira, P., & White, O. (2009). Scientific modeling for inquiring teachers network (SMIT’N): The influence on elementary teachers’ views of nature of science, inquiry, and modeling. *Journal of Science Teacher Education*, 20(1), 21–40. <http://dx.doi.org/10.1007/s10972-008-9116-5>
- 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, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM*, 60(6), 72–80.
- Belland, B. R. (2017). *Instructional scaffolding in STEM education*. <https://doi.org/10.1007/978-3-319-02565-0>
- Berland, L. K., Schwarz, C. V., Krist, C., Kenyon, L., Lo, A. S., & Reiser, B. J. (2016). Epistemologies in practice: Making scientific practices meaningful for students. *Journal of Research in Science Teaching*, 53(7), 1082–1112. <https://doi.org/10.1002/tea.21257>

- Blair, E. (2015). A reflexive exploration of two qualitative data coding techniques. *Journal of Methods and Measurement in the Social Sciences*, 6(1), 14–29.
- Blikstein, P., & Wilensky, U. (2009). An atom is known by the company it keeps: A constructionist learning environment for materials science using agent-based modeling. *International Journal of Computers for Mathematical Learning*, 14(1), 81–119.
- Bloch, M., Buchanan, L., Katz, J., & Quealy, K. (2018). An extremely detailed map of the 2016 election [Interactive map of election results]. *The New York Times*. Retrieved from <https://www.nytimes.com/interactive/2018/upshot/election-2016-voting-precinct-maps.html#3.74/39.57/-97.98>
- Booth, T., & Stumpf, S. (2013). End-user experiences of visual and textual programming environments for Arduino. In Y. Dittrich, M. Burnett, A. Mørch, & D. Redmiles (Eds.), *End-User Development* (Vol. 7897, pp. 25–39). https://doi.org/10.1007/978-3-642-38706-7_4
- Boyatzis, R. E. (1998). *Transforming qualitative information: Thematic analysis and code development*. Thousand Oaks, CA: Sage Publications.
- Boyd, R., Gasper, P., & Trout, J. D. (n.d.). *The philosophy of science*. Cambridge, MA: MIT Press.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada*, 1–25. Retrieved from <http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf>

- Buckley, B. C. (2000). Interactive multimedia and model-based learning in biology. *International Journal of Science Education*, 22(9), 895–935.
<https://doi.org/10.1080/095006900416848>
- Buckley, B. C. (2012). Model-based learning. In N. M. Seel (Ed.), *Encyclopedia of the sciences of learning* (pp. 2300–2303). Retrieved from
http://www.springerlink.com/index/10.1007/978-1-4419-1428-6_589
- Carrier, S. J. (2011). Implementing and integrating effective teaching strategies including features of lesson study in an elementary science methods course. *The Teacher Educator*, 46(2), 145–160. <https://doi.org/10.1080/08878730.2011.552666>
- Cheng, M., Lin, J., Chang, Y., Li, H., Wu, T., & Lin, D. (2014). Developing explanatory models of magnetic phenomena through model-based inquiry. *Journal of Baltic Science Education*, 13(3), 351–360.
- Chiou, G. L., & Anderson, O. R. (2010). A study of undergraduate students understanding of heat conduction based on mental model theory and an ontology-process analysis. *Science Education*, 94(5), 825–854.
- Chu, S. L., Deuermeyer, E., & Quek, F. (2017). Supporting scientific modeling through curriculum-based making in elementary school science classes. *International Journal of Child-Computer Interaction*. <https://doi.org/10.1016/j.ijcci.2017.09.002>
- Clarke, V., & Braun, V. (2013). Teaching thematic analysis: Overcoming challenges and developing strategies for effective learning. *The Psychologist*, 26(2), 120–123.
- Cotterman, M., & Kenyon, L. (2009). *Preservice elementary teachers' design of scientific modeling lessons: The strategies they employ and the understandings they draw upon*.

Presented at the Annual Meeting of the National Association for Research in Science Teaching, Garden Grove, CA.

- Crawford, B., & Cullin, M. (2004). Supporting perspective teachers' conceptions of modeling in science. *International Journal of Science Education*, 26(11), 1379–1401.
- Creswell, J. W., & Miller, D. L. (2000). Determining validity in qualitative inquiry. *Theory Into Practice*, 39(3), 124–130. https://doi.org/10.1207/s15430421tip3903_2
- De Smet, M., Van Keer, H., De Wever, B., & Valcke, M. (2010). Studying thought processes of online peer tutors through stimulated-recall interviews. *Higher Education*, 59(5), 645–661. <https://doi.org/10.1007/s10734-009-9273-2>
- Duschl, R. (2008). Science education in three-part harmony: Balancing conceptual, epistemic, and social learning goals. *Review of Research in Education*, 32(1), 268–291. <https://doi.org/10.3102/0091732X07309371>
- Fereday, J., & Muir-Cochrane, E. (2006). Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. *International Journal of Qualitative Methods*, 5(1), 80–92.
- Field, A. (2013). *Discovering statistics using IBM SPSS statistics: And sex and drugs and rock'n'roll* (3rd ed.). London: Sage Publications.
- Gaddis, G. M., & Gaddis, M. L. (1990). Introduction to biostatistics: Part 5, statistical inference techniques for hypothesis testing with nonparametric data. *Annals of Emergency Medicine*, 19(9), 1054–1059. [https://doi.org/10.1016/S0196-0644\(05\)82571-5](https://doi.org/10.1016/S0196-0644(05)82571-5)
- Gale, N. K., Heath, G., Cameron, E., Rashid, S., & Redwood, S. (2013). Using the framework method for the analysis of qualitative data in multi-disciplinary health research. *BMC Medical Research Methodology*, 13(117), 1–8. <https://doi.org/10.1186/1471-2288-13-117>

- Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation is critical. *ACM Inroads*, 1(1), 61–66.
- Gilbert, J. K., & Boulter, C. J. (2000). *Developing models in science education*. Netherlands: Kluwer Academic Publishers.
- Gilbert, S. W. (1991). Model building and a definition of science. *Journal of Research in Science Teaching*, 28(1), 73–79. <https://doi.org/10.1002/tea.3660280107>
- Glaser, B. G., & Strauss, A. L. (1967). *The discovery of grounded theory: Strategies for qualitative research*. Chicago, IL: Aldine.
- Google, & Gallup. (2015). *Images of computer science: Perceptions among students, parents and educators in the U.S.* Retrieved from <http://g.co/cseduresearch>
- Gouvea, J., & Passmore, C. (2017). ‘Models of’ versus ‘models for’: Toward an agent-based conception of modeling in the science classroom. *Science & Education*, 26(1–2), 49–63. <https://doi.org/10.1007/s11191-017-9884-4>
- Greene, J. C. (2007). *Mixed methods in social inquiry*. San Francisco, CA: Jossey-Bass.
- Grosslight, L., Unger, C., Jay, E., & Smith, C. L. (1991). Understanding models and their use in science: Conceptions of middle and high school students and experts. *Journal of Research in Science Teaching*, 28(1), 73–79. <http://dx.doi.org/10.1002/tea.3660280907>
- Grover, S., & Basu, S. (2017). Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE '17*, 267–272. <https://doi.org/10.1145/3017680.3017723>

- Grover, S., Cooper, S., & Pea, R. (2014). Assessing computational learning in K-12. *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, 57–62. <https://doi.org/10.1145/2591708.2591713>
- 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>
- Harel, I., & Papert, S. (1991). *Constructionism: Research reports and essays, 1985-1990*. Norwood, NJ: Ablex Publishing.
- Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education*, 22(9), 1011–1026. <http://dx.doi.org/10.1080/095006900416884>
- Hernández, M. I., Couso, D., & Pintó, R. (2015). Analyzing students' learning progressions throughout a teaching sequence on acoustic properties of materials with a model-based inquiry approach. *Journal of Science Education and Technology*, 24(2–3), 356–377. <https://doi.org/10.1007/s10956-014-9503-y>
- Herrenkohl, L. R., Tasker, T., & White, B. (2011). Pedagogical practices to support classroom cultures of scientific inquiry. *Cognition and Instruction*, 29(1), 1–44. <https://doi.org/10.1080/07370008.2011.534309>
- Hesse-Biber, S. (2010). Qualitative approaches to mixed methods practice. *Qualitative Inquiry*, 16(6), 455–468. <https://doi.org/10.1177/1077800410364611>
- Hofer, B. K. (2001). Personal epistemological research: Implications for learning and teaching. *Journal of Educational Psychology Review*, 13(1), 353–383.

- Hokayem, H., & Schwarz, C. (2014). Engaging fifth graders in scientific modeling to learn about evaporation and condensation. *International Journal of Science and Mathematics Education, 12*(1), 49–72. <https://doi.org/10.1007/s10763-012-9395-3>
- Holbert, N., & Wilensky, U. (2018). Designing educational video games to be objects-to-think-with. *Journal of the Learning Sciences, 1*–41.
<https://doi.org/10.1080/10508406.2018.1487302>
- Holton, J. (2010). The coding process and its challenges. *The Grounded Theory Review, 9*(1), 21–40.
- K-12 Computer Science Framework Steering Committee. (2016). *K–12 computer science framework*. Retrieved from <http://www.k12cs.org>
- Kafai, Y. (2012). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 35–46). New York, NY: Cambridge University Press.
- Kazempour, M., & Amirshokohi, A. (2014). Transitioning to inquiry-based teaching: Exploring science teachers' professional development experiences. *International Journal of Environmental & Science Education, 9*(1), 285–309.
- Kelly, G. J., McDonald, S., & Wickman, P. (2012). Science learning and epistemology. In B. J. Fraser, K. Tobin, & C. J. McRobbie (Eds.), *Second International Handbook of Science Education* (pp. 281–291). Dordrecht, Netherlands: Springer. https://doi.org/10.1007/978-1-4020-9041-7_20
- Kenyon, L., Davis, E. A., & Hug, B. (2011). Design approaches to support preservice teachers in scientific modeling. *Journal of Science Teacher Education, 22*(1), 1–21.
<https://doi.org/10.1007/s10972-010-9225-9>

- Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, 46(5), 767–787.
<https://doi.org/10.1007/s11251-018-9453-5>
- Kim, E., Oliver, J. S., & Jackson, D. F. (2016). *Connecting the imperatives of STEM, NGSS, deep learning and assessment: A conceptual paper*. Presented at the National Association for Research in Science Teaching, Baltimore, MD.
- Kim, Y., & Oliver, J. S. (2018). Supporting preservice teachers' use of modeling: Building a water purifier. *Innovations in Science Teacher Education*, 3(1), 1–14.
- Knapp, T. R., & Mueller, R. O. (2010). Reliability and validity of instruments. In G. R. Hancock & R. O. Mueller (Eds.), *The reviewer's guide to quantitative methods in the social sciences* (pp. 337–341). New York, NY: Routledge.
- Knuuttila, T. (2009). Some consequences of the pragmatic approach to representation: Decoupling the model-target dyad and indirect reasoning. *Proceedings of the Founding Conference of European Philosophy of Science Association*. https://doi.org/10.1007/978-90-481-3263-8_12
- Knuuttila, T. (2011). Modelling and representing: An artefactual approach to model-based representation. *Studies in History and Philosophy of Science*, 42(2), 262–271.
<https://doi.org/10.1016/j.shpsa.2010.11.034>
- Koenig, K., Schen, m., & Bao, L. (2012). Explicitly targeting pre-service teacher scientific reasoning abilities and understanding of nature of science through an introductory science course. *Science Educator*, 21(2), 1–9.
- Konicek-Moran, R., & Keeley, P. (2015). *Teaching for conceptual understanding in science*. Arlington, VA: NSTA Press.

- Krajcik, J., & Merritt, J. (2012). Engaging students in scientific practices: What does constructing and revising models look like in the science classroom? *The Science Teacher*, 79(3), 38–41.
- Krell, M., & Krüger, D. (2016). Testing models: A key aspect to promote teaching activities related to models and modelling in biology lessons? *Journal of Biological Education*, 50(2), 160–173. <https://doi.org/10.1080/00219266.2015.1028570>
- Krell, M., Reinisch, B., & Krüger, D. (2014). Analyzing students' understanding of models and modeling referring to the disciplines biology, chemistry, and physics. *Research in Science Education*, 45(3), 367–393. <https://doi.org/10.1007/s11165-014-9427-9>
- Krell, M., Upmeier zu Belzen, A., & Krüger, D. (2012). Students' understanding of the purpose of models in different biological contexts. *International Journal of Biology Education*, 2(2). Retrieved from <http://dergipark.gov.tr/download/article-file/90020>
- Knuuttila, T. (2011). Modelling and representing: An artefactual approach to model-based representation. *Studies in History and Philosophy of Science*, 42(2), 262–271. <https://doi.org/10.1016/j.shpsa.2010.11.034>
- Kwon, K. (2017). Student's misconception of programming reflected on problem-solving plans. *International Journal of Computer Science Education in Schools*, 1(4), 14. <https://doi.org/10.21585/ijcses.v1i4.19>
- Lederman, N. G., Schwartz, R. S., Abd-El-Khalick, F., & Bell, R. L. (2001). Preservice teachers' understanding and teaching of the nature of science: An intervention study. *The Canadian Journal of Science, Mathematics, and Technology Education*, 1(2), 135–160.

- Lehrer, R., & Schauble, L. (2000). Developing model-based reasoning in mathematics and science. *Journal of Applied Developmental Psychology, 21*(1), 39–48.
[http://dx.doi.org/10.1016/S0193-3973\(99\)00049-0](http://dx.doi.org/10.1016/S0193-3973(99)00049-0)
- Leung, L. (2015). Validity, reliability, and generalizability in qualitative research. *Journal of Family Medicine and Primary Care, 4*(4), 324–327. <https://dx.doi.org/10.4103%2F2249-4863.161306>
- Lowe, R. (2004). Interrogation of a dynamic visualization during learning. *Learning and Instruction, 14*(3), 257–274. <https://dx.doi.org/10.1016/j.learninstruc.2004.06.003>
- Lyle, J. (2003). Stimulated recall: A report on its use in naturalistic research. *British Educational Research Journal, 29*(6), 861–878. <https://doi.org/10.1080/0141192032000137349>
- Maiorana, F., Berry, M., Nelson, M., Lucarelli, C., Phillipps, M., Mishra, S., & Benassi, A. (2017). *International perspectives on CS teacher formation and professional development*. 236–237. <https://doi.org/10.1145/3059009.3059067>
- Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. *Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference*, 1–29. <https://doi.org/10.1145/2713609.2713610>
- McCrum-Gardner, E. (2008). Which is the correct statistical test to use? *British Journal of Oral and Maxillofacial Surgery, 46*(1), 38–41. <https://doi.org/10.1016/j.bjoms.2007.09.002>
- Merrill, S. (2017, December 7). The future of coding in schools. Retrieved from Edutopia website: <https://www.edutopia.org/article/future-coding-schools>

- Mladenović, M., Boljat, I., & Žanko, Ž. (2018). Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. *Education and Information Technologies*, 23(1), 1483–1500. <https://doi.org/10.1007/s10639-017-9673-3>
- Momsen, J. L., Long, T. M., Wyse, S. A., & Ebert-May, D. (2010). Just the facts? Introductory undergraduate biology courses focus on low-level cognitive skills. *CBE - Life Sciences Education*, 9(4), 435–440.
- Namdar, B., & Shen, J. (2015). Modeling-oriented assessment in K-12 science education: A synthesis of research from 1980 to 2013 and new directions. *International Journal of Science Education*, 37(7), 993–1023. <https://doi.org/10.1080/09500693.2015.1012185>
- National Research Council. (1996). *National science education standards: Observe, interact, change, learn*. Washington, DC: National Academies Press.
- National Research Council. (2000). *Inquiry and the national science education standards: A guide for teaching and learning*. Washington, DC: National Academies Press.
- National Research Council. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. Washington, DC: The National Academies Press.
- Nersessian, N. J. (2008). *Creating scientific concepts*. Cambridge, MA: MIT Press.
- NGSS Lead States. (2013). *Next Generation Science Standards: For states, by states*. Washington, DC: National Academies Press.
- Osborne, J. (2014). Teaching scientific practices: Meeting the challenge of change. *Journal of Science Teacher Education*, 25(2), 177–196. <https://doi.org/10.1007/s10972-014-9384-1>
- Papadakis, S., Kalogiannakis, M., Zaranis, N., & Orfanakis, V. (2016). Using Scratch and App Inventor for teaching introductory programming in secondary education. A case study.

International Journal of Technology Enhanced Learning, 8(3/4), 217–233.

<https://doi.org/10.1504/IJTEL.2016.10001505>

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Passmore, C., Schwarz, C. V., & Mankowski, J. (2016). Developing and using models. In C. V. Schwarz, C. Passmore, & B. J. Reiser (Eds.), *Helping students make sense of the world using next generation science and engineering practices* (pp. 109–134).

<https://doi.org/10.2505/9781938946042>

Paul, A. M. (2016). The coding revolution. *Scientific American*, 35(1), 42–49.

<https://doi.org/10.1038/scientificamerican0816-42>

Plano Clark, V., & Creswell, J. W. (2011). *Designing and conducting mixed methods research* (2nd ed.). Los Angeles, CA: Sage Publications.

Price, T. W., & Barnes, T. (2015). Comparing textual and block interfaces in a novice programming environment. *Proceedings of the Eleventh Annual International Computing Education Research*, 91–99. <https://doi.org/10.1145/2787622.2787712>

Project Growing Up Thinking Scientifically (GUTS). (n.d.). Retrieved from <http://www.projectguts.org/resources>

Qian, Y., & Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education*, 18(1), 1–24. <https://doi.org/10.1145/3077618>

Rea-Ramirez, M. A., Clement, J., & Núñez-Oviedo, M. C. (2008). An instructional model derived from model construction and criticism theory. In J. Clement & M. A. Rea-

- Ramirez (Eds.), *Model based learning and instruction in science*. (pp. 23–43). Dordrecht, Netherlands: Springer.
- Reinisch, B., & Krüger, D. (2018). Preservice biology teachers' conceptions about the tentative nature of theories and models in biology. *Research in Science Education*, 48(1), 71–103. <https://doi.org/10.1007/s11165-016-9559-1>
- Renken, M., Peffer, M., Otrell-Cass, K., Girault, I., & Chiocciariello, A. (2016). *Simulations as scaffolds in science education*. Cham: Springer.
- Roth, K. J. (1990). Developing meaningful conceptual understanding in science. In B. F. Jones & L. Idol (Eds.), *Dimensions of thinking and cognitive instruction* (pp. 139–175). Hillsdale, N.J: Erlbaum.
- Roulston, K. (2010). *Reflective interviewing: A guide to theory and practice*. Los Angeles, CA: Sage.
- Samarapungavan, A., Tippins, D., & Bryan, L. (2015). A modeling-based inquiry framework for early childhood science learning. In K. C. Trundle & M. Saçkes (Eds.), *Research in Early Childhood Science Education* (pp. 259–277). https://doi.org/10.1007/978-94-017-9505-0_12
- Sandelowski, M., Voils, C. I., & Knafl, G. (2009). On quantizing. *Journal of Mixed Methods Research*, 3(3), 208–222. <https://doi.org/10.1177/1558689809334210>
- Sandoval, W. A., & Reiser, B. J. (2004). Explanation-driven inquiry: Integrating conceptual and epistemic scaffolds for scientific inquiry. *Science Education*, 88(3), 345–372. <https://doi.org/10.1002/sce.10130>

- Schwarz, C. (2009). Developing preservice elementary teachers' knowledge and practices through modeling-centered scientific inquiry. *Science Education*, 93(4), 720–744. <https://doi.org/10.1002/sce.20324>
- Schwarz, C. V., & Gwekwerere, Y. N. (2007). Using a guided inquiry and modeling instructional framework (EIMA) to support pre-service K-8 science teaching. *Science Education*, 91(1), 158–186.
- Schwarz, C. V., Meyer, J., & Sharma, A. (2007). Technology, pedagogy, and epistemology: Opportunities and challenges of using computer modeling and simulation tools in elementary science methods. *Journal of Science Teacher Education*, 18(2), 243–269. <https://doi.org/10.1007/s10972-007-9039-6>
- Schwarz, C. V., Reiser, B. J., Davis, E. A., Kenyon, L., Achér, A., Fortus, D., ... Krajcik, J. (2009). Developing a learning progression for scientific modeling: Making scientific modeling accessible and meaningful for learners. *Journal of Research in Science Teaching*, 46(6), 632–654. <https://doi.org/10.1002/tea.20311>
- Schwarz, C. V., & White, B. Y. (2005). Metamodeling knowledge: Developing students' understanding of scientific modeling. *Cognition and Instruction*, 23(2), 165–205. http://dx.doi.org/10.1207/s1532690xci2302_1
- Schwartz, R. S., Lederman, N. G., & Crawford, B. A. (2004). Developing views of nature of science in an authentic context: An explicit approach to bridging the gap between nature of science and scientific inquiry. *Science Education*, 88(1), 610-645.
- Seel, N. M. (2017). Model-based learning: A synthesis of theory and research. *Educational Technology Research and Development*, 65(4), 931–966. <https://doi.org/10.1007/s11423-016-9507-9>

- 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>
- Shen, J., Lei, J., Chang, H., & Namdar, B. (2014). Technology-enhanced, modeling-based instruction (TMBI) in science education. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of research on educational communications and technology* (pp. 529–540). https://doi.org/10.1007/978-1-4614-3185-5_41
- Siegel, S. Y. (1957). Nonparametric statistics. *The American Statistician, 11*(3), 13–19.
<https://doi.org/10.2307/2685679>
- Stammen, A., Malone, K., & Irving, K. (2018). Effects of modeling instruction professional development on biology teachers' scientific reasoning skills. *Education Sciences, 8*(3), 1–19. <https://doi.org/10.3390/educsci8030119>
- Strauss, A., & Corbin, J. M. (1998). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Thousand Oaks, CA: Sage.
- Swidan, A., Hermans, F., & Smit, M. (2018). Programming misconceptions for school students. *Proceedings of the 2018 ACM Conference on International Computing Education Research - ICER '18*, 151–159. <https://doi.org/10.1145/3230977.3230995>
- Teddlie, C., & Tashakkori, A. (2006). A general typology of research designs featuring mixed methods. *Research in the Schools, 13*(1), 12–28.
- Tracy, S. J. (2010). Qualitative quality: Eight “big-tent” criteria for excellent qualitative research. *Qualitative Inquiry, 16*(10), 837–851.
<https://doi.org/10.1177/1077800410383121>

- Upmeier zu Belzen, A., & Krüger, D. (2010). Modellkompetenz im Biologieunterricht [Model competence in biology teaching]. *Zeitschrift Für Didaktik Der Naturwissenschaften*, *16*, 41–57.
- Valanides, N., & Angeli, C. (2005). Preservice teachers as ICT designers: An instructional design model based on an expanded view of pedagogical content knowledge. *Journal of Computer Assisted Learning*, *21*(4), 292–302.
- Valanides, N., & Angeli, C. (2008). Learning and teaching about scientific models with a computer-modeling tool. *Computers in Human Behavior*, *24*(2), 220–233.
<https://doi.org/10.1016/j.chb.2007.01.005>
- Van Driel, J. H., & Verloop, N. (1999). Teachers' knowledge of models and modelling in science. *International Journal of Science Education*, *21*(11), 1141–1153.
- Vasconcelos, L., & Kim, C. (under review). Coding in scientific modeling lessons (CS-Model). *Under Review in Educational Technology Research & Development*.
- Vasconcelos, L., & Kim, C. (2019). *Integrating block-based coding into scientific modeling lessons*. Presented at the American Educational Research Association Annual Meeting, Toronto, Canada.
- Webster-Wright, A. (2009). Reframing professional development through understanding authentic professional learning. *Review of Educational Research*, *79*(2), 702–739.
<https://doi.org/10.3102/0034654308330970>
- Weintrop, D. (2015). Blocks, text, and the space between: The role of representations in novice programming environments. *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 301–302. Atlanta, GA: IEEE.

- Weiss, I. R., & Pasley, J. D. (2006). *Scaling up instructional improvement through teacher professional development: Insights from the local systemic change initiative*. Retrieved from https://repository.upenn.edu/cpre_policybriefs/32
- White, B. Y., & Frederiksen, J. R. (1998). Inquiry, modeling, and metacognition: Making science accessible to all students. *Cognition and Instruction, 16*(1), 3–118.
https://doi.org/10.1207/s1532690xci1601_2
- Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology, 8*(1), 3–19.
- Wilkerson, M. H., Andrews, C., Shaban, Y., Laina, V., & Gravel, B. E. (2016). What's the technology for? Teacher attention and pedagogical goals in a modeling-focused professional development workshop. *Journal of Science Teacher Education, 27*(1), 11–33. <https://doi.org/10.1007/s10972-016-9453-8>
- Windschitl, M., Thompson, J., & Braaten, M. (2008a). Beyond the scientific method: Model-based inquiry as a new paradigm of preference for school science investigations. *Science Education, 92*(5), 941–967. <http://doi.dx.org/10.1002/sce.20259>
- Windschitl, M., Thompson, J., & Braaten, M. (2008b). How novice science teachers appropriate epistemic discourses around model-based inquiry for use in classrooms. *Cognition and Instruction, 26*(3), 310–378. <https://doi.org/10.1080/07370000802177193>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.
- Xiang, L., & Passmore, C. (2015). A framework for model-based inquiry through agent-based programming. *Journal of Science Education and Technology, 24*(1), 311–329.
<https://doi.org/10.1007/s10956-014-9534-4>

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education.

Communications of the ACM, 60(4), 55–62.

Zacharia, Z. (2003). Beliefs, attitudes, and intentions of science teachers regarding the educational use of computer simulations and inquiry-based experiments in physics.

Journal of Research in Science Teaching, 40(8), 792–823.

<https://doi.org/10.1002/tea.10112>

Zacharia, Z. C. (2007). Comparing and combining real and virtual experimentation: An effort to enhance students' conceptual understanding of electric circuits. *Journal of Computer*

Assisted Learning, 23(2), 120–132. <https://doi.org/10.1111/j.1365-2729.2006.00215.x>

Zucker, D. M. (2009). How to do a case study. In *Teaching Research Methods in the Humanities and Social Sciences*. Retrieved from http://works.bepress.com/donna_zucker/14/

CHAPTER 5

CONCLUSION

This purpose of this dissertation research was to scaffold science teachers' learning to code science simulations and to effectively integrate coding into scientific modeling lessons. This research was motivated by two pressing challenges in K-12 education. First, many K-12 science teachers lack an accurate understanding of scientific models and how to integrate authentic scientific modeling into their classrooms (Akerson et al., 2009; Krell & Krüger, 2016; Schwarz & Gwekwerere, 2007; Stammen, Malone, & Irving, 2018). Scientific modeling is a critical skill for K-12 students as it entails authentic use, manipulation, and development of models (Kim & Oliver, 2018; NGSS Lead States, 2013; Samarapungavan, Tippins, & Bryan, 2015). Second, most K-12 teachers are not typically prepared for teaching with coding (Barr & Stephenson, 2011; Google & Gallup, 2015a, 2015b; Paul, 2016). Coding-enhanced instruction prepares students to solve interdisciplinary problems using computer science concepts and processes (Baratè, Ludovico, Mangione, & Rosa, 2015; Lye & Koh, 2014; Obama, 2016; Paul, 2016; Wing, 2006).

To accomplish the purpose of this dissertation research, an instructional module and online tool named Coding in Scientific Modeling Lessons (CS-Model) was designed and developed based on guidelines that emerged from a comprehensive review of relevant literature (Vasconcelos & Kim, under review). This literature review focused on scientific modeling, reform-based STEM teaching and learning, epistemic agency, instructional scaffolding, constructionist learning, block-based coding, and science simulations.

Subsequently, a pilot study was conducted to implement, evaluate, and improve CS-Model in spring 2018. CS-Model was implemented in a methods of science teaching course with five preservice and one in-service science teachers. This single-case study investigated how participants (a) engaged in epistemic discourse while coding science simulations, (b) perceived coding as a teaching tool, and (c) integrated simulation coding into scientific modeling lessons. Results suggested that coding simulations in pairs is an effective way to help participants unveil and overcome science misconceptions, in addition to developing a refined consensus model. However, analysis of participants' epistemic discourse revealed that they alternated between epistemic dialogue and dialogue about code error debugging. Lack of debugging skills detracted from the quality of epistemic discourse because teachers would interrupt dialogue about a science phenomenon due to code errors and would not resume their conversation later. Results also revealed few instances of conflict argumentation about the science concept, which suggest that participants were not prepared to engage in epistemic conflict. Results showed that participants perceived coding as a key skill for K-12 students and its use is beneficial for scientific modeling instruction. Results indicated that most participants failed to design lessons that featured authentic scientific inquiry and to connect science and computer science tasks. Results from this pilot study informed the design of additional scaffolding strategies to be integrated into a redesigned CS-Model. Scaffolds included (a) a coding workshop prior to use of coding for modeling purposes, (b) a hypotheses sheet for recording original and revised models, (c) a debugging sheet for documenting strategies used to identify and fix code errors, (d) an empirical data sheet for recording data from physical experiments, and (e) guidance in the CS-Model tool on selecting and addressing both science and computer science standards and tasks in their lessons.

The main dissertation study was conducted in a technology for science teaching course in fall 2018. Eighteen preservice and one in-service science teachers accepted to join as participants. The study investigated if and how participants' epistemological understanding of scientific models and scientific modeling along with their conceptual understanding of computer science concepts changed after participating in CS-Model. Moreover, the study examined participants' use of coding in their designed lessons to support scientific modeling. This predominantly qualitative mixed methods study found a statistically significant difference in participants' epistemological understanding of the nature and purpose of models before and after CS-Model. Changes in their understanding of multiple models and changing models were not statistically significant. Qualitative analysis suggested that many participants developed a more sophisticated epistemological understanding after CS-Model. Results also indicated that participants corrected several misconceptions about loops, delays, and conditionals after participation in CS-Model. Additionally, results indicated that most participants successfully designed lessons that entailed authentic scientific inquiry and well-integrated science and computer science activities. Block-based coding was either used as a tool to explore a science phenomenon or as a research tool to test and develop one's models and hypotheses. Participants mostly addressed computer science practices (e.g., debugging, abstraction) rather than specific computer science concepts (e.g., loops) in their lessons. It was found that participants' level of epistemological understanding of models and modeling was not reflected in their designed scientific modeling lessons. Although results from this study seem promising, they must be carefully considered given study limitations such as small sample size, short duration of CS-Model, and the fact that three participants had experienced CS-Model during the pilot study.

CS-Model has potential as a framework for future professional learning on coding and scientific modeling. This dissertation is a stepping stone towards integration of computer science concepts and practices into K-12 science education. Additional research is needed to refine the design and implementation of CS-Model. Future research directions and implications of this study for research and practice are discussed in the following sections.

Limitations of the Studies and Future Research Directions

Further investigation using CS-Model is needed given the limitations of the studies in this dissertation. First, the sample size was small in both empirical studies, especially for the statistical analyses conducted in the main study. Future studies can implement CS-Model to groups of teachers attending different sections of the same course. A control group can be included to examine differences in participants in the control and experimental groups.

Participants in the main dissertation study were pursuing a degree in science education with different areas of teaching emphasis, such as biology, chemistry, physics, and life sciences. To some extent, this diversity was addressed by the inclusion of exemplary instructional materials and simulations focusing on topics related to their interests. Under ideal circumstances, constructivist learning involves open-ended discovery. But given the limited face-to-face time in teacher education courses, it was not possible to design scientific modeling and coding activities to target topics related to all participants' interests. The researcher experienced this dilemma during the design and development of CS-Model. The topic on water filtration systems limits participants' inquiry in terms of the science concepts and variables to be investigated. Scaffolded decision-making strategies (e.g., creation of research questions during whole-class discussion) were also adopted to guide study participants in their inquiry process. The researcher attempted to balance out the amount of guidance provided by allowing participants to design their own

water filtration systems using combined materials. During recruitment for both empirical studies, the researcher explained to participants that CS-Model can be adapted to target topics in chemistry, physics, life science, and more. Future research can include scientific modeling and coding experiences that target topics in different areas of science education.

Second, there were differences between the two courses wherein CS-Model was implemented. The pilot study course focused on methods of secondary science teaching, and the main study course focused on effective use of technologies for middle and secondary science teaching. It is possible that preservice teachers' learning experiences with each course prior to study implementation affected them differently. For instance, the course on technology-enhanced science teaching may have better prepared participants for using block-based coding as an instructional technology. Under ideal circumstances, future research can target groups of participants attending different sections of the same course.

Third, the empirical studies featured synchronous and asynchronous online activities due to the limited face-to-face time available for implementation of CS-Model. It would be difficult to extend the study timeline given the constraints of teacher education courses. CS-Model can be implemented as a summer workshop for preservice and/or in-service science teachers. With an extended timeline, the researcher can design and offer valuable opportunities for modeling concepts pertinent to science, chemistry, physics, and biology.

Fourth, study participants did not receive prior training on lesson design. Further investigation is needed to examine research designs that feature (a) assessment of preservice science teachers' lesson design prior experience, (b) scaffolded discussion and dissection of instructional lessons and accompanying simulations prior to design of code-enhanced lessons, and (c) formative peer and instructor feedback on lesson design.

Finally, participants designed one lesson as individual homework assignment in both empirical studies. It was not possible to assess how participants decided on the science topic and determined the level of complexity in science learning activities to include block-based coding activities. This invites further research. The dissertation studies also did not investigate actual lesson implementation because the teacher education courses lacked a practicum component. Future studies can offer teachers opportunities for in-person lesson design in pairs and record their discussions to examine how their epistemologies of scientific models, modeling, scientific inquiry, and science manifest during lesson design. Additionally, future studies can investigate how teachers implement their designed lessons in K-12 settings.

Implications for Research and Practice

This dissertation research presented an approach to make scientific modeling and coding accessible to K-12 teachers so they can offer instruction that targets these critical skills to their future students. Results from this research serve as a stepping stone for the design of professional learning experiences for preservice and in-service science teachers within and beyond teacher education courses. Implications for research and practice are discussed as follows.

This dissertation proposed a unique conceptual framework for integrating coding into scientific modeling instruction (Vasconcelos & Kim, under review). Although scientific modeling, block-based coding, and science simulations have been extensively and individually researched, current literature lacks frameworks and empirical studies that aim to support scientific modeling through coding of science simulations. This framework is supported by national K-12 education policies and standards that urge professional learning for teachers on instruction that features authentic and interdisciplinary Science, Technology, Mathematics, and Engineering (STEM) learning. K-12 teachers who take part in professional learning that uses CS-

Model are expected to not develop content knowledge and pedagogical content knowledge on use of coding for scientific modeling instruction. Teachers also learn about authentic scientific inquiry, science simulations, and use of block-based coding beyond modeling. This framework can be implemented in further research and practice by teacher educators seeking to systematically investigate other applications of computer science practices and concepts (e.g., coding, debugging) in a variety of STEM learning contexts.

Findings from the pilot study indicated the potential of using CS-Model to promote epistemic agency in science learning and to support scientific model development (Vasconcelos & Kim, 2019). Using block-based coding to express, manipulate, and develop one's own models of science phenomena is a promising approach. Numerous empirical studies have used science simulations to support science learning though reviews on their effectiveness revealed mixed results (Bell & Smetana, 2008; D'Angelo et al., 2014; Smetana & Bell, 2012). This is partly because simply manipulating simulation variables does not entail authentic epistemic challenges that one would face in the real world when investigating and generating knowledge of a phenomenon. Conversely, constructing artifacts that embody one's own models (Holbert & Wilensky, 2014, 2018; Kafai, 2012; Papert, 1980) to simulate science phenomena empowers learners as epistemic agents (Berland et al., 2016; Gouvea & Passmore, 2017; Knuuttila, 2011), who engage in mindful, personally-relevant, and responsible scientific modeling practices. It is possible to integrate CS-Model into science teacher education courses or professional learning initiatives, so teachers experience scientific modeling from different standpoints, as a student and as an educator.

Findings from the main dissertation study suggest the potential of CS-Model to support development of epistemological understanding of scientific models and modeling, conceptual

understanding of computer science concepts, and design of code-enhanced scientific modeling lessons (Vasconcelos & Kim, in preparation). This dissertation can inform future research and practice focusing on integrated STEM teaching and learning (Berland et al., 2016; Guzdial & Morrison, 2016; Kim, Oliver, & Jackson, 2016). It is important to note that methods of data collection and analysis in the study focused on participants before and after CS-Model. This invites studies that also include a comprehensive assessment of participants' epistemologies in action (Berland et al., 2016), i.e., while performing scientific modeling and coding practices.

The topic of this dissertation research is at the intersection of educational technology, computer science, and science education. On a micro level, CS-Model can be beneficial for training preservice and in-service teachers to design and implement scientific modeling and coding-enhanced instruction. On a societal level, CS-Model serves as a first step towards a research agenda that addresses U.S. K-12 schools' need for teachers capable of teaching scientific modeling and teaching with coding.

References

- Akerson, V. L., Townsend, J. S., Donnelly, L. A., Hanson, D. L., Tira, P., & White, O. (2009). Scientific modeling for inquiring teachers network (SMIT'N): The influence on elementary teachers' views of nature of science, inquiry, and modeling. *Journal of Science Teacher Education*, 20(1), 21–40. <http://dx.doi.org/10.1007/s10972-008-9116-5>
- Baratè, A., Ludovico, L. A., Mangione, G. R., & Rosa, A. (2015). Playing music, playing with music: A proposal for music coding in primary school. *International Association for Development of the Information Society*. Presented at the International Conference for e-Learning. Retrieved from <http://files.eric.ed.gov/fulltext/ED562460.pdf>
- 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>
- Bell, R. L., & Smetana, L. K. (2008). Using computer simulations to enhance science teaching and learning. In R. L. Bell, J. G. Newsome, & J. Luft (Eds.), *Technology in the Secondary Science Classroom* (pp. 23–32). Arlington, VA: NSTA Press.
- Berland, L. K., Schwarz, C. V., Krist, C., Kenyon, L., Lo, A. S., & Reiser, B. J. (2016). Epistemologies in practice: Making scientific practices meaningful for students. *Journal of Research in Science Teaching*, 53(7), 1082–1112. <https://doi.org/10.1002/tea.21257>
- D'Angelo, C., Rutstein, D., Harris, C., Bernard, R., Borokhovski, E., & Haertel, G. (2014). *Simulations for STEM learning: Systematic review and meta-analysis*. Menlo Park, CA: SRI International.
- Google, & Gallup. (2015a). *Images of computer science: Perceptions among students, parents and educators in the U.S.* Retrieved from <http://g.co/cseduresearch>


- Google, & Gallup. (2015b). *Searching for computer science: Access and barriers in U.S. K-12 education*. Retrieved from <http://g.co/cseduresearch>
- Gouvea, J., & Passmore, C. (2017). 'Models of' versus 'models for': Toward an agent-based conception of modeling in the science classroom. *Science & Education*, 26(1–2), 49–63. <https://doi.org/10.1007/s11191-017-9884-4>
- Guzdial, M., & Morrison, B. (2016). Growing computer science education into a STEM education discipline. *Communications of the ACM*, 59(11), 31–33. <https://doi.org/10.1145/3000612>
- Holbert, N., & Wilensky, U. (2014). Constructible authentic representations: Designing video games that enable players to utilize knowledge developed in-game to reason about science. *Technology, Knowledge and Learning*, 19(1–2), 53–79. <https://doi.org/10.1007/s10758-014-9214-8>
- Holbert, N., & Wilensky, U. (2018). Designing educational video games to be objects-to-think-with. *Journal of the Learning Sciences*, 1–41. <https://doi.org/10.1080/10508406.2018.1487302>
- Kafai, Y. (2012). Constructionism. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 35–46). New York, NY: Cambridge University Press.
- Kim, E., Oliver, J. S., & Jackson, D. F. (2016). *Connecting the imperatives of STEM, NGSS, deep learning and assessment: A conceptual paper*. Presented at the National Association for Research in Science Teaching, Baltimore, MD.
- Kim, Y., & Oliver, J. S. (2018). Supporting preservice teachers' use of modeling: Building a water purifier. *Innovations in Science Teacher Education*, 3(1), 1–14.

- Knuuttila, T. (2011). Modelling and representing: An artefactual approach to model-based representation. *Studies in History and Philosophy of Science*, 42(2), 262–271.
<https://doi.org/10.1016/j.shpsa.2010.11.034>
- Krell, M., & Krüger, D. (2016). Testing models: A key aspect to promote teaching activities related to models and modelling in biology lessons? *Journal of Biological Education*, 50(2), 160–173. <https://doi.org/10.1080/00219266.2015.1028570>
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61. <https://doi.org/10.1016/j.chb.2014.09.012>
- NGSS Lead States. (2013). *Next Generation Science Standards: For states, by states*. Washington, DC: National Academies Press.
- Obama, B. (2016). *Computer science for all*. Retrieved from <https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all>
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Paul, A. M. (2016). The coding revolution. *Scientific American*, 35(1), 42–49.
<https://doi.org/10.1038/scientificamerican0816-42>
- Samarapungavan, A., Tippins, D., & Bryan, L. (2015). A modeling-based inquiry framework for early childhood science learning. In K. C. Trundle & M. Saçkes (Eds.), *Research in Early Childhood Science Education* (pp. 259–277). https://doi.org/10.1007/978-94-017-9505-0_12

- Schwarz, C. V., & Gwekwerere, Y. N. (2007). Using a guided inquiry and modeling instructional framework (EIMA) to support pre-service K-8 science teaching. *Science Education*, 91(1), 158–186.
- Smetana, L. K., & Bell, R. L. (2012). Computer simulations to support science instruction and learning: A critical review of the literature. *International Journal of Science Education*, 34(9), 1337–1370. <https://doi.org/10.1080/09500693.2011.605182>
- Stammen, A., Malone, K., & Irving, K. (2018). Effects of modeling instruction professional development on biology teachers' scientific reasoning skills. *Education Sciences*, 8(3), 1–19. <https://doi.org/10.3390/educsci8030119>
- Vasconcelos, L., & Kim, C. (under review). Coding in scientific modeling lessons (CS-Model). *Under Review in Educational Technology Research & Development*.
- Vasconcelos, L., & Kim, C. (2019). *Integrating block-based coding into scientific modeling lessons*. Presented at the American Educational Research Association Annual Meeting, Toronto, Canada.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.


APPENDIX A

PRESENTATION ON SCIENTIFIC MODELS, MODELING, AND CODING



Coding in scientific modeling

Lucas Vasconcelos
Dr. ChanMin Kim (advisor)
Dr. Daniel Capps (course instructor)




https://www.theindychannel.com/stem

Have you ever heard of block-based coding?

Block-based Coding

- Coding is a key skill for K-12 students (K-12 Computer Science Steering Committee, 2016).
- Block-based coding consists of snapping blocks together to create an animation or simulation on the screen. Blocks embody programming commands
- Block-based coding prompts one to externalize their understanding of scientific phenomena while creating dynamic simulations.
- <https://tinyurl.com/ybrqqs18>



Information about study

Today:

- Read, sign, and return one copy of consent form
- Complete demographics survey
- Fill out availability for pre-interview

Upcoming classes:

- Pre-interview on date/time that fits your schedule (audio recorded).
- Course activities that are part of this course, planned with Dr. Capps.
- Observation of participation in activities. Computer screen recording.
- Post-interview on date/time that fits your schedule (audio recorded).

Questions?

APPENDIX B

EXAMPLE OF FOLLOW-UP QUESTIONS USED IN INTERVIEW

Researcher: Can there be different models for the same phenomenon?

Simon: Yes.

Researcher: For example?

Simon: Simple like atoms there. Well. I guess some of the early ones were. More speculative but even still if I remember correctly there are still multiple models of an atom that are all acceptable and all have different shortcomings and focuses and reasons for use.

Researcher: Do models represent absolute reality?

Simon: No. They oftentimes represent maybe an idealistic reality. I can think of like a physics model used by a programmer or something. Recently we did Phet simulations. So that's like a bunch of models. To visualize you know these basic laws. There's also a whole lot of just other variables that are unpredictable and not even unpredictable sometimes... it just treats as negligible like wind resistance and a lot of are... so yeah models are idealistic. They oftentimes just focus on like what we're trying to study, and they don't account for what we can't control.

Researcher: Got it. Would a scientist ever need to change a model?

Simon: Yes.

Researcher: Why?

Simon: Science just is a process. We're constantly learning and relearning things and a model that today could be seen as the perfect most ideal model could tomorrow be discovered to be missing some key component and require rework that could completely change a field. Super vague and I can't really think of an example because it's unlikely, but it could happen.

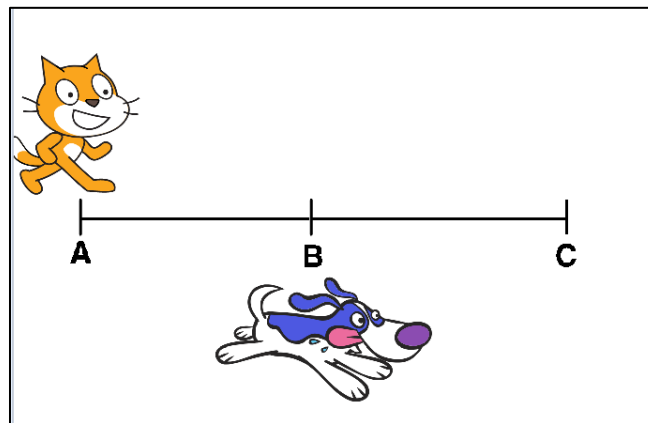
Researcher: You're saying that changing, constructing or reconstructing a model requires reworking it. Which types of tasks that scientists perform are involved in reworking it [model]?

Simon: Yeah. More first they have to identify that it's in need of a rework whatever model there are working with. And then if I had some vague abstract hypothetical model that I realized needed a rework... first I'd realize that. Then I would try to isolate like what it is about that model that needs to be worked with. And if I can isolate it great and if it's something that you know I can isolate and rework separately, that's super cool and I would do that. But I feel like a lot of times that's not the case and I may have to just start all the way from the ground up on like how to determine like how to fix this and not. I don't want to jump the gun and I really don't know much about coding but I had some friends that coded and there was always like jokes about how like you make a teeniest mistake up here you fix it and it just ruins the whole program or whatever. Feel like it's similar to that.

APPENDIX C

CODING PROBLEM APPLIED DURING PRE- AND POST-INTERVIEWS

- This is a block “repeat”. What do you think this block is for?
- This is the block “wait_secs”. What do you think this block is for?
- These are the blocks “when I receive” and “broadcast” What do you think these blocks are for?
- I brought this animation involving a cat and a dog. The simulation is coded so that the cat moves from point A to point C. If you had to make the cat stop at point B, which is halfway between these two points, how would you change the code?
- If you had to make the cat go slower from A to C, how would you change the code?
- If you had to make the cat say meow once the dog is clicked, how would you change the code?



APPENDIX D

SLIDES FOR CODING WORKSHOP

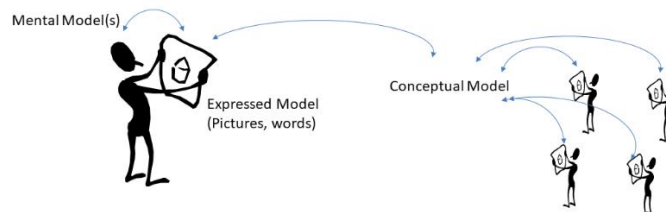
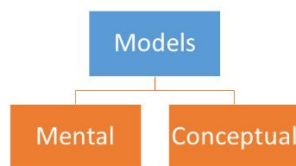
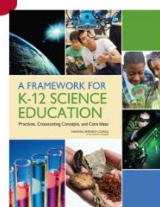


Coding and scientific modeling

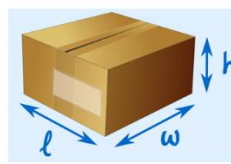
Lucas Vasconcelos

Dr. ChanMin Kim (advisor)

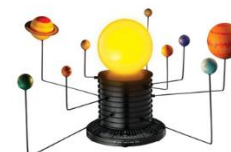
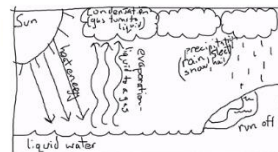
Dr. Daniel Capps (course instructor)



The various forms of a model



Volume = width x height x length



Models of vs. Models for

- What is the purpose of models?
- Models are tools designed by a cognitive agent to investigate a phenomenon, generate knowledge about it, and advance thinking.
 - One's need to investigate a phenomenon drives the creation and manipulation of models.
 - Models are *approximations* of its referent and they allow indirect experimentation
 - Alternative models of the same phenomenon may serve different purposes
 - **Scientific modeling** is the process of generating knowledge about a phenomenon or system while using and manipulating (e.g., creating, testing, evaluating, and revising) models. Through scientific modeling, scientists can predict, explain, test hypotheses...

Overview of Scratch

<https://scratch.mit.edu/>

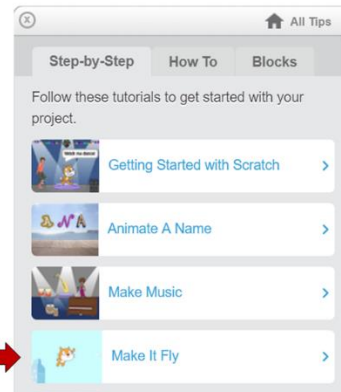
Let's code!

- | | |
|-------------------------------|--------------------------------|
| • Jonathan & Weston | • Alyssa and Emily |
| • Grace and Maggie | • Hayley & Mariah |
| • Shea and Rosemary | • Catherine Riley and Mohammed |
| • Catherine Nord and Chandler | • Emily and Johnisha |
| • Felexia and James | • Rachel and _____ |
| • Randall and Christopher | |

Let's code!

In pairs, complete Scratch challenges of your choice.

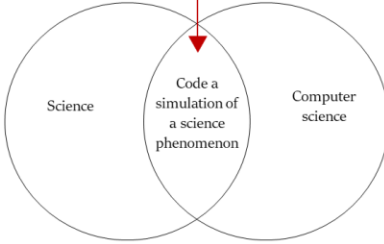
Note: one teammate uses the computer for this activity, the other teammate will use the computer for the next one.



Coding activity 2

Discussion: [airplane takeoff](#) and [cell diffusion](#)

- What science concept(s) could be taught with such a simulation?
- Are there any variables that would be important for your students to explore? What would you have students do with such variables?
- What might be some examples of research questions that would help students investigate such a phenomenon using this simulation?
- How can students use such a simulation as a test bed? Would you change it?
- Would you hand your students the simulation so they can play with it? Or would you guide them through the process of creating a simulation?



A Venn diagram with two overlapping circles. The left circle is labeled 'Science' and the right circle is labeled 'Computer science'. The overlapping area in the center is labeled 'Code a simulation of a science phenomenon'. A red arrow points from the text 'What is the value, if any, of coding a simulation of a science phenomenon for students' learning?' to the intersection area.

What is the value, if any, of *coding a simulation of a science phenomenon* for students' learning?

Does it resemble what scientists do in the real world in any way?

Homework

- Read Merrill (2017).
- Read one of the lessons provided that involve using block-based coding in scientific modeling instruction.
- Locate one Scratch simulation pertinent to their future teaching and play with it.
- Complete the reflection activity.

Note: submit reflection activity to Dr. Capps by Sunday, September 16 at 5pm.

APPENDIX E

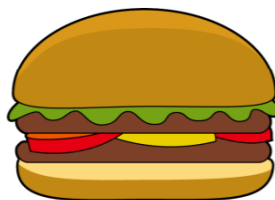
CODING WORKSHOP ACTIVITY: KNOCK KNOCK JOKE

Strategies to avoid coding errors:

- Plan the knock knock joke before you start coding.
- Name the broadcast block specifically so you'll remember where it falls in the sequence.

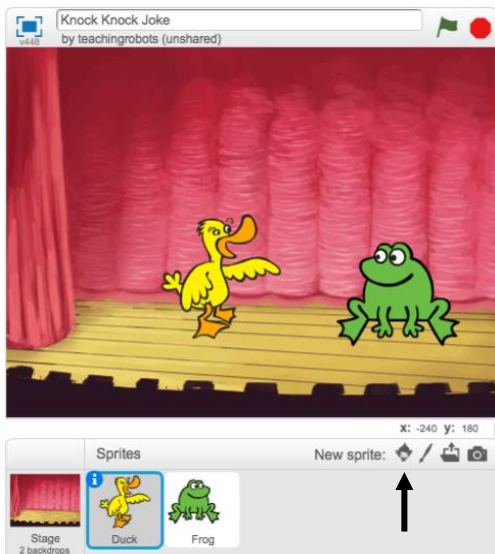
How to identify and fix coding errors:

- If the action takes place out of sequence, check if the broadcast block is out of order.
- Follow a broadcast: find the sprite that initially sends a broadcast, then look for the sprite that receives it, check your scripts to see if the broadcast and receive are initiating the correct actions.
- Broadcast Hamburger: – Broadcast scripts that start with an Event or Control hat block and end with sending another broadcast look like a hamburger: a top bun (the hat block), stuff in the middle (scripts), and a bottom bun (the broadcast new message). Checking to see if you code has a “top bun” and a “bottom bun” is a quick way to identify bugs in the code. Note: the final script that starts with a “When I receive...” block won't need a “bottom bun”/new broadcast block.

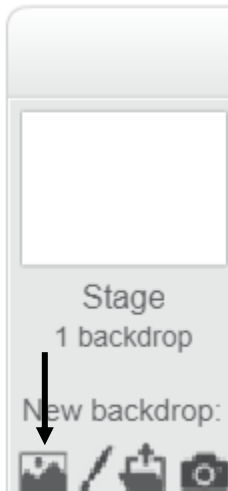
**Coding Instructions**

Follow the steps to code a knock knock joke. You can choose other sprites if you would like.

1. Select two sprites (e.g., a duck and a frog) from the library (black arrow). If you don't want to use the cat, you can right click on the sprite and delete it.



2. Select a backdrop from the library (see arrow).



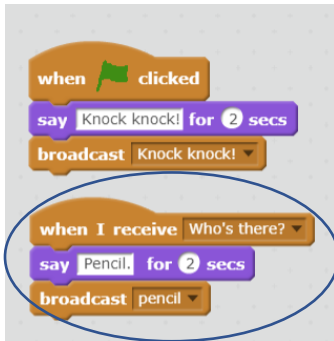
3. Map out the knock knock joke to determine which character says what.
4. Program the first sprite (duck) to start the joke when the Green Flag is clicked. End with the first broadcast.



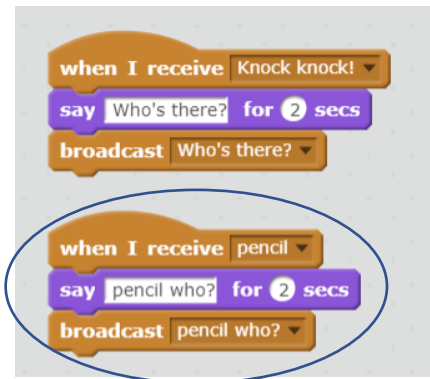
5. Program the second sprite (frog) to receive the first broadcast, then respond with the expected “Who’s there?” followed immediately with a broadcast of “Who’s there?”.



6. Write a new script for duck that starts with “When I receive “Who’s There?”...” (see circle).



7. Starting to get the idea? Now Frog needs to Receive “Pencil” and ask “A pencil who?”:



8. Now duck can come in with the punch line:



APPENDIX F

EXAMPLE OF SCIENTIFIC MODELING LESSON THAT USES CODING

Module 2: Water as a Shared Resource



2

Lesson 3

Adding More Water Pumps and Running Experiments

50 minutes (1 day)

Lesson Overview (New Learning and Exploration)

In this lesson, the students will modify the base Water Pumping model to include additional water pumps. In the first activity, the students will add a second water pump that pulls water from the aquifer. Next, students will add monitors and a line graph that collects and displays the cumulative amount of water pumped by each pump. In the second activity, the new model can then be used as an experimental test bed. Students develop a hypothesis, run an experiment, and analyze the results to see what effect the modification had on the system.

Teaching Summary**Getting Started** – 5 minutes

1. Review of the previous day's lesson and concepts and connection to today's lesson.

Activity #1: Adding a Water Pump – 20 minutes

2. CS review: find and decode the procedure that creates the initial pump.
3. Duplicate and alter the procedure to create a new pump.
4. Add monitors and line graphs to display and visualize data.
5. Test your model.

Activity #2: Running an Experiment – 20 minutes

6. Designing your experiment.
7. Running your experiments.
8. Collecting and analyzing data.

Wrap-Up – 5 minutes

9. What does the computer model enable us to do that would be difficult in the real world?
10. How could a computer model like the Water Pumping model be used to manage water resources?



Lesson Objectives

The student will:

- ✓ Learn that typically as human populations and consumption of natural resources increase, so do the negative impacts on Earth [LO11].
- ✓ Ask a question that can be answered using the model as an experimental test bed [LO12]. Design and conduct an experiment [LO13].
- ✓ Collect and analyze data to look for patterns [LO14].
- ✓ Modify a simple computer model and display output data using widgets [LO15].
- ✓ Practice Pair Programming and Iterative Design-Implement-Test cycle [LO16].

Teaching Guide

Materials, Resources and Preparation

For the Students

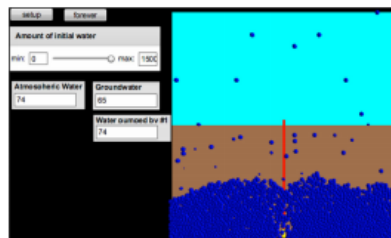
- Computers
- Water Pumping StarLogo Nova base model
- New commands and concepts sheet [student handout]
- Model Design Form document [student handout]
- Experimental Design Form document [student handout]

For the Teacher

- Computer and projector
- Water Pumping StarLogo Nova models: base model, base model plus new pumps
- Water Resources background videos [for reference]
- Guided Introduction to StarLogo Nova document [for reference]
- StarLogo Nova Blocks CS Concepts guide document [for reference]
- StarLogo Nova Blocks Reference Guide [for reference]
- Slide presentation with simple commands

Getting Started - 5 mins

1. **Review of the previous lesson and make connection to today's lesson** – 5 mins
 - Last time we learned about the base model, the abstractions included, and the mechanisms that are executed to make the simulation run. Today we are going to add another pump to the model, then add output widgets so we can assess the impact of the new pump when we run experiments. What do you predict will happen when we add a new pump? [\[Practice 1: Asking questions and defining problems\]](#) [\[Practice 2: Developing and using models\]](#) [\(CCC: Cause and Effect\)](#)
 - Review concepts of infiltration and aquifers.



Activity #1: Adding a Water Pump - 20 mins

We'll be adding a new pump that pulls water from the aquifer. Ask the students to review what we know about how the first pump was created. Remember to remix the project before making any changes.

2. CS Review: code and concepts useful for the modification

- Use Think-Pair-Share to have students discuss the existing code and report out.



- [Teacher notes:] Remember that in this model we are using a 2D view of Spaceland, rather than the 3D view.
- In the "makePump" procedure a red turtle is created at (0, 3) and is set to head towards the bottom edge of Spaceland.
- Then the turtle takes 49 steps forward while stamping the grid beneath it red at each step.
- Then the turtle sets its color to yellow and continues 5 more steps forward while stamping the grid beneath it yellow at each step.
- Finally, the turtle is no longer needed so we delete it.

3. Duplicate and alter the procedure to create a new pump

- Ask students for suggestions on how to make a new pump.
- Suggest that we start by making a copy of the existing code for creating a water pump! (n.b. this is a perfect opportunity to talk about remixing on a procedural level.)
- Demonstrate how to use the rectangular lasso to select, copy, and paste a whole code block.
- Give the students the challenge of repositioning the second pump at a distance from the first pump.
- Have the students show their neighbors their solutions to this challenge.
- Next, if time allows, tell the students that we will want to be able to distinguish the number of water particles drawn up by each pump so we will need to be able to tell whether a water molecule is pulled up by one pump or another.
- Have students brainstorm and attempt a solution to this challenge.

4. Add monitors and line graphs to display and visualize data

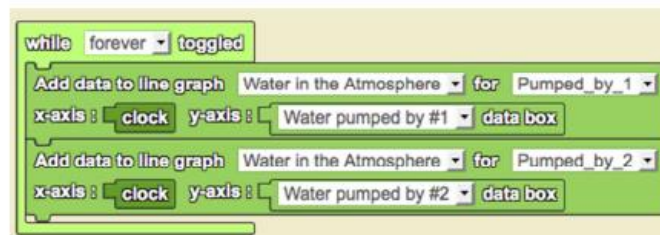
- Demonstrate to the students how to use the Edit Widgets tool to add two output data boxes and a line graph (or, alternatively, have a student who knows this technique demonstrate it).
- [Notes for the teacher:] Let's add some instrumentation so we can detect how fast the water is being pumped from each pump head.
- We will start with by adding two monitors, one for the water pumped by the first pump head and one from the new pump head.



- Click the "Edit Widgets" tool in the Spaceland window then click on "Create Widget."



- Select "Data Box" and then name the widget "Water pumped by #1" and click "Add Widget." Reposition the widget where it is clearly visible and does not overlap any existing user interface element. Do the same steps to create a widget called "Water pumped by #2."
- Note: these data boxes can now be used as global variables. The value held in the data box can be updated by any agent.
- Next, add the code that will initialize the values of these data boxes, then increment (or increase) the value anytime an agent interacts with the pump.
- Where do we initialize the values? [in the setup]
- Where do we increment the values? [in the "pump" procedure]
- In order to collect and visualize quantitative data we need to add a line graph in StarLogo Nova. With this information we will be able to compare patterns in the collected data.
- For this model, what products do we want to monitor? [We'd like the graph to collect data on the time elapsed since the model started running and the cumulative number of water molecules pumped by each pump over time.]
- Let's create a new line graph called "Water Pumped over Time."
- Demonstrate how to create a line graph in StarLogo Nova using Edit Widgets. Drag the line graph off to the side of Spaceland. Add new series to the graph by double clicking on New Series and changing the name and line color.
- For example,
 - Create a new series called "Pumped_by_1" then select red as its line color.
 - Create a new series called "Pumped_by_2" then select black as its line color.
- Finally, click "Edit Widgets" to leave editing mode and return to play mode.
- Next, we want "The World" to update the line graph each time through the forever loop, so we need to add a "while forever toggled" loop on the page labeled "The World."



- Notice that we need the "clock" along the x-axis and the cumulative number of water pumped on the y-axis. Where can we get a count of water agents pumped? [The value is held in the "Water pumped by #1" data box already, so use it.]



- Add in similar “Add data to line graph” command blocks to the “while forever toggled” loop for each of the other products you would like to monitor in the line graph.

5. Test your model

- Test your model: Click the “setup” button. Did the value in the “Water pumped by #1” and “Water pumped by #2” data boxes get reset to zero? Click on “forever.” Does the model behave as expected? Is the line graph displaying data? Are the water molecules getting sucked up by the pumps? [\(Practice 3: Planning and carrying out investigations\)](#)

Teaching Tip *Showing students how to lasso around a block of code then copy and paste that code into a new agent page or the same page can speed up their development time.*

Teaching Tip *This lesson can be scaffolded based on students' learning abilities by adding or removing the Optional sections. More advanced classes can experiment with additional modifications.*

Additional modifications: (optional)

- Change the pump depth
- Change the pump head surface area
- Add even more pumps

Activity #2: Running an Experiment - 20 mins

In this activity students will run an experiment using the model they have modified by adding another pump. Students will have freedom to design their own experiments and there are many options, from simple to more complex experiments (particularly if students have added in sliders).

6. Designing your experiment

- **Experimental Design**
Use the “Experimental Design” form as a guide and guide students as they develop a scientific question in pairs. Emphasize the need to run multiple trials at each setting and to clearly identify the variables, as well as the difference between a question and a *testable* question. [\(Practice 1: Asking questions and defining problems\)](#) [\(Practice 3: Planning and carrying out investigations\)](#) [\(Practice 5: Using Mathematical and computational thinking\)](#)
- Run your experiment in pairs so the question can be answered. Which variable will you be changing? What range? How many trials will be conducted at each setting? This information should be written into your template documents before beginning.
- **Collecting and analyzing data.**
Using the instrumentation in the model (the graph and the data boxes) to monitor the cumulative number of water molecules pumped over time under the different conditions you are testing. Record the data. Look for patterns in your data [draw a graph and/or make a table, record observations].



**PROJECT
GUTS**
Growing Up Thinking Scientifically

Module 2: Water as a Shared Resource

7. Running your experiment

- Example simple experiment: Run the experiment for 2400 ticks with a second pump located with $x = 25$. Hit forever to pause it every 100 ticks. Write down the count of water molecules pumped from each of the data boxes. Repeat the process until you reach approximately 2400 ticks. Then, clear everything and repeat the whole experiment as many times as you think you should. Compare the amounts pumped to the results from the same experiment, when run with just one pump.

Note: This experiment is just an example among many possible experiments.

8. Collecting and analyzing data.

- Graph your data points. Do you notice any trend? Did the amount of water pumped increase, decrease or stay the same over time with the modification added? What can you say now about your testable idea?
- Share out your experiment and results with the class.
- Discuss the difference between **correlation** and **causation**.

Teaching Tip *The experimental design can be tailored to students' abilities.*

Wrap-Up - 5 mins

8. What does the computer model enable us to do that would be difficult to do in the real world?
9. How could a computer model like this one be used to manage water resources?

Assessment Questions

- Describe potential negative impacts of adding additional water wells in a community with limited water resources [LO11].
- Assess student responses on the Model Design Form and Experimental Design Form [LO12, LO13, and LO14].
- Describe a procedure you added to the model [LO15].
- In your own words, describe how you tested and, if necessary, refined your procedure [LO16].

Standards Addressed

NGSS Performance Expectations

Earth and Human Activity

MS-ESS3-4. Construct an argument supported by evidence for how increases in human population and per-capita consumption of natural resources impact Earth's systems.

Earth's Systems

MS-ESS2-4. Develop a model to describe the cycling of water through Earth's systems driven by energy from the sun and the force of gravity.

NRC Disciplinary Core Ideas

ESS2.C: The Roles of Water in Earth's Surface Processes

Water continually cycles among land, ocean, and atmosphere via transpiration, evaporation, condensation and crystallization, and precipitation, as well as downhill flows on land. Global movements of water and its changes in form are propelled by sunlight and gravity.

ESS3.C. Human Impacts on Earth Systems

Typically as human populations and per-capita consumption of natural resources increase, so do the negative impacts on Earth unless the activities and technologies involved are engineered otherwise.



NRC Scientific and Engineering Practices

Practice 1. Asking questions and defining problems

Ask questions to identify and clarify evidence of an argument.

Practice 2. Developing and using models

Evaluate limitations of a model for a proposed object or tool.

Develop and/or use a model to predict and/or describe phenomena.

Develop and/or use a model to generate data to test ideas about phenomena in natural or designed systems, including those representing inputs and outputs, and those at unobservable scales.

Practice 3. Planning and carrying out investigations

Plan an investigation individually and collaboratively, and in the design: identify independent and dependent variables and controls, what tools are needed to do the gathering, how measurements will be recorded, and how many data are needed to support a claim.

Collect data to produce data to serve as the basis for evidence to answer scientific questions or test.

Practice 4. Analyzing and interpreting data

Analyze and interpret data to provide evidence for phenomena.

Practice 7. Engaging in argument from evidence

Construct an oral and written argument supported by empirical evidence and scientific reasoning to support or refute an explanation or a model for a phenomenon or a solution to a problem.

Practice 8. Obtaining, evaluating and communicating information

Communicate scientific and/or technical information (e.g. about a proposed object, tool, process, system) in writing and/or through oral presentations.

NRC Crosscutting Concepts

Cause and Effect

Cause and effect relationships may be used to predict phenomena in natural or designed systems.

Stability and Change

Stability might be disturbed either by sudden events or gradual changes that accumulate over time.

Energy and Matter

The transfer of energy can be tracked as energy flows through a designed or natural system.

Patterns

Patterns can be used to identify cause and effect relationships.

Graphs, charts, and images can be used to identify patterns in data.

Scale, proportion and quantity

Time, space, and energy phenomena can be observed at various scales using models to study systems that are too large or too small.

Systems and Systems models

Systems may interact with other systems; they may have sub-systems and be a part of larger complex systems.

Models can be used to represent systems and their interactions—such as inputs, processes and outputs—and energy, matter, and information flows within systems.

Models are limited in that they only represent certain aspects of the system under study.

CSTA K-12 Computer Science Standards

CT	Abstraction	2-12	Use abstraction to decompose a problem into sub problems.
CT	Abstraction	3A-9	Discuss the value of abstraction to manage problem complexity.
CT	Abstraction	3B-10	Decompose a problem by defining new functions and classes.
CT	Algorithms	2-4	Evaluate ways that different algorithms may be used to solve the same problem.
CT	Modeling & simulation	1:6-4	Describe how a simulation can be used to solve a problem.



**PROJECT
GUTS**
Growing Up Thinking Scientifically

Module 2: Water as a Shared Resource

CT	Modeling & simulation	2-11	Analyze the degree to which a computer model accurately represents the real world.
CT	Modeling & simulation	2-9	Interact with content-specific models and simulations to support learning and research.
CT	Modeling & simulation	3A-8	Use modeling and simulation to represent and understand natural phenomena.
CT	Modeling & simulation	3B-8	Use models and simulation to help formulate, refine, and test scientific hypotheses.
CT	Modeling & simulation	3B-9	Analyze data and identify patterns through modeling and simulation.
CPP	Programming	2-5	Implement a problem solution in a programming environment using looping behavior, conditional statements, logic, expressions, variables and functions.
CPP	Programming	3A-3	Use various debugging and testing methods to ensure program correctness.
CPP	Programming	3A-4	Apply analysis, design and implementation techniques to solve problems.

APPENDIX G
REFLECTION ACTIVITY

Name: _____

Use the *Explore* feature of Scratch to search for existing simulations that focus on a topic that is pertinent to your science teaching area. You may browse through a few simulations and then *choose one* for this activity. Explore simulation features, play with it, analyze its sprites, and review the coding blocks used to simulate specific science concepts or features. Subsequently, answer these questions:

1. Provide the link to the simulation you chose.
2. Which science systems, concepts or phenomena are represented in the simulation? (e.g., solar system)
3. Which specific aspects of its referent in the real world are represented in the simulation? (e.g., planet rotation around the sun).
4. How did the simulation creator try to accurately represent the target phenomenon? (e.g., used colors, size, proportions, etc.)
5. Which coding blocks are used to materialize dynamic science aspects and how? (e.g., block *forever* controls continuous rotation of planets and stars around the sun).
6. If you were to improve such a simulation for your teaching, how would you improve it?
7. Based on your analysis of the scientific modeling lesson (see Google Drive files):
 - a. What do students use the code-based simulated models for?
 - b. Which scientific modeling practices are students expected to engage in?
 - c. In your opinion, how can code-based simulations support students' scientific modeling and why?
8. Suppose you need to help your students develop scientific models of a certain science phenomenon.
 - a. Would you provide the simulation to your students or help them create science simulations from scratch? Why? Your answer should have at least 100 words.
 - b. Provide examples of scientific modeling activities that your students would perform while using the simulation. Your answer should have at least 100 words.

APPENDIX H

SLIDES FOR FACE-TO-FACE CS-MODEL ACTIVITIES



Coding in scientific modeling

Lucas Vasconcelos

Dr. ChanMin Kim (advisor)

Dr. Daniel Capps (course instructor)

Reflection activity

- Overall positive reaction to Scratch and coding
- Very creative ideas for teaching with coding
- Several benefits of using coding for scientific modeling instruction
- Critical thinking
- Testing/assessing students' knowledge
- Student-centered pedagogy

Anchoring problem

- **Scenario:** if you were to run out of drinkable water at a campground, what would be the next thing to do?
- **Criteria/constraints:** there is a nearby river. You have these materials: plastic bottles, coffee filters, cotton, gravel, and different activated charcoal grains.
- **Rule:** You need to create a filter that is effective in removing impurities from river water. What would you need to know about it?



Before we start...

- Which ideas are important to create a filter?
- Which variables could be tested?
- What are the independent and dependent variables?



Independent variables	Dependent variables
Cotton	Water quality (color)
Gravel	Input-output ratio
Activated charcoal sizes	Water filtration time

Before we start...

- What are examples of research questions that could guide experiments with a water filter?

Example:

How do different materials affect water quality, input-output ratio, and filtration time in a water filter experiment? Why does that happen?

- Today we will create, test, evaluate, and revise models related to a water filter experiment.
- What is the easiest way to start testing water filter designs?

Teams

- Jonathan and Weston
- Maggie and Grace
- Shea and Rosemary
- Catherine Nord and Chandler
- Felexia and James
- Randall and Christopher
- Emily Huffman and Johnisha
- Catherine Riley and Rachel
- Emily Jennings and Alyssa
- Hayley and Moriah
- Mohammed and Callie

Creating hypotheses

As a team, you will fill out the **Experiment Hypotheses Sheet** to create hypotheses/predict results of experiments 1, 2, 3, and 4. Example:

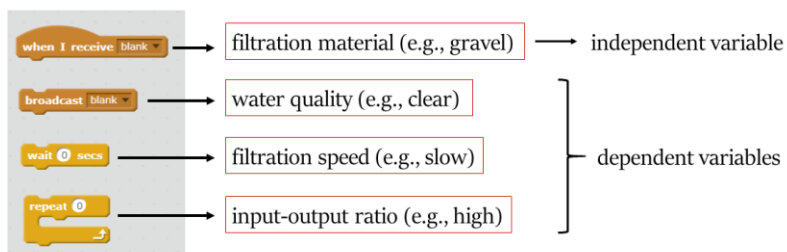
Cotton Cotton yields brackish (dark, brackish, clear) water color
 Cotton yields slow (slow, average, fast) water filtration
 Cotton yields average (high, low, average) input-output ratio

For experiments 3 and 4: create a design that you think is effective and you would like to test in the science lab. The design needs to have at least two different materials.

Example: charcoal (bottom), cotton (middle), charcoal (top).

Coding simulations

Code simulations of water filter experiments. Your team will code the blocks:





[How to code](#)

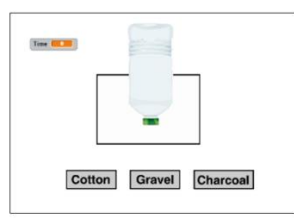
Debugging sheet

- Review common coding errors
- Review debugging strategies to identify and fix errors
- Answer question #1: a reflection prompt on experiment variables
- Document errors and debugging process using table. This helps you avoid repeated effort.
- **Note:** the teammate who is **not** using the computer will write down the errors, hypotheses, attempted solutions, etc.

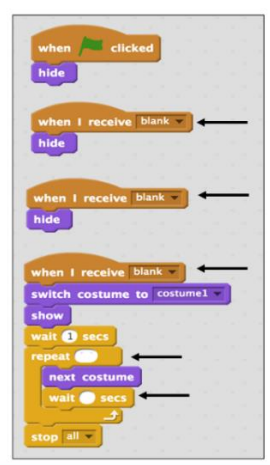
Error	Hypothesis	Executed solution	Did solution work?	Cause of the problem
The water flow with gravel is fast but simulation was too slow.	The block wait_secs needs a smaller number.	I changed the number in the wait_secs block until the speed is slow.	Yes, it worked after a few attempts to find the best number.	The wait_secs block needs a smaller number.

Coding activity 1: cotton, gravel, and charcoal



1. Go to <https://scratch.mit.edu/>
2. Create one account for the team
3. Go to <https://tinyurl.com/y8q9bd8w>
4. Click  and then 
5. Reminder: Code **material** and the **water sprites**
6. Keep the tab open when you are done.

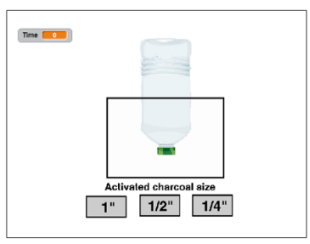


Reminder: Code **material** and the **water sprites**





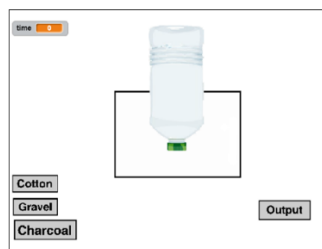
Coding activity 2: activated charcoal grains

1. Open a new tab
2. Go to <https://tinyurl.com/yb4j7e7d>
3. Click  and then 
4. Reminder: Code **material** and the **water sprites**
5. Keep the tab open when you are done.



Coding activity 3: combining materials

1. Open a new tab
2. Go to <https://tinyurl.com/ybntm2aa>
3. Click  and then 
4. Reminder: Code **material** and the **water sprites**
5. Keep the tab open when you are done.



Science lab: rooms 215 and 216

It is time to:

- Test your models and hypotheses
- Collect empirical data on the phenomenon
- Evaluate your models and hypotheses
- Note: bring your **hypotheses-results sheet** with you.

Each team will collect and document data using the **experiment handout**.

- Take pictures of water quality before and after the experiment.
- Record the input-output ratio before and after the experiment.
- Time the filtration process from beginning to end.

Please do not turn off or log out of computers.

Science lab

Prior to experiment:

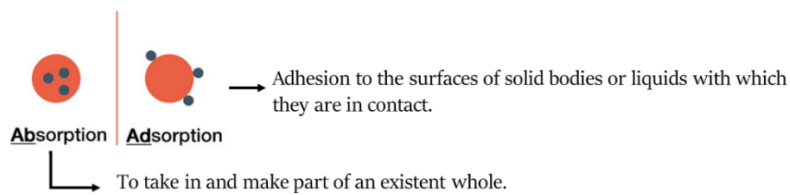
- Which material yields clear, brackish, and dark water?
- How about fast, slow, and average filtration time?
- How about low, high, and average input-output ratio?

After experiment:

- Complete the **results** section on the **hypotheses sheet**.
- Compare the empirical data with your original hypotheses.

Explaining / evaluating the results

- Does the collected data confirm or contradicts your initial hypotheses?
- What are the underlying science concepts or ideas that explain the performance of the different filtration materials?
- Do you need to revise your models to accommodate new information?



Experiments 3 and 4

- It is time to test the experiments 3 and 4 you designed.
- Your stations should have all materials to combine materials and test the experiment.

Revising simulations

- Now we will return to the computer lab to adjust the simulations based on your most current understanding of how each material performs in the water filter experiment.
- Access your Scratch account and revise the three simulations.

Discussion

- What was the process of coding a simulation of a science phenomenon?
- In your experience, what is the value of coding for the creation and development of students' models, if any?
- Is there any value in pair coding / conducting experiments in pairs?
- Did you experience any challenges during today's activities?
- As a future teacher, how likely are you to use coding in your lessons?

Homework

- Use [this link](#) to design a lesson to use block-based coding simulations in scientific modeling instruction. The topic should be pertinent to your area of emphasis.
- Write essay to describe how coding can simulate features of scientific models about the water filter experiment.

Note: submit assignments to Dr. Capps by Sunday, September 23 at 5pm.

APPENDIX I

HYPOTHESES SHEET FOR CODING WATER FILTER SIMULATION

Team: _____

Experiment Hypotheses

Experiment 1: cotton, gravel, and activated charcoal	Experiment 2: small, medium, and big activated charcoal
<p>Cotton yields _____ (dark, brackish, clear) water color Cotton yields _____ (slow, average, fast) filtration time Cotton yields _____ (low, average, high) input-output ratio</p> <p>Gravel yields _____ (dark, brackish, clear) water color Gravel yields _____ (slow, average, fast) filtration time Gravel yields _____ (low, average, high) input-output ratio</p> <p>Activated charcoal yields _____ (dark, brackish, clear) water color Activated charcoal yields _____ (slow, average, fast) filtration time Activated charcoal yields _____ (low, average, high) input-output ratio</p>	<p>Big activated charcoal yields _____ (dark, brackish, clear) water color Big activated charcoal yields _____ (slow, average, fast) filtration time Big activated charcoal yields _____ (low, average, high) input-output ratio</p> <p>Medium activated charcoal yields _____ (dark, brackish, clear) water color Medium activated charcoal yields _____ (slow, average, fast) filtration time Medium activated charcoal yields _____ (low, average, high) input-output ratio</p> <p>Small charcoal yields _____ (dark, brackish, clear) water color Small charcoal yields _____ (slow, average, fast) filtration time Small charcoal yields _____ (low, average, high) input-output ratio</p>

Instructions for experiments 3 and 4: you will test two filter model designs for experiments 3 and 4. Each design should have at least two filtration materials. Write down below your hypotheses and results from each experiment.

For example: design 1: gravel (bottom) and charcoal (top) | design 2: charcoal (bottom), cotton (middle), charcoal (top).

Experiment 3: combined media	Experiment 4: combined media
_____ yields:	_____ yields:
_____ (dark, brackish, clear) water color	_____ (dark, brackish, clear) water color
_____ (slow, average, fast) filtration time	_____ (slow, average, fast) filtration time
_____ (low, average, high) input-output ratio	_____ (low, average, high) input-output ratio

Experiment Results

Experiment 1: cotton, gravel, and activated charcoal	Experiment 2: small, medium, and big activated charcoal
Cotton yields _____ (dark, brackish, clear) water color	Big activated charcoal yields _____ (dark, brackish, clear) water color
Cotton yields _____ (slow, average, fast) filtration time	Big activated charcoal yields _____ (slow, average, fast) filtration time
Cotton yields _____ (low, average, high) input-output ratio	Big activated charcoal yields _____ (low, average, high) input-output ratio
Gravel yields _____ (dark, brackish, clear) water color	Medium activated charcoal yields _____ (dark, brackish, clear) water color
Gravel yields _____ (slow, average, fast) filtration time	Medium activated charcoal yields _____ (slow, average, fast) filtration time
Gravel yields _____ (low, average, high) input-output ratio	Medium activated charcoal yields _____ (low, average, high) input-output ratio
Activated charcoal yields _____ (dark, brackish, clear) water color	Small charcoal yields _____ (dark, brackish, clear) water color
Activated charcoal yields _____ (slow, average, fast) filtration time	Small charcoal yields _____ (slow, average, fast) filtration time
Activated charcoal yields _____ (low, average, high) input-output ratio	Small charcoal yields _____ (low, average, high) input-output ratio

Experiment Results

Experiment 3: combined media	Experiment 4:
_____ yields:	_____ yields:
_____ (dark, brackish, clear) water color	_____ (dark, brackish, clear) water color
_____ (slow, average, fast) filtration time	_____ (slow, average, fast) filtration time
_____ (low, average, high) input-output ratio	_____ (low, average, high) input-output ratio

APPENDIX J

DEBUGGING SHEET

Common coding mistakes

- Forgetting to either code when I receive or broadcast when creating independent-dependent variable relationships.
- Not naming the block “broadcast” appropriately.
- Leave block parameters blank or with zero.
- Edit the wrong block. For example, coding the wrong wait__secs block from a sequence that may contain a few of those.
- Accidentally deleting a sprite (right click).
- Accidentally hiding a sprite (right click).

Debugging strategies to identify and fix coding errors

- Review the simulation output (visual feedback)
- Create hypotheses for why the simulation does not work as expected.
- Review code in specific sprites that show error.
- Discuss hypotheses and ideas for solution with your teammate.
- Document hypotheses, solutions, and whether solutions worked to avoid repeated effort.
- Create a rationale to inform your actions. Avoid random trial and error.
- Compare sprites that have similar or identical code.

Debugging cases

Answer the question and complete the table below as accurately as possible *while* coding simulations. Use of science- and coding-related terminology.

Simulation 1: cotton, gravel, and activated charcoal.

1. Which independent-dependent variable relationships do you want to simulate?

2. Complete the table below every time the simulation does not work as expected.

Error	Hypothesis	Executed solution	Did solution work?	Cause of the problem
The water flow with gravel is fast but simulation was too slow.	The block wait_secs needs a smaller number.	I changed the number in the wait_secs block until the speed is slow.	Yes, it worked after a few attempts to find the best number.	The wait_secs block needs a smaller number.

Simulation 2: small, medium, and big activated charcoal grains

1. Which independent-dependent variable relationships do you want to simulate?
2. Complete the table below every time the code does not work as expected.

Error	Hypothesis	Executed solution	Did solution work?	Cause of the problem
The water flow with gravel is fast but simulation was too slow.	The block wait_secs needs a smaller number.	I changed the number in the wait_secs block until the speed is slow.	Yes, it worked after a few attempts to find the best number.	The wait_secs block needs a smaller number.

Simulation 3: combined media

1. Which independent-dependent variable relationships do you want to simulate?
2. Complete the table below every time the code does not work as expected.

Error	Hypothesis	Executed solution	Did solution work?	Cause of the problem
The water flow with gravel is fast but simulation was too slow.	The block wait_secs needs a smaller number.	I changed the number in the wait_secs block until the speed is slow.	Yes, it worked after a few attempts to find the best number.	The wait_secs block needs a smaller number.

APPENDIX K
EXPERIMENTAL DATA SHEET

Team: _____

Experiment 1: cotton, gravel, and activated charcoal.

Material	Quality (color)	Input-output ratio (ml)	Time (secs)

Experiment 2: small, medium, and big activated charcoal.

Material	Quality (color)	Input-output ratio (ml)	Time (secs)

Experiment 3: combined media (at least two types of materials).

Materials	Quality (color)	Input-output ratio (ml)	Time (secs)

Experiment 4: combined media (at least two types of materials).

Materials	Quality (color)	Input-output ratio (ml)	Time (secs)

APPENDIX L

HOMEWORK INSTRUCTIONS

- Lesson Design

You will use [this link](#) to design a lesson in which you plan to teach scientific modeling using block-based coding. The lesson should focus on a topic that is relevant to your intended context of instruction (e.g., life science, physics, chemistry, etc.)

You can design the lesson based on:

- (1) An existing Scratch simulation that can be used or adapted for your lesson **or**
- (2) A hypothetical Scratch simulation that does not exist but could be envisioned for this lessons.

If you choose option (1), use the Explore feature in Scratch to find a simulation. Even if you do not find a great simulation on the topic you want to teach, you can still use a simulation that you could further develop or modify for your lesson.

Note 1: you can design the lesson to use simulation you found for the reflection activity if you wish to use it again.

Note 2: This is an **individual** homework assignment.

- Essay

Write a short essay in which you discuss your experience coding a water filter simulation in class. Use the questions below to guide your essay but feel free to add more relevant information.

- (a) How is coding used to materialize science concepts/phenomena from the water filter experiment?
- (b) How does coding a simulation contributed to your creation/development of a model of the target science concepts?

Please use specific terminology learned in class about the science phenomenon (e.g., types of variables, materials) and the coding blocks you used. The essay should be *at least* 200 words long.

APPENDIX M

ONE PAGE OF RESEARCHERS' DATA ANALYSIS JOURNAL

Regarding Research Question 2:

Researchers' assumptions:

- Participants may develop understanding of computer science concepts after intervention.
- Most participants may fail to define and apply *conditionals* in pre-interviews. A few of them may define and apply delays and loops (intermediate level).

Data analysis:

- To score participants' understanding of conditionals: they need to correctly define both broadcast and when I receive (input and output) to get a full score.
- Application of code: To get full score, participants need to articulate an explanation verbally and correctly use the block to solve the challenge. If they fail at doing both, they get half score (1).

To conduct statistical tests:

1. All variables are normally distributed. Can I use paired-samples t-tests?
 - a. After talking to advisor and fellow graduate student: use nonparametric tests due to small sample size. Wilcoxon Signed-Rank tests seem suitable.
2. Only 7 participants have complete data set. Three are missing one score (code application) due to university wireless outage. Can the Expectation Maximization Algorithm be used to automatically estimate those missing values?
 - a. Based on conversation with advisor, replacing 30% of data set with EMA might not be wise. Just exclude the three participants from this analysis.
3. Two participants did not attend post-interviews. Should I use the EM algorithm to replace their three missing scores?
 - a. No. Given the small sample size, I should refrain from using EMA.
4. Seven participants neither attended pre-interviews nor post-interviews
 - a. It is not possible to draw conclusions on changes in their scores based on comparisons before and after CS-Model.

To conduct qualitative analysis:

1. Participants' responses to pre- and post-interviews should be compared against the rubric and the target accurate definition of each computer science concept.
2. Impressions about preliminary interview qualitative analysis: participants have a superficial understanding of concepts. They seem to have misconceptions. It would be interesting to identify misconceptions and check if they were corrected after CS-Model.
3. Open coding should be used to identify misconceptions. Research on literature does not show an existing coding scheme on misconceptions for those computer science concepts.
4. Be mindful that that three participants attended the CS-Model pilot study. Results about them should be discussed with caution.

APPENDIX N

SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 1

The table below presents analysis of Regina's interview responses about models and modeling.

Item	Pre-interview		Post-interview	
	Quote	Level	Quote	Level
Nature of models	“[A model] is a mental representation, it's a representation that helps you understand something else. Usually something that's hard to conceptualize.”	Level 2	“A model is a concept that you have in your head of how something works or how a phenomenon occurs in science, and it's something that you kind of create out of that to express that idea, that concept.”	Level 3
Multiple models	“In biology sometimes when you're thinking about cells... sometimes you're thinking about this analogy. Sometimes the analogies are also the same thing, but you can use different analogies to describe cells. Um, well over history there's also been atomic models that are various levels of accuracy, but it kind of helps people visualize what an atom is depending on how accurate you need to be. You can choose more accurate models.”	Level 1	“If you're trying to show how the moon goes around the earth, you could try to show the phases of the moon, there's a variety of ways to kind of depict that depending on what you're interested in knowing about the phases so you can look directly down at them and see how they're interacting with the sun, where you can kind of see what plane they're on relative to the earth.”	Level 2
Purpose of models	“[Models] are for better understanding something that is difficult to understand otherwise.”	Level 1	“Depending on what you're, you're interest in, knowing about the phases so you can look directly down at them and see how they're interacting with the sun, where you can kind of see what plane they're on relative to the earth.”	Level 3
Changing models	“[Scientists change models] because they have evidence that their current understanding is flawed in some way.”	Level 2	“If there is something in the real world, if there's something that they [scientists] observe that goes against what their model would have predicted than they have to alter their model to fit that.”	Level 3

APPENDIX O

SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 2

The table below presents analysis of William’s interview responses about computer science concepts

Pre-interview			Post-interview	
Item	Quote	Score level	Quote	Score level
Conditionals	When I receive: “You're giving a command to the figure. And then I see you have meow there. So. It's saying when I receive this. I'm going to do this action pretty much or maybe when I receive this distance from me I don't know.” Broadcast: “That's portraying it onto the screen.”	Terminology: 0 Definition: 0 Application: 0	“It's signaling, you can think of it kind of like a conversation, like one character is going to do an action or say something based on another character broadcasting that and in the first character receiving that. So it's kind of, I would say a way of communication between your sprites.”	Terminology: 2 Definition: 2 Application: 2
Delays	“I mean waiting for the dude, the cartoon character to walk. I mean pausing the animation.”	Terminology: 1 Definition: 1 Application: 2	“It's used for like a pause in time for a certain sprite I would say. So if, if it was saying, wait one second, you would click start and that cat wouldn't start moving or talking or whatever code you put in there for one second.”	Terminology: 2 Definition: 2 Application: 2
Loops	“Play the animation again.”	Terminology: 0 Definition: 0 Application: 1	“It's just going to do that certain task that many times that you put into the box.”	Terminology: 2 Definition: 2 Application: 2

Note. 0 is inexistent, 1 is basic, 2 is advanced.

APPENDIX P

SAMPLE OF DATA ANALYSIS: RESEARCH QUESTION 3

This table presents an overview of thematic analysis administered to Fiona’s and Mary’s interview responses.

Participant	Theme 1		Theme 2	
	Quote	Category	Quote	Category
Fiona	“The student will analyze and apply the building block method to understand the cell cycle.”	Exploration tool	“Each of the models will give the same information. If you or your classmates finds an error in the model that was created, please correct it.”	Practice: Testing and debugging
Mary	“Have them [students] try to create their own food web model of a specific habitat based on their speculated interactions.”	Research tool	“Allow them to go back through their food webs and change the models in response to the research and peer critiques.”	Practice: Reusing and remixing
			“Students should be able to utilize and modify the existing simulation code. The addition of "organisms" into the food web will cause the students to understand how to manipulate and debug different code blocks to properly present the information.”	Practice: testing and debugging
			“Allow them to go back through their food webs and change the models in response to the research and peer critiques.”	Practice: being iterative