Journal of Science Education and Technology (2023) 32:607–628
https://doi.org/10.1007/s10956-023-10049-w

Examining Student Testing and Debugging Within a Computational Systems Modeling Context

Jonathan Bowers[1] · Emanuel Eidin[1] · Lynn Stephens[2] · Linsey Brennan[1]

Accepted: 27 April 2023 / Published online: 22 May 2023

Abstract
Interpreting and creating computational systems models is an important goal of science education. One aspect of computational systems modeling that is supported by modeling, systems thinking, and computational thinking literature is "testing, evaluating, and debugging models." Through testing and debugging, students can identify aspects of their models that either do not match external data or conflict with their conceptual understandings of a phenomenon. This disconnect encourages students to make model revisions, which in turn deepens their conceptual understanding of a phenomenon. Given that many students find testing and debugging challenging, we set out to investigate the various testing and debugging behaviors and behavioral patterns that students use when building and revising computational system models in a supportive learning environment. We designed and implemented a 6-week unit where students constructed and revised a computational systems model of evaporative cooling using SageModeler software. Our results suggest that despite being in a common classroom, the three groups of students in this study all utilized different testing and debugging behavioral patterns. Group 1 focused on using external peer feedback to identify flaws in their model, group 2 used verbal and written discourse to critique their model's structure and suggest structural changes, and group 3 relied on systemic analysis of model output to drive model revisions. These results suggest that multiple aspects of the learning environment are necessary to enable students to take these different approaches to testing and debugging.

Introduction

Science education researchers and policymakers increasingly recognize the importance of involving learners in modeling. From the Next Generation Science Standards (NGSS) in the USA to South Korea's new Korean Science Education Standards (KSES) and Germany's science educational standards (KMK), policymakers have written scientific modeling into their science standards (KMK, 2005a, b, c; National Research Council [NRC], 2012; NGSS Lead States, 2013; Song et al., 2019). While each of these key policy documents has somewhat different viewpoints on using modeling in science classrooms, they, along with many scholars, generally agree that scientific modeling is a process of creating or interpreting a representation of a phenomenon that can be used to explain or predict the behavior of that phenomenon (Harrison & Treagust, 2000; Louca & Zacharia, 2012; Mittelstraß, 2005; Schwarz & White, 2005; Schwarz et al., 2009). There are multiple ways of approaching modeling within science classrooms. Teachers can have students examine and interpret preexisting models, investigating what these models demonstrate about natural phenomena and their inherent limitations (Krell et al., 2015). Students can also construct models of phenomena as sense-making tools and to communicate their ideas to others (Bierema et al., 2017; Passmore et al., 2014; Schwarz et al., 2009). Just as there are multiple approaches to using models, students can construct multiple types of models including mathematical models, diagrammatic models, and computational models (Grosslight et al., 1991; Harrison & Treagust, 2000; Zhang et al., 2014).

Computational modeling uses algorithms or algorithmic thinking to create a model that represents the behavior of a system in a quantitative or semi-quantitative manner

✉ Jonathan Bowers
bowersj8@msu.edu

Emanuel Eidin
eidinema@msu.edu

Lynn Stephens
lstephens@concord.org

Linsey Brennan
mcmull74@msu.edu

1 Michigan State University, East Lansing, MI, USA

2 The Concord Consortium, Concord, MA, USA

(Shin et al., 2021, 2022; Fisher, 2018; Pierson & Clark, 2018; Sengupta et al., 2013; Weintrop et al., 2016). Computational models can be valuable tools for science learning; by combining the visual aspects of diagrammatic models with the mathematical capabilities of mathematical models, computational models are responsive to new data inputs and can be tested and debugged (Shin et al., 2022; Campbell & Oh, 2015; Fisher, 2018; Pierson & Clark, 2018; Sengupta et al., 2013; Sins et al., 2005; Weintrop et al., 2016; Wilensky & Reisman, 2006). While computational modeling programs have existed for decades, their use in K-12 classrooms remains limited. This absence can partially be attributed to the siloed nature of the three main bodies of literature underpinning our conceptualization of computational modeling: modeling, systems thinking (ST), and computational thinking (CT) (Shin et al., 2022). These three cognitive processes are all recognized individually as important goals for science learning, and their intrinsic synergy is a growing interest in the field (Shin et al., 2022; NRC, 2012; Sengupta et al., 2013; Shute et al., 2017; Weintrop et al., 2016).

Within computational modeling, several overlapping practices allow students to utilize ST and CT as they build computational models (Shin et al., 2022). One computational modeling practice that has strong foundations in ST and CT literature is the practice of testing, evaluating, and debugging model behavior (Aho, 2012; Shin et al., 2022; Basu et al., 2016; Grover & Pea, 2018; Stratford et al., 1998; Weintrop et al., 2016; Wilensky & Reisman, 2006). Debugging, in particular, is a practice largely unique to computational contexts as it requires that the model be defined in an algorithmic manner such that its output can be calculated by changing the relative amount of each input variable (Emara et al., 2020; Li et al., 2019; McCauley et al., 2008). By manipulating the relative amount of each input variable, students can test their models to see if they behave according to their understanding and expectations of the phenomena and make changes based on these tests (Shin et al., 2022; Brennan & Resnick, 2012; Hadad et al., 2020; Li et al., 2019; Stratford et al., 1998). Likewise, students can compare their model output to real-world data to further modify and improve their computational models (Shin et al., 2021; Campbell & Oh, 2015; Weintrop et al., 2016; Wilensky & Reisman, 2006). While testing and debugging are an important aspect of computational modeling, students often find it challenging (Li et al., 2019; Sins et al., 2005; Stratford et al., 1998; Swanson et al., 2021; Wilensky & Reisman, 2006). Grapin et al. (2022) and Stratford et al. (1998) suggest that students are reluctant to examine and interpret model output to inform later model revisions.

Given that testing and debugging are both an affordance and a challenge within computational modeling, it is important to investigate how students test and debug as they revise computational models. In this paper, we categorize how students test and debug computational models within a constructivist classroom environment. We are interested in the different testing and debugging behavioral patterns students utilize during the model revision process. By categorizing how student test and debug their models within a constructivist learning environment (centered on project-based learning [PBL] principles), we can hypothesize which aspects of the learning environment best support students in this endeavor. Before summarizing our investigative methods, we review the literature underpinning our conceptualization of constructivism, computational modeling, and the modeling practice of testing and debugging.

Literature Review

Constructivism and Project-Based Learning

For the past several decades, efforts at improving science education have centered on enacting constructivist philosophies and pedagogies in science classrooms (Fosnot, 1996; NRC, 2007, 2012). Constructivism argues that people do not absorb new knowledge in a pure form but instead interpret new information through the lens of prior knowledge, experiences, and social relationships, thereby constructing their own knowledge based on their interactions with the world around them (Fosnot, 1996; Krahenbuhl, 2016; Pass, 2004). Advocates for constructivist approaches in science education push back against transmission-based approaches to teaching and learning, such as the initiate, respond, and evaluate (IRE) model of classroom discourse (Berland & Reiser, 2009; Lemke, 1990; Mehan, 1979). Instead, they endorse classroom environments that allow students to engage in meaningful investigations of real-world phenomena so that they can build a deeper understanding of both science content and scientific practices (Berland et al., 2016; Krajcik & Shin, 2022; NRC, 2012; Windschitl et al., 2020). In the USA, this push for a constructivist approach to science education led to the development and adoption of the Next Generation Science Standards (NGSS), which prioritizes having students engage in authentic science practices, including modeling and computational thinking (NRC, 2012; NGSS, 2013).

Within the broader umbrella of constructivist approaches to science education, there are several frameworks for designing and implementing constructivist lessons in K-12 classrooms, including ambitious science teaching (Windschitl et al., 2020), the 5E instructional model (Duran & Duran, 2004), and project-based learning (PBL) (Krajcik & Shin, 2022;). PBL is a student-centered, constructivist approach to teaching and learning science (Krajcik & Blumenfeld, 2006) that emphasizes collaboration, inquiry, authentic problem-solving, student autonomy, and teacher facilitation. The PBL approach to curriculum design is built around five key principles: centering lesson planning on learning goals that allow students to show mastery of both science ideas and science practices, building student engagement using intriguing phenomena and driving questions, allowing students to explore the driving question and phenomena using authentic scientific practices, tasking students with creating knowledge products (models, explanations, or arguments) that demonstrate student learning, and scaffolding student learning through the use of appropriate learning technologies (Shin et al., 2021; Krajcik & Shin, 2022). This approach has been shown to enhance students' understanding of scientific concepts (Geier et al., 2008; Hmelo-Silver et al., 2007; Karacalli & Korur, 2014; Schneider et al., 2022) and positively impact some affective aspects like self-efficacy and motivation for learning (Fernandes et al., 2014; Schneider et al., 2016; Wurdinger et al., 2007).

Computational Modeling

Computational models are algorithmic representations that allow users to simulate the behavior of a phenomenon under multiple starting conditions (Shin et al., 2021, 2022; Brennan & Resnick, 2012; Fisher, 2018; Pierson & Clark, 2018; Sengupta et al., 2013). Students engage in computational modeling as they construct, test, revise, and evaluate computational models. Computational modeling is rooted in constructionist pedagogies, some of which strongly advocate for computational modeling as a mechanism for science learning (Papert, 1980; Papert & Harel, 1991; Pierson & Clark, 2018; Sengupta et al., 2013). Constructionist pedagogies argue that students learn best when given opportunities to construct and revise knowledge products in ways that promote authentic sense-making (Kafai, 2005; Papert & Harel, 1991; Pierson & Clark, 2018). As computational models provide an environment where students can build and test different ways of representing a phenomenon, computational modeling facilitates sense-making and therefore connects well with constructionism (Farris et al., 2019; Fisher, 2018; Papert, 1980; Pierson & Clark, 2018; Sengupta et al., 2013).

Over the past few decades, the integration of computational modeling in science classrooms has been piloted by many researchers from both systems thinking (ST) and computational thinking (CT) perspectives (Arnold & Wade, 2017; Booth-Sweeney & Sterman, 2007; Brennan & Resnick, 2012; Forrester, 1971; Stratford et al., 1998; Weintrop et al., 2016; Wilensky & Reisman, 2006). Systems thinking approaches the exploration of a phenomenon as a series of interconnected elements that work together to create a system with emergent behavior that is more than the sum of its constituent parts (Arnold & Wade, 2015; Cabrera et al., 2008; Forrester, 1971; Hmelo-Silver & Azevedo, 2006; Meadows, 2008; Riess & Mischo, 2010). ST literature encompasses both agent-based modeling and system dynamics modeling. In the context of system dynamics, this literature tends to focus on how students include key structural elements in their computational models and how they represent a system's behavior over time (Booth-Sweeney & Sterman, 2007; Cronin et al., 2009; Sterman & Sweeney, 2002).

Other researchers often focus on how students use CT as they build and revise computational models (Brennan & Resnick, 2012; Swanson et al., 2021; Weintrop et al., 2016; Wilensky & Reisman, 2006). CT is a form of sense-making that uses an iterative and quantitative approach to decompose a phenomenon or problem to explore, explain, and predict the behavior of that phenomenon or to find a solution to a problem through the creation and revision of algorithms (Shin et al., 2022; Grover & Pea, 2018; Psycharis & Kallia, 2017; Schwarz et al., 2017; Weintrop et al., 2016; Wing, 2006). Because the CT community has its origins in computer science education, CT literature emphasizes the algorithmic nature of computational models, in how students construct and revise their models (Brennan & Resnick, 2012; Weintrop et al., 2016). Additionally, the relationship between computational modeling and computational thinking has been well-established in the fields of mathematics and engineering education (Bakos & Thibault, 2018; Benton et al., 2017; Magana & Silva Coutinho, 2017; Zhang et al., 2020). Zhang et al. (2020) found that engineering students who incorporated the practice of computational thinking within their model construction practices experienced a significant increase in learning outcomes. Similarly, Magana and Silva Coutinho (2017) demonstrated the consensus among engineering experts in academia and industry on the crucial role of preparing future engineers to use computational models in problem-solving. Furthermore, in mathematics education, studies have shown improved learning outcomes as students engage in computational thinking through basic programming (Bakos & Thibault, 2018; Benton et al., 2017; Gleasman & Kim, 2020).

Researchers in both disciplines have, at various times, addressed similar research questions and agree on many of the core components of computational modeling, including the crucial nature of testing and debugging (Shin et al., 2022; Barlas, 1996; Brennan & Resnick, 2012; Sins et al., 2005; Stratford et al., 1998; Swanson et al., 2021; Wilensky & Reisman, 2006). Given this overlap between the ST and CT literature within computational modeling, "A Framework for Computational Systems Modeling" describes how ST and CT are expressed within computational systems modeling and support students in building, testing, and revising computational models (Shin et al., 2022). Within this framework, five computational systems modeling practices build on key aspects of both ST and CT (Shin et al., 2022). While each of these modeling practices represent possible avenues for students to develop ST and CT competencies, it is impractical to develop a singular research instrument to assess all aspects of this framework. Therefore, to conduct a more cohesive study, we chose to specifically focus on the modeling practice of "Test, evaluate, and debug model behavior" as it is a particularly challenging aspect of computational systems modeling for many students (Fig. 1) (Grapin et al., 2022; Li et al., 2019; Sins et al., 2005).

Test, Evaluate, and Debug Model Behavior

Testing, evaluating, and debugging model behavior describes a broad range of strategies found across modeling, system dynamics, and computational thinking literature (Barlas, 1996; Brennan & Resnick, 2012; Campbell & Oh, 2015; Csizmadia et al., 2015; Gilbert, 2004; Li et al., 2019; Sins et al., 2005). Testing and evaluating hypotheses is a core aspect of scientific inquiry (Gilbert, 2004; Lederman, 2013; NRC, 2012). Through this iterative process, scientists revise their understanding of natural phenomena. Testing and evaluating are also crucial for students constructing scientific models in K-12 settings (Campbell & Oh, 2015; Gilbert, 2004; Louca & Zacharia, 2012; Schwarz et al., 2009). Ideally, students have multiple opportunities to test their models through experiments and revise their models based on their results. As iterative refinement helps students make sense of a phenomenon in a constructionist manner, it is considered a key element of metamodeling knowledge (Schwarz et al., 2009; Krell & Kruger, 2016).

In computational modeling, both the systems dynamics and CT communities agree on the importance of testing and debugging (Barlas, 1996; Brennan & Resnick, 2012; Csizmadia et al., 2015; Sins et al., 2005). Several system dynamics studies recognize model evaluation or interpretation (i.e., students' ability to meaningfully analyze model output data and determine

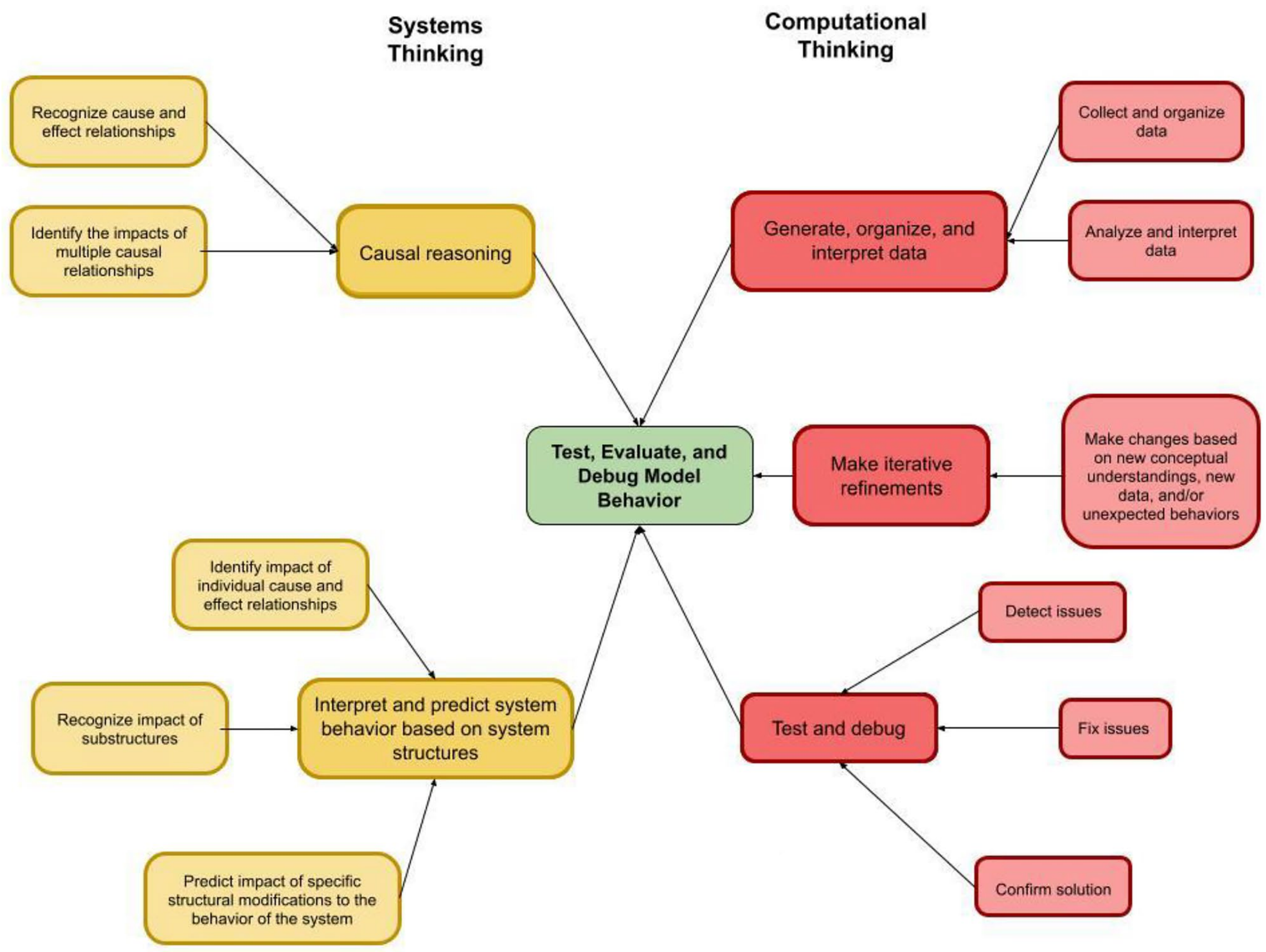

Systems Thinking
Computational Thinking
Recognize cause and effect relationships
Identify the impacts of multiple causal relationships
Causal reasoning
Collect and organize data
Analyze and interpret data
Generate, organize, and interpret data
Test, Evaluate, and Debug Model Behavior
Make iterative refinements
Make changes based on new conceptual understandings, new data, and/or unexpected behaviors
Identify impact of individual cause and effect relationships
Recognize impact of substructures
Interpret and predict system behavior based on system structures
Predict impact of specific structural modifications to the behavior of the system
Detect issues
Fix issues
Confirm solution
Test and debug

Fig. 1 Visual representation of our framework for "Test, evaluate, and debug model behavior." This diagram is a visual representation of the various ST and CT aspects that are included in our understanding of the computational systems modeling practice of "Test, evaluate, and debug model behavior" based on the work of Shin et al. (2022). On the left-hand side are the various ST sub-aspects that flow into the ST aspects that support this practice, while the right-hand side shows the CT aspects and subaspects involved in this practice

how their model functions based on its structures) and model revision (i.e., changes students make to their models based on their model evaluations) as core components of computational modeling (Barlas, 1996; Hogan & Thomas, 2001; Stave, 2002). Likewise, CT literature also emphasizes the importance of troubleshooting or debugging and iterative refinement (Brennan & Resnick, 2012; Csizmadia et al., 2015; Katz & Anderson, 1987; Li et al., 2019; Swanson et al., 2021; Wilensky & Reisman, 2006). Troubleshooting occurs when a problem is identified in an algorithmic system (Jonassen & Hung, 2006; Li et al., 2019). Once identified, a systematic search for the source of the problem is often conducted through debugging techniques (Aho, 2012; Li et al., 2019; Sullivan & Heffernan, 2016). Iterative refinement involves making gradual changes to an algorithmic system (in this case a computational model) and often happens in response to new information (Brennan & Resnick, 2012; Ogegbo & Ramnarain, 2021; Shute et al., 2017; Wilensky & Reisman, 2006).

Building on this literature, our view of the modeling practice of testing, evaluating, and debugging involves students first evaluating model structure (Hmelo-Silver et al., 2017) and model output (Hadad et al., 2020) and then comparing their model to their conceptual understanding and/or external data (Weintrop et al., 2016), and, finally, making informed changes to their model based on these analyses (Aho, 2012; Sengupta et al., 2013). Within our framework (Fig. 1), the synergy between ST and CT in supporting students in this practice is thoroughly fleshed out (Shin et al., 2022). The ST aspects of causal reasoning and predicting system behavior based on system structure often help students evaluate their model structure and make informed decisions about model revisions (Lee & Malyn-Smith, 2020; Shute et al., 2017). The CT aspects of iterative refinement, data analysis, and systematic troubleshooting help students identify flaws in their models so that they can make necessary changes (Aho, 2012; Sengupta et al., 2013; Türker & Pala, 2020; Yadav et al., 2014).

Despite being identified as a core aspect of computational modeling across many studies, testing and debugging are challenging (Grapin et al., 2022; Li et al., 2019; Sins et al., 2005; Stratford et al., 1998). Students often hesitate to make revisions to their models based on new evidence, and those who make changes tend to be conservative with their model revisions (Grapin et al., 2022; Swanson et al., 2021; Wilensky & Reisman, 2006). Another study suggests that students often take an ad hoc outcome-oriented stance toward testing and debugging (Sins et al., 2005). In these cases, students seek to modify their models so that they match an external set of data using the minimal number of changes possible rather than focusing on having their models match their conceptual understanding of the phenomenon (Li et al., 2019; Sins et al., 2005; Wilensky & Reisman, 2006). This often results in models that functionally produce the correct outcome but often lack internal consistency and explanation power (Sins et al., 2005; Wilensky & Reisman, 2006). Additionally, this outcome-oriented approach greatly reduces the potential of testing and debugging to support student learning by shifting the modeling process away from being a sense-making tool toward being an ad hoc engineering problem (Hogan & Thomas, 2001; Sins et al., 2005). Given these challenges, finding evidence of students using testing and debugging in sophisticated ways and identifying aspects of a learning environment that can support students in this work becomes critical.

Research Questions

Although the ST and CT literature argues for the importance of students using the modeling practice of test, evaluate, and debug model behavior, research shows that students often have challenges with this practice (Li et al., 2019; Sins et al., 2005; Stratford et al., 1998; Swanson et al., 2021; Wilensky & Reisman, 2006). Our goal was to identify instances of students testing and debugging their computational models and to examine different behavioral patterns that student groups can use to engage in this practice. In this paper, we define "behavior" as a distinct student action or series of actions occurring within a discrete timeframe and "cognitive behavioral pattern" as a long pattern of behaviors found across multiple episodes that suggest a generalized approach to testing and debugging. Additionally, we recognize that the learning environment can either support or hinder students (Assaraf & Orion, 2005) in building proficiency with testing and debugging. We thus set out to answer the following research questions within a design-based research environment centered on a high school chemistry unit on evaporative cooling developed according to PBL principles.

RQ1: what different cognitive behavioral patterns do students use to approach testing and debugging within a computational modeling unit on evaporative cooling?
RQ2: what testing and debugging behaviors do students seem to use more frequently within the context of a computational modeling unit on evaporative cooling?

Methods

Study Context and Learning Environment

Learning Environment and Participants

This study is based on data collected in January–February 2020 from a 6-week high school chemistry unit on evaporative cooling. This unit was implemented at a STEM magnet school (which we call Faraday High School or FHS as a pseudonym) in a small midwestern city. While it is a publicly funded institution, students need to apply to this school from across a broad catchment area consisting of three counties with admission based primarily on student academic test scores. Approximately 21% of the student body is part of a racial or ethnic minority, and approximately 54% of students are on free or reduced lunches. Two of the authors (observers A and B) partnered directly with two high school chemistry teachers (Mr. H and Mr. M). Mr. H is a middle-aged White male with approximately 15 years of teaching experience, and Mr. M is a young White male with 4 years of teaching experience. For this unit, Mr. H and Mr. M each taught 2–3 sections of 10–25 students for a total of 103 student participants. As a sophomore chemistry class, this was the first high school level chemistry class for these students, with their freshmen year spent learning key physics concepts. Because FHS runs on a block schedule, each section meets for 80 min every other day.

Curriculum

The evaporative cooling unit was developed using PBL principles, which include starting the unit with a driving question grounded in a real-world phenomenon, exploring the driving question and the phenomenon through engaging in science practices, and scaffolding the unit with learning technologies (Krajcik & Shin, 2022). The evaporative cooling process results in liquids getting colder during evaporation as faster moving liquid particles with a high average kinetic energy (KE) tend to be the first particles to overcome the intermolecular forces (IMFs) of attraction. Overcoming these forces is what causes molecules in the liquid phase to enter the gas phase. As these high-KE particles leave the liquid phase, the average KE of the remaining liquid molecules decreases, making the substance colder. The KE of the faster moving liquid particles is transferred to the potential energy (PE) of the gas particles.

At the beginning of the unit, students were initially tasked with drawing a two-dimensional model of the evaporative cooling phenomenon on whiteboards. Students were then introduced to the SageModeler computational modeling program (Damelin et al., 2017) along with some of the key aspects of computational modeling, such as the need to recontextualize the phenomenon as a set of interacting variables in order to create a workable computational model in SageModeler. Students then worked in small groups (two to three students) to construct and revise a computational model of this phenomenon that addressed the cooling effect of evaporation and included IMF to explain why some liquids evaporate faster than others. Students had multiple opportunities to test, debug, and revise their computational models over the 6-week unit.

These opportunities for students to test and debug their computational models included teacher and peer feedback, written reflections, and specific features embedded in the computational modeling program. The classroom teacher regularly visited each student group to ask them questions about their computational models. These questions provided opportunities for students to identify areas in their models that needed improvement and make changes accordingly. Student groups provided structured feedback to each other. By examining the computational models of other student groups and receiving feedback on their own models, the peer feedback cycle helped students identify aspects of their computational models that needed improvement. The students were also instructed to write down their reflections on the revision process after each revision cycle. These written reflections helped students assess any recent changes they had made to their models and consider what additional revisions might be needed in later modeling sessions. Finally, students were encouraged to use the testing and debugging features embedded in the computational modeling program (defined below) as they worked within their small groups to make changes to their computational models.

SageModeler

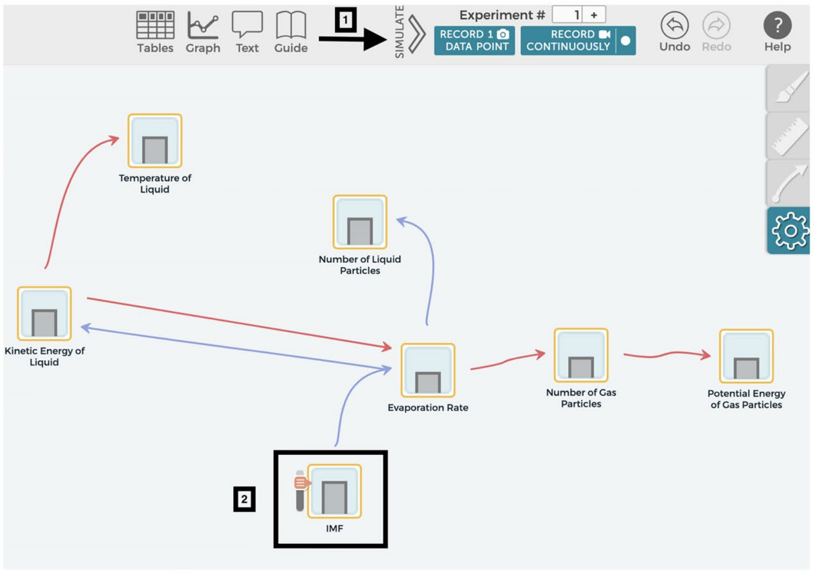
Within the evaporative cooling unit, students build, test, and revise computational models using SageModeler, a free, browser-based, open-source software program. SageModeler allows students to set certain variables as “collectors” (variables that can accumulate an amount over time) and transfer valves or flows between these collector variables. Additionally, SageModeler offers two main testing and debugging features that students can use to evaluate model output and compare their models to real-world data: simulation and graphing features. Using the simulation feature, students can generate model output for all variables in their model, enabling them to test how the model output changes under different initial conditions (Fig. 2a). Students can assess both the overall behavior of their model and examine how specific structural changes might impact this behavior. The graphing feature of SageModeler facilitates students in testing the relationship between any two variables in their model as one input variable is being manipulated (Fig. 2b). Graphs serve two important functions; they allow students to 1) look at the correlation between two distal variables and 2) compare their model’s output to real-world data. Students can generate a graph between two variables in their model and then compare this model-generated graph to a graph of real-world data (Fig. 2c).

Data Collection

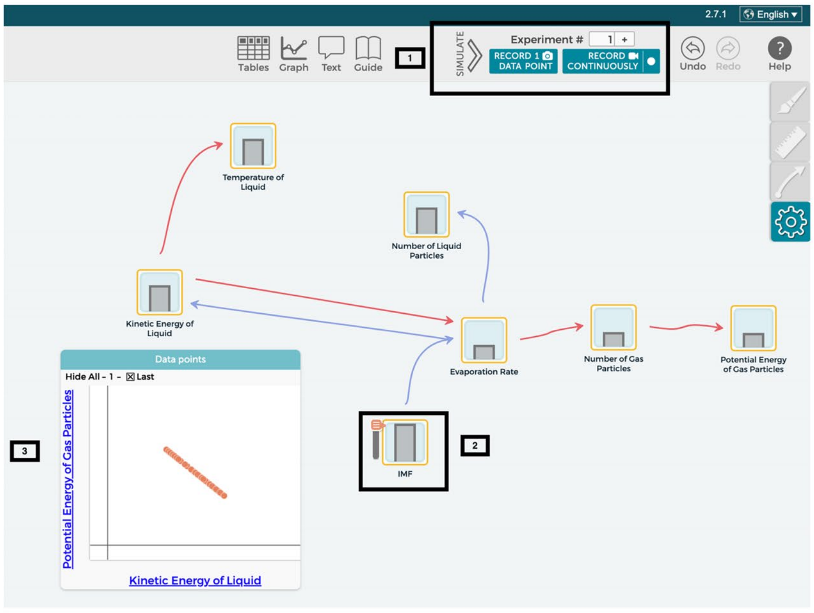
The primary source of data for this research are screencasts, which are both video recordings that capture the

Fig. 2 Testing and debugging features of SageModeler. **a** Simulation feature. This figure demonstrates the simulation feature of SageModeler. The simulate function is turned on to allow for the student to generate model output (1). The student then manipulated the input variable "IMF" (by moving its associated slider bar up and down) (2) to determine its impact on downstream variables. **b** Graphing feature of SageModeler. This figure demonstrates the graphing feature of SageModeler. The students begin by using the Record Continuously icon (1), which allows them to record how the different variables change as the input variable (2) is manipulated. Using these recorded data, the students can then create a graph in SageModeler showing the relationship between any two variables (3). **c** Data comparison using SageModeler. This figure shows how students can input external data into SageModeler and compare it to their model output. Notice that the external data (graph on the right) shows an exponential relationship between potential energy and kinetic energy, which suggests that these students need to revise their computational model

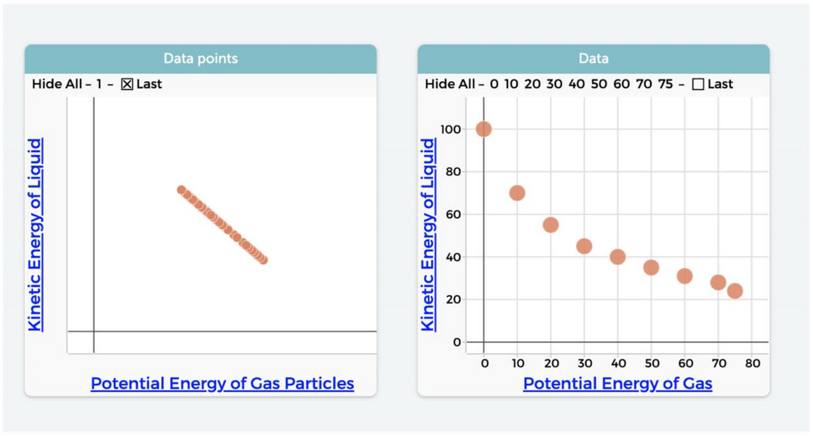
a



b



c



various activities occurring on a laptop screen and audio recordings of the computer's microphone. Screencasts allow researchers to observe how students construct and revise their computational models as well as the dialogue between group partners. From these screencasts, we can observe changes students make to their models, ascertain their reasoning for making these changes through their dialogue, and glean insights into their approach to testing and debugging. For this study, we focus on screencast data from five groups of students, three from Mr. H's class and two from Mr. M's class (Table 1). These screencast groups were recommended to us by Mr. H and Mr. M as they were among their more talkative students and gave consent for the screencast process. While other students were not chosen to be screen casted, they were present in the classroom and gave permission for their classroom discourse (including conversations with screencast groups) to be recorded for this project. Note that all names described in this manuscript are pseudonyms meant to protect student identities. Non-screencast students are given letter-based pseudonyms (e.g., student A, student B, etc.) when engaging in conversations with screen casted students.

Instrument Development

To categorize how students test and debug their models, we use the ST and CT identification tool (ID Tool). The ID tool is based on "A Framework for Computational Systems Modeling" (Shin et al., 2022; Bowers et al., 2022) and was validated by a team of experts who reached a 92% agreement (Cohen's kappa 0.87) among raters. Given that the six indicators of this tool are all contextualized within the computational modeling practice of test, evaluate, and debug model behavior, we used these indicators to investigate student testing and debugging behaviors in the evaporative cooling unit. The six testing and debugging indicators are listed in Table 2. Each indicator contains a four-part classification system (from levels 1 to 4 in ascending order) that explores the sophistication of observed student behavior.

Table 1 Student demographics

Student group	Teacher	Grade level	Gender identity
Group 1: Andy and Ben	Mr. H	10th	Male/male
Group 2: Leslie and Aubrey	Mr. H	10th	Female/female
Group 3: Robert, Mark, and Jerry	Mr. H	10th	Male/male/male
Group 4: Ron and Tom	Mr. M	10th	Male/male
Group 5: Rashida and Donna	Mr. M	10th	Female/female

Data Analysis

Using the ID Tool and Primary Analysis

We used the ID tool to conduct a primary analysis of the screencast data. Using Atlas.ti software, we annotated the screencast videos to mark instances where students were exhibiting testing and debugging behaviors based on the rubric described by our ID tool. This annotation method was previously validated by Bowers et al. (2022). To maintain interrater reliability throughout this study, we engaged in periodic member checking where all scorers independently scored a 30-min segment of student video to see if our coding results drifted from each other. As we annotated these specific instances using the ID tool, we also developed descriptive memos to record notes of what was occurring in each specific episode. These descriptive memos summarized student actions in a narrative manner to make it easier for us to begin looking at broader behavioral patterns governing student testing and debugging. Because the ID tool is primarily useful for identifying the extent of student testing and debugging behaviors, the descriptive memos were necessary for determining broader testing and debugging behavioral patterns. The primary analysis using the ID tool along with the supplementary memos allowed us to identify instances where students were testing and debugging their models. In subsequent analysis, the ID tool coding of the screencast results was used to create a timeline of the testing and debugging behaviors, which, along with the supplementary memos, informed our narrative analysis of testing and debugging behavioral patterns.

Timeline Construction and Analysis

After analyzing screencast videos, we constructed a spreadsheet-based timeline for each of the five screencast groups that show which indicators students exhibited within a specific 5-min interval (Table 3). If students had separate or overlapping episodes between two adjacent time points where they exhibited indicators of A, B, and E, all three indicators were included within the timeline for that interval. Along with marking which indicators were present in each time interval, the highest level of student performance associated with said indicator within that time frame was also noted. The constructed timeline served as a tool for recording and organizing patterns of student testing and debugging behaviors and subsequently informed later narrative analysis of student testing and debugging cognitive behavioral patterns and a summative quantitative analysis of student testing and debugging behaviors.

Table 2 Description of key indicators from the ST and CT identification tool

Indicator	Description	Brief level descriptions
A: sense-making through discourse	Students either verbalize their reasoning for making changes to their models or engage in conversations about why specific aspects of their models need to be improved	Level 1: verbalize changes to model or identify areas needing revisions but no reasoning Level 2: verbalize reasoning but no mutual dialogue Level 3: back and forth dialogue with verbal reasoning Level 4: back and forth dialogue with verbal reasoning and impact on other parts of model
B: analyzing model output: simulations	Students use embedded model output tools to analyze how their model behaves under different input conditions. In this case, students use the simulation tool in SageModeler to test their models	Level 1: adjusting one or more input variables but no verbal reasoning Level 2: adjusting input variables with verbal reasoning but no dialogue Level 3: adjusting input variables with verbal reasoning and dialogue, focus on local behavior Level 4: adjusting input variables with verbal reasoning and dialogue, holistic model discussion
C: analyzing model output: graphs	Students use embedded model output tools to analyze how their model behaves under different input conditions. In this case, students generate and analyze graphs in SageModeler	Level 1: unsuccessful attempt to make a graph in SageModeler Level 2: successful graph creation but no interpretation Level 3: successful graph creation with discussion of implications for the graphed variables Level 4: successful graph creation with discussion of the broader implications for model behavior
D: analyzing and using external data	Students use external data sources to verify model behavior. At more sophisticated levels, students compare specific external data sources directly to their models and discuss the validity of the external data	Level 1: superficial reference to data or referencing inaccurate data Level 2: reference external data to inform revisions but no direct comparisons to model output Level 3: compare specific external data to model output without discussion of data validity Level 4: compare specific external data to model output with discussion of data validity
E: using feedback	Students receive meaningful feedback from others (teachers or peers), discuss the validity of the feedback, and use feedback to inform model revisions. At more sophisticated levels, students test their models after making recommended changes and have a follow-up discussion with others to share their new insights	Level 1: students receive feedback but do not discuss it or use it to inform revisions Level 2: students make changes to their models based on feedback but do not discuss the validity of the feedback Level 3: students receive feedback, discuss its validity, and make or do not make changes to their models based on feedback Level 4: students receive feedback, discuss its validity, make or do not make changes to their models based on feedback, and share reflections with another group
F: reflecting upon iterative refinement	Students reflect through writing or discourse on the changes they have made to their models. At more sophisticated levels, students give a defined rationale for the changes they have made	Level 1: ambiguous surface level reflection without reasoning Level 2: list specific model changes but do not provide detailed reasoning Level 3: list changes and reflect upon reasoning Level 4: list changes, reflect upon reasoning (with a defined rationale), and discuss broader changes to models

Table 3 Testing and debugging timeline for group 1

Episode	13-Jan	15-Jan	15-Jan	27-Jan	27-Jan	27-Jan	27-Jan
Time	10:00	55:00	60:00	50:00	55:00	60:00	65:00
Codes	A(3)	A(1)	A(1), E(2), F(2)	E(1)	A(1), E(2)	E(2)	A(3), D(2), E(2)
Episode	29-Jan	29-Jan	29-Jan	29-Jan	29-Jan	31-Jan	10-Feb
Time	5:00	10:00	15:00	25:00	30:00	5:00	60:00
Codes	A(3) D(2) E(3)	A(3) D(2) E(3)	A(3) D(2) E(2)	B(3) E(1)	B(3) D(2) E(4) F(2)	A(1) F(2)	A(3) B(4) E(2)
Episode	10-Feb	10-Feb	12-Feb	12-Feb	12-Feb	14-Feb	14-Feb
Time	65:00	70:00	5:00	10:00	15:00	5:00	10:00
Codes	A(2) B(2)	B(2)	A(3) E(2)	A(2) B(3)	A(3) B(2) D(2) E(1) F(2)	D(2)	D(2)
Episode	14-Feb	14-Feb	14-Feb	14-Feb	14-Feb	14-Feb	14-Feb
Time	15:00	20:00	25:00	35:00	40:00	55:00	60:00
Codes	D(2)	D(2) E(1)	D(1) F(3)	D(2)	D(2)	A(2) B(2) E(2)	F(3)

Narrative Analysis

Once the initial timeline was constructed, we conducted a narrative analysis for three student groups. While the timeline demonstrated general patterns of student testing and debugging behaviors, a more comprehensive analysis focusing on key episodes from the screencasts was needed to describe student testing and debugging cognitive behavioral patterns. Returning to the descriptive memos of each group, we started by looking for specific episodes that clearly demonstrated students exhibiting specific indicators. We also looked for patterns and outliers between episodes within the same student group and between student groups, so that we could articulate the major differences in the testing and debugging behaviors of these five groups to write a cohesive narrative for each group. We then compared these narratives to the timeline analysis to check for internal consistency. This allowed us to address RQ1.

Although we conducted a quantitative analysis using data from all five groups, we selected three groups for the narrative analysis that represent the breadth of student testing and debugging cognitive behavioral patterns. We did not select groups 4 and 5 for the narrative analysis because their behavioral patterns overlapped greatly with those of groups 1 and 2, respectively. We also endeavored to show the diversity of behavioral patterns that can occur within a single class of students, so our narrative analysis deliberately only includes students from Mr. H's class.

Semi-Quantitative Analysis

After conducting the narrative analysis, we returned to the timeline (which also served to help structure our narrative analysis) to examine student testing and debugging behaviors from a more quantitative perspective. We constructed a frequency table based on this timeline to respectively show how frequently each indicator was observed across all five student groups and how each group differed in exhibiting the six testing and debugging indicators. By aggregating the timeline data into a single frequency table, we were able to determine which testing and debugging behaviors were most common across these five student groups. This semi-quantitative comparison of student testing and debugging behaviors primarily served to supplement the qualitative analysis of student testing and debugging behavioral patterns and functions as an additional method for visualizing our findings from our narrative analysis in order to address RQ2.

Results

Research question 1: What different cognitive behavioral patterns do students use to approach testing and debugging within a computational modeling uniton evaporative cooling?

Group 1: Andy and Ben

Compared to the other groups, group 1 relied more on collaboration with the broader classroom community (indicator E: using feedback) as a form of checking the validity of their model and figuring out ways to refine their conceptual understanding (Table 4). For example, on day 2 when trying to set the boundaries of their system, Ben wrote to students C and D (both from a non-screencast group) in the online platform, "*What is the scale range you will be using to model the system? Will you focus only on what you have been able to observe?*" Student C responded, "*We will focus on what we have observed in combination with what is going on at a particle level.*" The nature of this question is further clarified by observer A who explained to these students that the idea of a "scale range" is the level at which they are modeling the phenomenon. Group 1 then decided that their model should focus on the particle level of evaporative cooling.

Table 4 Student testing and debugging behavior patterns

Group	Behavior pattern summary
Group 1	• Initially focused on receiving external evaluation and feedback from peers • Shifted toward internal analysis of model output using simulation feature
Group 2	• Sense-making discourse drove model evaluation and revision • Reflected on reasoning behind modeling decisions to identify areas of uncertainty in their models
Group 3	• Utilized simulation and graphing features of SageModeler to systematically assess model output and drive revision

This table presents a summary outlining the general testing and debugging behavior patterns each student group used and how these behaviors shifted over the course of this unit

While this behavioral pattern of borrowing ideas from other groups was generally beneficial to these students, it also occasionally led them toward considering adding non-canonical variables to their model. The following excerpt is an example of a conversation between group 1 and students C and D that took place during a peer revision discussion.

Student C: *Spacing of the molecules? Isn't that density*?
Ben: *I mean, it is talking about how far apart they are*.
Student C: *That is density*.
Student C (to student E from a second non-screencast group): *Didn't you use density in your model*?
Student E*: Do not use density in your model. He (Mr. H) will get upset. But the spacing of the particles is important*.

In this conversation, student C tried to convince Andy and Ben to add density as a variable to their model; they were stopped from doing so by student E's appeal to authority (Mr. H). Although group 1 is heavily influenced by this appeal to authority, because Andy and Ben do not simply accept peer feedback at face value but discuss it with multiple individuals and consider the validity of this feedback, they were coded at level 3 for indicator E.

Later in the unit, their behavioral patterns shifted away from focusing on peer feedback and toward incorporating the use of simulations of their model output (B: analyzing,model output: simulations) as the complexity of their model increased. While they previously opted not to use the simulation features embedded in SageModeler, they began a more deliberate testing and debugging approach. For example, after including a positive relationship between the variable "Spacing of Molecules" and the transfer valve between "Kinetic Energy" and "Potential Energy," group 1 decided to simulate their model output (Fig. 3). Through this simulation, they recognized that although they conceptually agreed with this specific relationship, they questioned the overall behavior of their model. In particular, they were concerned about the decrease in "Potential Energy" that occurs after all of the "Kinetic Energy" has been converted into "Potential Energy." Yet, despite their use of the simulation function to identify this behavioral anomaly within their model, they did not determine which specific relationship was responsible for this behavior and, therefore, were unable to make the necessary changes so that their model matched their conceptual understanding. While this was an example of Ben and Andy systematically testing their model, they had difficulty interpreting their model's structure in a way where they can identify the source of the behavioral anomaly, suggesting a gap in their computational thinking skills. Overall, group 1 seemed to rely initially on external feedback to help them identify flaws in their models before shifting toward using the simulation features to interpret their model's output.

Group 2: Leslie and Aubrey

While group 1 tended to utilize discussions with other groups, group 2 often depended on discussions with each other to make sense of the phenomenon as a system of interconnected elements and to identify where revisions were needed (indicator A: sense-making through discourse). Early in the unit, as the students were trying to decide which variables to include in their model, they had the following discussion:

Aubrey: *I don't know if it is right, but it makes sense*.
Leslie: *Now we need to add another box*.
Aubrey: *The only other variable we have is temperature. But isn't temperature a constant?*
Leslie: *Yes it is. So, our model is just two things long. That's boring. So, molecular energy goes into molecular spacing of substance. Is this all about evaporation?*
Aubrey: *Yeah*.

From this conversation, we see that these students were considering the boundaries of the system while they were also using causal reasoning by reviewing the relationship between "molecular energy" (which appears to be a student-generated term that is roughly equivalent to kinetic energy) and molecular spacing. Through this discussion, they were also identifying an area of their model that needed revision, proposing a change, and considering the ramifications of this change on their model's behavior. Thus, this is an example of students verbally testing and debugging their model.

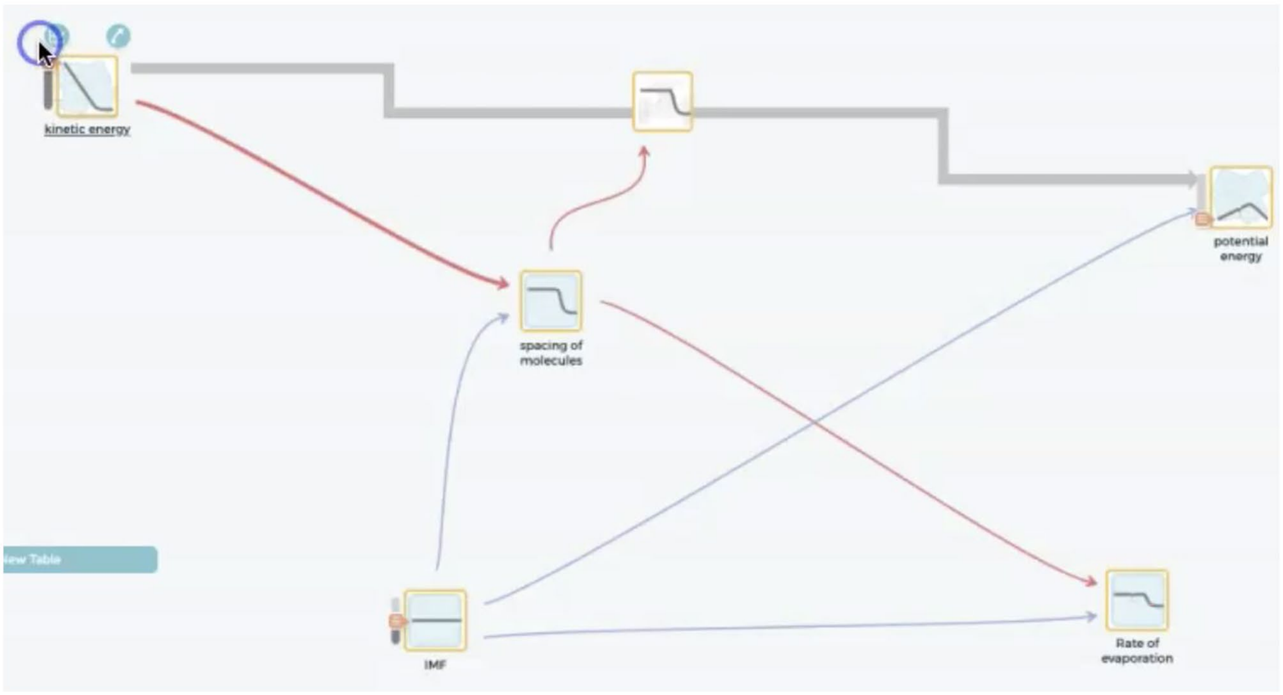


Fig. 3 Screenshot of group 1 testing and debugging their dynamic model. In this figure, the students from group 1 used the simulation features to determine how kinetic energy is impacting other variables in this model

These testing and debugging are also evident in later discussions where they verbalized their interpretation of their model's structure as they considered which changes were necessary for creating a more robust model of evaporative cooling (indicator A: sense-making through discourse). In this example, Leslie and Aubrey were trying to figure out how to revise their models in response to a recent investigation on the role of potential and kinetic energy in evaporation (day 7). In particular, they were trying to determine how the "spacing of molecules" variable (formerly called "molecular spacing") fits in their new conceptual understanding of evaporative cooling.

Leslie: *So, maybe the temperature of water also affects the spacing of molecules and then kinetic energy affects potential energy, which also affects the spacing of molecules.*
Aubrey: *Maybe*.
Leslie: *We'll try it. But maybe it doesn't.*
Aubrey: *Well, for sure this one* [pointing to the "Temperature of Water" variable].
Leslie*: Okay, spacing of molecules. As the temperature of water increases, the spacing of molecules increases more and more. So, remember that one model that we did.*
Aubrey: *Yeah, like the hexagon thing where they kept on getting more and more spread apart* (referencing a simulation that showed how as the liquid heated up, the kinetic energy increased until it hit the boiling point, after which the potential energy started to increase as the molecules moved farther apart).

Not only does this excerpt show how dialogue is manifested in the practice of testing and debugging, it is also a clear example of how these students used external data to validate their sense-making (indicator D: analyzing and using external data). This is subsequently followed up by the use of written reflections as an additional form of sense-making. At first the students wrote, "*as the temperature of the water (average kinetic energy) increases the molecules start gradually moving faster and hitting each other and breaking their force of attraction keeping them together and become gas*." While this initial written explanation is an accurate justification for this relationship, they disagreed with the second part of this explanation and replaced everything after "hitting each other" with "*... and move farther apart. As the temperature of water increases they move more quickly than before and move farther apart than they were*." This writing seemed to help these students reflect upon their causal reasoning for this relationship.

Group 2 expanded upon this use of written reflection by placing explanations of each relationship directly on the SageModeler canvas (Fig. 4). Writing these notes supported their causal reasoning about relationships and served as a means of considering the validity of each relationship they had encoded into SageModeler, thereby acting as an alternative approach to the type of formal testing and debugging that is often conducted at this stage in model development. Their embedded notes also had the potential to support later revision efforts as they could have identified their original rationale behind a particular relationship and considered if new evidence supported or undermined that explanation for that causal relationship (indicator F: reflecting upon iterative refinement). Collectively, their verbal dialogue and written reflections demonstrate that group 2 engaged with testing and debugging primarily through discourse (Table 4).

Group 3: Robert, Mark, and Jerry

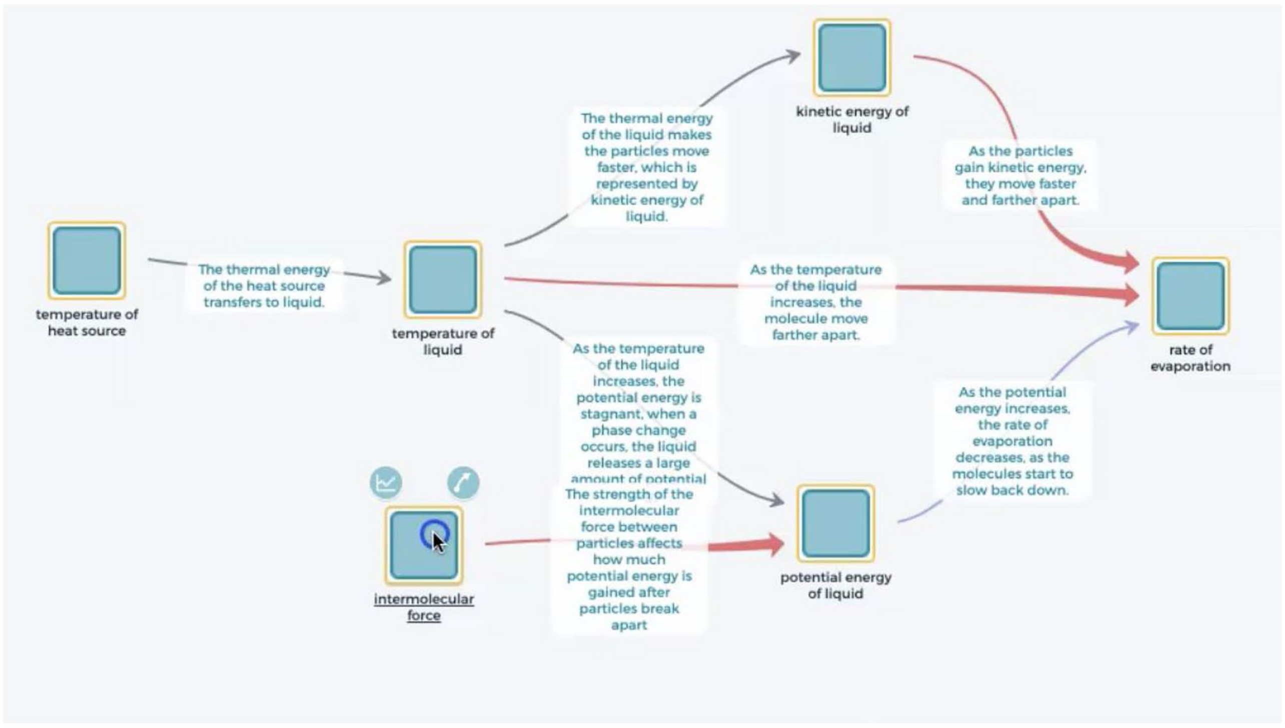
Group 3 utilized the testing and debugging features embedded within SageModeler (indicator B: analyzing model output: simulations and indicator C: analyzing model output: graphs) as they analyzed their models to determine which changes to make. One interesting example of systematic testing and debugging occurred when the students inserted a "dummy variable" into their model to see the effects of adding a fourth variable on the behavior of their model (Fig. 5). After inserting this dummy variable, they used the simulation feature to observe its impact on the behavior of the model as a system (indicator B: analyzing model output: simulations). However, they quickly removed the dummy variable, suggesting their dissatisfaction with its effect on model behavior. This use of a dummy variable along with their subsequent discourse is strong evidence that these students were using testing and debugging as they made a deliberate change to their model to see how it would impact model behavior and then removed this after testing this change and determining that it was unsatisfactory.

Another example of group 3 using the model simulation features to support their testing and debugging occurred as they were trying to decide which relationships to set between the variables of "kinetic energy," "potential energy," and "density." It is important to note that other screencast excerpts demonstrate that these students held non-canonical ideas about "density" at this stage in the unit. Most notably, they viewed "density" as an extrinsic characteristic of a substance that decreased as a substance changed from a liquid to a gas. As such, their understanding of "density" is closer to the canonical understanding of "molecular spacing of molecules."

Jerry: *Kinetic energy does what?*

Robert: *Okay, so as intermolecular force (IMF) increases, does density increase in the end?*

Jerry*: It would be the other way around. As IMF increases, density decreases. But...*



The thermal energy of the heat source transfers to liquid.

The thermal energy of the liquid makes the particles move faster, which is represented by kinetic energy of liquid.

As the particles gain kinetic energy, they move faster and farther apart.

As the temperature of the liquid increases, the molecule move farther apart.

As the temperature of the liquid increases, the potential energy is stagnant, when a phase change occurs, the liquid releases a large amount of potential

The strength of the intermolecular force between particles affects how much potential energy is gained after particles break apart

As the potential energy increases, the rate of evaporation decreases, as the molecules start to slow back down.

Fig. 4 Screenshot of group 2's annotated model. The students in group 2 wrote their rationales for each relationship on their model as a form of sense-making during the testing and debugging process

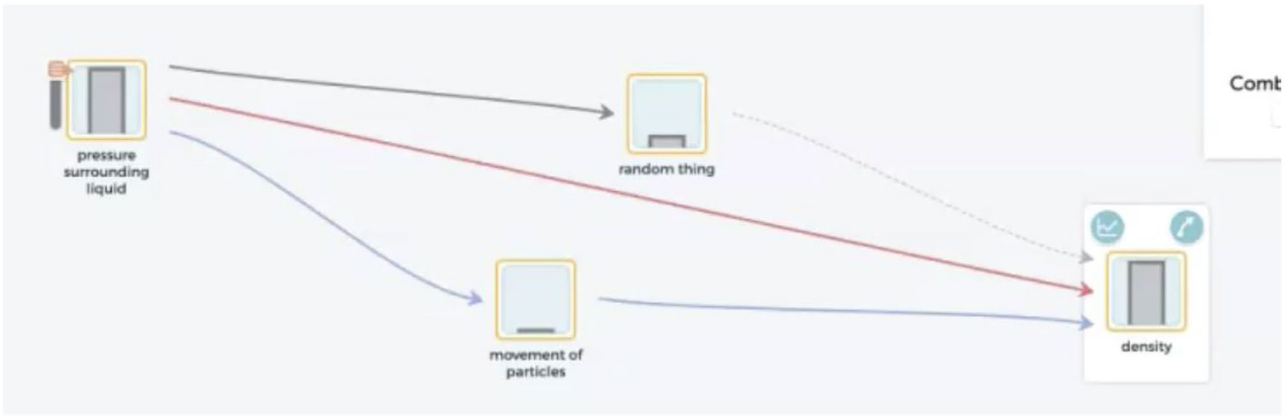


Fig. 5 Example of group 3 utilizing a "dummy" variable to facilitate testing and debugging. When group 3 was trying to decide if any additional variables might be needed in their model, they inserted a "dummy" variable (which they named "random thing") to see how it would impact model behavior

> Robert*: ...which means one of these* [relationships] *has to be increased and the other has to be decreased, or they both have to be decrease.*

Jerry: *The kinetic energy (KE) is opposite of potential energy (PE).*

Robert: *So, this would probably be the one that is decreasing?*

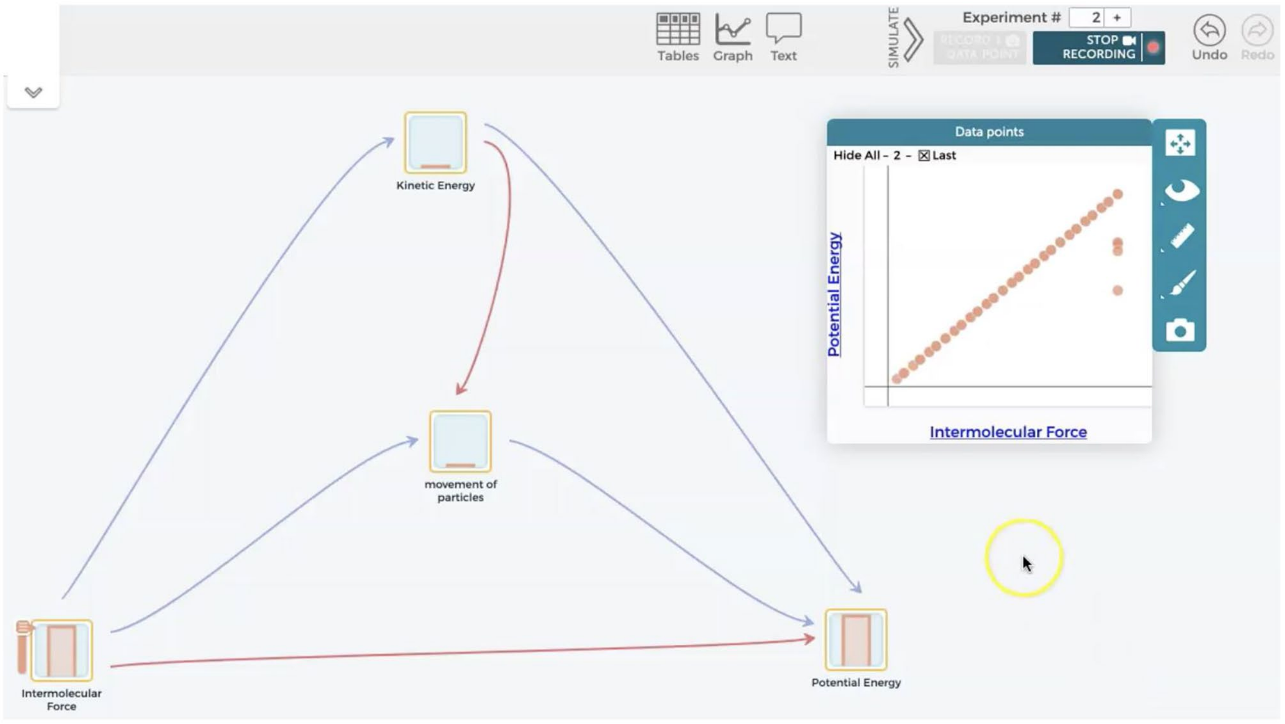
> Jerry: *But that doesn't make sense because of the graph he* [Mr. H] *showed us. Just put decreasing. Wait. Actually, it would be increasing* [KE to PE] *and the last one* [PE to Density] *would be decreasing?*

The students changed the relationship between "PE" to "Density" to decreasing and then simulated the model and saw that an increase in IMF causes the density to decrease (which in their understanding would mean an increase in the molecular spacing of the substance). In this example, the students first considered the overall behavior of their model of evaporative cooling. The students then analyzed the individual relationships within this system to determine how these relationships would influence system behavior. Upon identifying how these relationships would impact the model's output, they considered how these individual relationships reflected their understanding of the real-world phenomenon and ultimately selected a specific relationship to modify. After modifying this relationship, they once again used the simulation features to see the impact of this change upon the model output. This is an example of indicator A: sense-making through discourse and indicator B: analyzing model output: simulations.

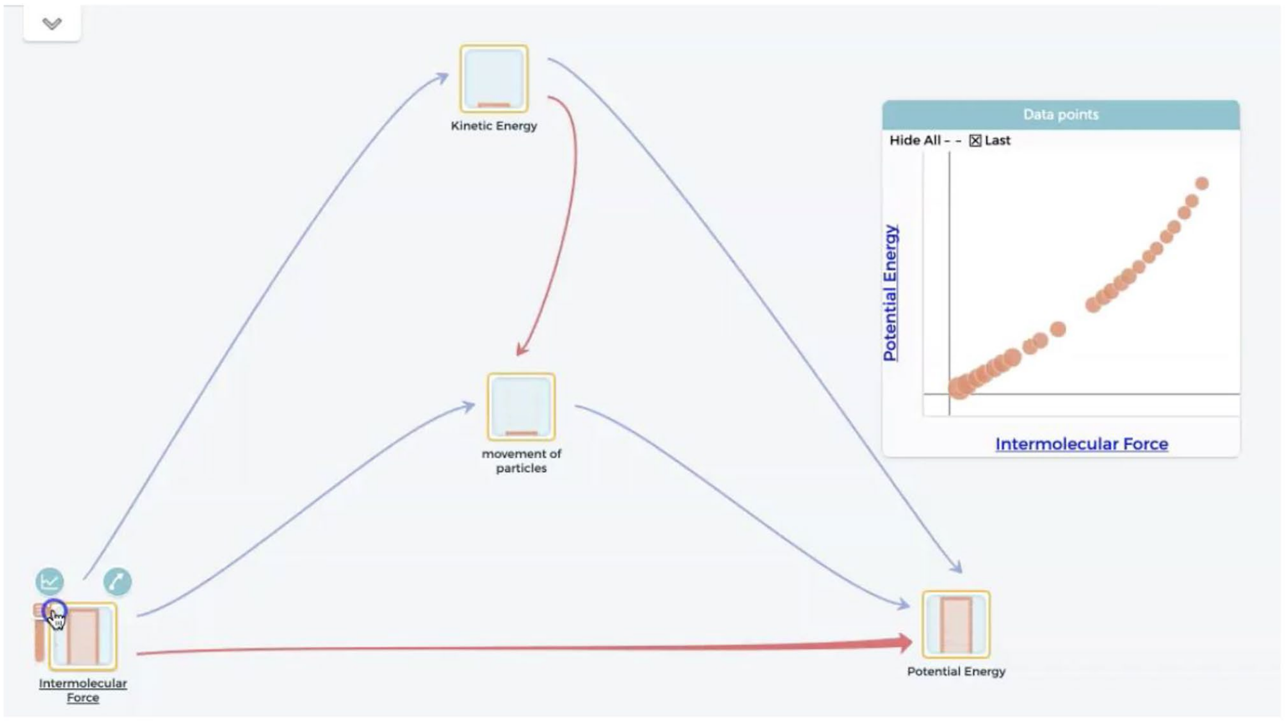
Group 3 also used the model output generated from SageModeler to make a graph of the relationship between IMF and PE. After making several changes to their model, the students tested to see how these changes impacted the overall behavior of the system. They used the simulate feature to look at how manipulating the input variable (Intermolecular Force) of their model would affect intermediate and distal output variables. Given that they were specifically interested in how the IMF impacted PE, they used the simulation output to generate a graph in SageModeler (Fig. 6a). Upon making this graph, they recognized that apart from a few outlier points at the end (likely artifacts from previous simulations), there was a linear relationship between IMF and PE, which was not in line with their understanding of the relationship between these two variables. They subsequently changed the individual relationship between IMF and PE to an exponential one, which in turn made its associated graph into an exponential relationship (Fig. 6b). This is an example of students using indicator C: analyzing model output: graphs. While other student groups periodically used the simulation features to explore the output of their models, only this group used the model output to successfully make graphs of the relationships between two variables in their model. Overall, group 3 tended to focus on testing and debugging behavioral patterns that prioritized systematically analyzing their model output to identify areas of their model that needed improvement (Table 4).

Research question 2: What testing and debugging behaviors do students seem to use more frequently within the context of a computational modeling unit on evaporative cooling*?*

Based on our semi-quantitative analysis of student screencasts from all five focus groups, it appears that there is evidence that all six testing and debugging indicators were used at least once by a student group within this dataset (Table 5). Although all indicators were present, some indicators were more commonly used than others. In particular, indicators associated with sense-making discourse (as exemplified by A) were more frequently used while indicators that



a



b

Fig. 6 Example of group 3 exhibiting evidence of indicator c: analyzing model output: graphs. **a** Group 3 pre-revision model (day 8) with graphical representation of relationship between IMF and PE. **b** Group 3 revised model (day 8) with graphical representation of relationship between IMF and PE

are linked to comparing models to external data (D) were less frequently used. This implies that while the learning environment supported students in using sense-making discourse, it did not sufficiently support students in comparing their models to external data (D). Additionally, there is a strong contrast between the frequency at which students used SageModeler's simulation (B) and graphing (C) features. Even though both of these SageModeler features allow students to examine their model output, the simulation feature focuses on how the relative amount of each variable changes as the input variables are manipulated. In contrast, the graphing feature allows students to compare the relationship between two individual variables in isolation from other aspects of the model (Fig. 2b). Student preference for the simulation feature (B) over the graphing feature (C) suggests that these students found the simulation feature easier to navigate and/or more useful for the learning tasks in this unit. Given that the graphing feature tends to better support direct comparison with external data and the noted low use of external data by students in this unit, the latter explanation has merit.

The results from the indicator analysis are largely consistent with the results from the narrative analysis as these data show a preferred set of behaviors for each group (Table 5). Group 1 was more apt to reference external data (D) and use external feedback (E) to drive their revision process while minimizing their use of model output simulations (B) and not using model output graphs (C). This contrasts greatly with group 3, which strongly prioritized analyzing model output (B) over using external data (D) and external feedback (E). Indeed, group 3 is one of only two groups to use the graphing feature (C) and the only one to do so successfully. While group 2 attempted to use the graphing feature (C), they strongly preferred using discourse as their primary means of model analysis (A). Group 5 also prioritized discourse (A) for testing and debugging but also frequently utilized model simulation features (B). Finally, group 4 had a strong preference for using feedback (indicator E) but tended to have limited discussions on the meaningfulness of the feedback, so their behavior was assessed at a lower level using the ID tool. Overall, these comparative results from both our narrative and semi-quantitative analyses show that despite having a common learning environment, student groups found opportunities to approach testing and debugging in unique ways. This suggests that multiple scaffolds and supports are likely needed to help all students test, evaluate, and debug their models.

Discussion

Our results demonstrate that within the evaporative cooling learning environment, there is evidence of student behavior that corresponds to all six of testing and debugging indicators in the ID tool. As anticipated, some behaviors corresponding to certain testing and debugging indicators occurred more frequently than others, with students particularly spending more time analyzing their models through discourse (A) compared to other indicators (Table 5). However, it is also important to mention that differences in student behaviors, as noted by both narrative analysis and quantitative analysis, demonstrate that even within a common learning environment, student groups may adopt different approaches to testing and debugging (Tables 4 and 5).

Importance of Learning Environment

Because each student group used a different set of cognitive behavioral patterns for testing and debugging within a common PBL-aligned learning environment, it appears that multiple supports are likely needed to accommodate these different approaches to testing and debugging. Having multiple pathways for students to engage in the learning process allows students to leverage their unique strengths

Table 5 Relative occurrence of each testing and debugging behavior

Indicator	Group 1		Group 2		Group 3		Group 4		Group 5		Total	
	N	Percent	*N*	Percent	*N*	Percent	*N*	Percent	*N*	Percent	*N*	Percent
Sense-making via discourse (A)	15	53.6	39	83.0	24	66.7	12	50.0	22	68.8	112	67.1
Model output: simulations (B)	8	28.6	23	48.9	26	72.2	7	29.2	19	59.4	83	49.7
Model output: graphs (C)	0	0.0	1	2.1	6	16.7	0	0.0	0	0.0	7	4.2
Use external data (D)	13	46.4	11	23.4	6	16.7	3	12.5	6	18.8	39	23.4
Feedback (E)	15	53.6	16	34.0	11	30.6	13	54.2	12	37.5	67	40.1
Reflecting on refinement (F)	6	21.4	13	27.7	12	33.3	8	33.3	12	37.5	51	30.5
Total number of intervals	28		47		36		24		32		167	

N is the number of 5-min intervals where we observed a particular exhibited behavioral indicator. Percent is the percentage of testing and debugging intervals during which this indicator was observed. Note that because student groups often exhibited multiple behaviors within any given 5-min interval, their percentages do not add up to 100. Total represents data from all groups and prior knowledge to further their sense-making endeavors (Basham & Marino, 2013; Hansen et al., 2016; Scanlon et al., 2018). Because this study suggests that students in the same class can utilize different testing and debugging behaviors and behavioral patterns, it reinforces the need to design multifaceted learning environments, so that all learners can fully participate in computational modeling. Although the learning environment in our evaporative cooling unit provided multiple pathways for students to participate in testing and debugging, two features that seemed to be the most meaningful for supporting students in testing and debugging were the simulation feature embedded in SageModeler and the use of student small groups, which facilitated discourse. Because the simulation feature allowed students to generate model output data in real time, students could test how changes in their model structures impacted model behavior and detect flaws in their models. This allowed students to analyze model output in a way that would not have been possible through traditional paper–pencil modeling. By making it easier to detect areas of their models that needed improvement, the simulation feature further assisted students in revising their models, thereby encouraging testing and debugging (Fan et al., 2018; Lee et al., 2011; Shen et al., 2014).

Another feature of the learning environment that supported students in testing and debugging is the use of student groups. In this unit, students worked in small groups of two to three students and were encouraged to collaborate with each other and verbalize their thought processes. By the nature of using collaborative student groups as opposed to having each student build their own models independently, students were implicitly encouraged to share their design choices and modeling behavioral patterns with their partners. These partners could in turn ask each other to provide evidence or reasoning to defend their design choices or provide a counterclaim of their own. For example, one student might state that density impacts the rate of evaporation because higher-density particles evaporate slower. Another student could then argue that density does not impact the rate of evaporation because oil is less dense than water but evaporates far slower (if at all). Through such productive argumentative discourse, students can identify flaws in their reasoning and in their model construction, prompting them to revise their models (Campbell & Oh, 2015; Kyza et al., 2011; Lee et al., 2015). As such, placing students in pairs or small groups encourages them to have these sense-making conversations (indicator A), which in turn facilitate model evaluation and model revision, both of which are key aspects of testing, evaluating, and debugging model behavior. In a similar manner, peer reviews provided further support for the model revision process (King, 1998). Having another group of students analyze their models and provide meaningful feedback often gave students a fresh perspective on their models. Their peers could detect flaws in their models that the student pair might have otherwise ignored. Students then had the opportunity to use that feedback to prompt additional sensemaking discourse and to inform future model revisions. Thus, by receiving and using feedback (indicator E), students were able to have an external party evaluate their models and provide key insights on aspects of their models that needed further review, thereby supporting students in the model revision process.

Limitations

Although this study shows some promising aspects of the design of our learning environment to support students in testing and debugging, there are both limitations with our methodology and aspects of the learning environment. It is important to note that this research took place at a STEM magnet school in a classroom environment that encouraged student discourse and collaboration. Because traditional classroom environments often lack a strong culture of student discourse, our results might not be fully applicable to all classrooms (Grifenhagen & Barnes, 2022; Jimenez-Aleixandre et al., 2000; Kelly, 2013). We also recognize that the limited sample size makes it difficult to draw broader conclusions from our semi-quantitative analysis. While these results do suggest a diversity in student approaches to testing and debugging and that certain testing and debugging behaviors are more common than others within this class, we cannot argue from this analysis alone that these are universal patterns. Given the design-based nature of this study, it was not feasible to isolate specific aspects of the learning environment to determine definitively if either the SageModeler simulation feature or the use of student groups is the most important scaffold for testing and debugging for these students. While our qualitative analysis does suggest that these factors helped support students in testing and debugging, additional factors such as teacher instructions and prior student experiences also might have contributed to our results. Additionally, it is difficult to determine why any specific student groups chose to use a particular set of testing and debugging behaviors. It is possible that more introverted students preferred testing and debugging behavioral patterns that were more focused on analyzing model outputs (indicator B) compared to extroverted students who might have gravitated towards more social approaches to model validation, such as peer feedback (indicator E). Another explanation could be that more mathematically inclined students preferred using simulations and graphs (indicators B and C) to interpret their models compared to using more verbally intensive behavioral patterns. However, both of these ideas are difficult to assess without targeted interviews and/or additional written assessments, neither of which occurred for this study.

Conclusion

Testing, evaluating, and debugging models are an important competency. Being able to identify the flaws in a model helps students engage in revisions, improving both their representational and conceptual models of a phenomenon (Barlas, 1996; Grapin et al., 2022; Sterman, 1994; Stratford et al., 1998). Frequent testing, debugging, and revision cycles also reinforce the scientific principle of iterative refinement through experimentation, which is essential to the scientific thinking process (Gilbert, 2004; NRC, 2012; Schwarz et al., 2009). Within our framework, we view testing, evaluating, and debugging model behavior as an integral aspect of computational modeling, drawing upon modeling, ST, and CT traditions (Shin et al., 2022). As it facilitates model revision and iterative refinement, this practice benefits student modeling (Schwarz et al., 2009). Likewise, model evaluation, model interpretation, and model revision are all important concepts in system dynamics that overlap with our understanding of testing, evaluating, and debugging model behavior (Barlas, 1996; Martinez-Moyano & Richardson, 2013; Richardson, 1996). Additionally, students often need to consider parts of their model structure from a systems thinking perspective to accurately identify areas of their model that need improvement and to guide the subsequent revision process (Lee & Malyn-Smith, 2020; Shute et al., 2017). Finally, our understanding of testing, evaluating, and debugging model behavior incorporates the CT concepts of debugging, wherein students systematically review their computational models to identify flaws and structural errors, and iterative refinement, the process by which students make changes to their models in response to new information (Aho, 2012; Sengupta et al., 2013; Türker & Pala, 2020; Yadav et al., 2014).

This investigation into how students used testing and debugging within the context of an evaporative cooling unit demonstrates both the possibilities and challenges of integrating this practice into secondary science education. Although we have evidence of students testing and debugging their models, the relative absence of using external data to directly validate their models is an area of concern (Table 5). It suggests that more direct curricular support is needed to encourage students to compare their models to external data. This, along with a desire to better support other aspects of testing and debugging (such as the peer review process), have led us toward making several curriculum changes. We have developed a set of model design guidelines to help students identify areas of their models that can be improved during the model revision process. These model design guidelines ask students to consider if their models have appropriately named/relevant variables, define appropriate relationships between variables, have clearly defined boundaries, and work appropriately when simulated. The last section of these guidelines asks students to consider how their models compare to real-world data, further emphasizing the importance of using external data to validate their models. In addition to scaffolding the model revision process, students are encouraged to use these guidelines when giving feedback to their peers. We also plan on being more explicit with students about which specific pieces of experimental data they should use to validate their models during model revisions. For example, we have added a built-in table to the SageModeler canvas where students can input experimental data showing how the temperature of liquids change over the course of evaporation. Finally, we are streamlining the unit to allow for more in-depth classroom discourse and more scaffolded model revisions. In this way, we hope to reduce student and teacher fatigue over the course of this unit.

In addition to curricular scaffolds, future iterations of this design-based research could investigate the role of teacher scaffolds in supporting students in using external data to validate their models, as instructional supports offer another avenue to bring this aspect of testing and debugging into the classroom. We also might further investigate the role of student groups in supporting collaborative discourse around testing and debugging and find additional ways to leverage peer revisions to best support student model revisions. Finally, future work needs to investigate how different testing and debugging behavioral patterns are linked to model outcomes. Given our relatively small sample size, it was difficult to determine any meaningful correlations between student testing and debugging behavioral patterns and either their post-unit understanding of disciplinary core ideas or the conceptual accuracy of their final models. While we hypothesize that student groups that systematically analyze the model output and frequently compare their models to real-world experimental data will end up with models that more accurately represent a canonical understanding of the system they are modeling, it is also possible that such behaviors lead to model fitting and limit opportunities for students to reflect on their evolving conceptual understanding of the phenomenon (Sins et al., 2005; Wilensky & Reisman, 2006). Therefore, while it is likely that student groups with the most robust models and deepest understanding of underlying disciplinary core ideas will use testing and debugging behavioral patterns that combine dialogic analysis (ala group 2) with systematic analysis and comparison of model output to experimental data (ala group 3), future work will be needed to address this hypothesis.

The findings of this research can not only guide future research, they have the potential to inform teacher educators and teachers in their efforts to effectively engage students in computational tasks that involve testing and debugging. By increasing awareness of the different behavioral patterns exhibited during testing and debugging, such as seeking advice from peers or making inferences based on simulations, teachers can provide a more nuanced facilitation that supports students' strategies. For instance, teachers can prompt students to utilize tools such as graphs. Teachers can also encourage the use of effective simulation strategies such as holding all variables constant except for one and comparing results across various scenarios. To foster productive discussion and critical thinking during testing and debugging, teachers can guide students in asking questions such as "How do you know that?" and "What does your model show?" and encourage simulation as a means of exploration and validation. Ultimately, by showing some of the different ways that students can test and debug their models, we hope that this research will encourage teachers to adopt holistic approaches to supporting students with this practice.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant Nos. DRL-1842035 and DRL-184203. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Data Availability Given the nature of our screencast data, which record student audio, we have decided not to make the screencasts available to the general public. All quantitative data generated from student screencasts (via our ID Tool) have been compiled into Excel spreadsheets that will be made available to the public via Researchgate at the time of this paper's acceptance. To help other authors in interpreting these spreadsheets, additional written documentation will be provided and uploaded to Researchgate alongside these spreadsheets at the time of this paper's acceptance.

Declarations

Ethical Approval This research was conducted according to the general ethical guidelines common to the field of science educational research. Prior to conducting this research, our methodologies were reviewed and approved by the appropriate institutional review board. Our research protocol was designed to be minimally invasive and to be beneficial to both the student and teacher participants. Teachers, students, and parents gave informed consent and were given the opportunity to opt out of any and all aspects of this research project. Additionally, all participant names have been anonymized to protect student and teacher privacy.

Consent to Participate Our protocol for providing teachers, parents, and students with the information necessary to make decisions about participating in this research were approved by an institutional review board prior to beginning this research. All teachers, students, and parents were given the opportunity to opt out of participating in this research without penalty. For students participating in the screencast collection process, both the students and their parents signed an informed consent form agreeing to have their screen actions and classroom conversations recorded using Screencast-O-matic. Students and parents were not penalized for opting not to participate in the screencast collection process.

Conflict of interest The authors declare no competing interests.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832–835.

Arnold, R. D., & Wade, J. P. (2015). A definition of systems thinking: A systems approach. *Procedia Computer Science, 44*, 669–678.

Arnold, R. D., & Wade, J. P. (2017). A complete set of systems thinking skills. *Insight, 20*(3), 9–17.

Assaraf, O. B. Z., & Orion, N. (2005). Development of system thinking skills in the context of Earth system education. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching, 42*(5), 518–560.

Bakos, S., & Thibault, M. (2018). Affordances and tensions in teaching both computational thinking and mathematics. *Proceedings of the 42nd Conference of the International Group for the Psychology of Mathematics Education.* Vol. 2., 107–144.

Barlas, Y. (1996). Formal aspects of model validity and validation in system dynamics. *System Dynamics Review: The Journal of the System Dynamics Society, 12*(3), 183–210.

Basham, J. D., & Marino, M. T. (2013). Understanding STEM education and supporting students through universal design for learning. *Teaching Exceptional Children, 45*(4), 8–15.

Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016). Identifying middle school students' challenges in computational thinking-based science learning. *Research and Practice in Technology-Enhanced Learning, 11*(1), 13.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education, 3*, 115–138.

Berland, L., & Reiser, B. (2009). Making sense of argumentation and explanation. *Science Education, 93*, 26–55.

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.

Bierema, A. M. K., Schwarz, C. V., & Stoltzfus, J. R. (2017). Engaging undergraduate biology students in scientific modeling: Analysis of group interactions, sense-making, and justification. *CBE—Life Sciences Education*, *16*(4), 68.

Booth-Sweeney, L., & Sterman, J. (2007). Thinking about systems: Student and teacher conceptions of natural and social systems. *System Dynamics Review: The Journal of the System Dynamics Society, 23*(2–3), 285–311.

Bowers, J., Shin, N., Brennan, L., Eidin, E., Stephens, L., & Roderick, S. (2022). *Developing the systems thinking and computational thinking identification tool.* International Conference of the Learning Sciences.

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* (Vol. 1, p. 25).

Cabrera, D., Colosi, L., & Lobdell, C. (2008). Systems thinking. *Evaluation and Program Planning, 31*(3), 299–310.

Campbell, T., & Oh, P. S. (2015). Engaging students in modeling as an epistemic practice of science: An introduction to the special issue of the “Journal of Science Education and Technology.”*Journal of Science Education and Technology*, *24*(2), 125–131.

Cronin, M. A., Gonzalez, C., & Sterman, J. D. (2009). Why don’t well-educated adults understand accumulation? A challenge to researchers, educators, and citizens. *Organizational Behavior and Human Decision Processes, 108*(1), 116–130.

Csizmadia, A., Curzon, P., Dorling, M., Humphreys, S., Ng, T., Selby, C., & Woollard, J. (2015). *Computational thinking - a guide for teachers*. Computing At School.

Damelin, D., Krajcik, J. S., McIntyre, C., & Bielik, T. (2017). *Students Making Systems Models. Science Scope, 40*(5), 78–83.

Duran, L. B., & Duran, E. (2004). The 5E instructional model: A learning cycle approach for inquiry-based science teaching. *Science Education Review, 3*(2), 49–58.

Emara, M., Grover, S., Hutchins, N., Biswas, G., & Snyder, C. (2020). Examining students’debugging and regulation processes during collaborative computational modeling in science. In *International Conference of The Learning Sciences 2020 Proceedings (ICLS 2020).*

Fan, C., Liu, X., Ling, R., & Si, B. (2018). Application of proteus in experimental teaching and research of medical electronic circuit. In *2018 3rd International Conference on Modern Management, Education Technology, and Social Science (MMETSS 2018)* (pp. 512–515). Atlantis Press.

Farris, A. V., Dickes, A. C., & Sengupta, P. (2019). Learning to interpret measurement and motion in fourth grade computational modeling. *Science & Education, 28*(8), 927–956.

Fernandes, S., Mesquita, D., Flores, M. A., & Lima, R. M. (2014). Engaging students in learning: Findings from a study of project-led education. *European Journal of Engineering Education, 39*(1), 55–67.

Fisher, D. M. (2018). Reflections on teaching system dynamics modeling to secondary school students for over 20 years. *Systems, 6*(2), 12.

Forrester, J. W. (1971). Counterintuitive behavior of social systems. *Theory and Decision, 2*(2), 109–140.

Fosnot, C. T. (Ed.). (1996). *Constructivism: Theory, perspectives, and practice*. Teachers College Press.

Geier, R., Blumenfeld, P. C., Marx, R. W., Krajcik, J. S., Fishman, B., Soloway, E., & Clay-Chambers, J. (2008). Standardized test outcomes for students engaged in inquiry-based science curricula in the context of urban reform. *Journal of Research in Science Teaching: The Official Journal of the National Association for Research in Science Teaching, 45*(8), 922–939.

Gilbert, J. K. (2004). Models and modelling: Routes to more authentic science education. *International Journal of Science and Mathematics Education, 2*(2), 115–130.

Gleasman, C., & Kim, C. (2020). Pre-service teacher’s use of block-based programming and computational thinking to teach elementary mathematics. *Digital Experiences in Mathematics Education, 6*, 52–90.

Grapin, S. E., Llosa, L., Haas, A., & Lee, O. (2022). Affordances of computational models for English learners in science instruction: Conceptual foundation and initial inquiry. *Journal of Science Education and Technology, 31*(1), 52–67.

Grifenhagen, J. F., & Barnes, E. M. (2022). Reimagining discourse in the classroom. *The Reading Teacher, 75*(6), 739–748.

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*(9), 799–822.

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer Science Education: Perspectives on Teaching and Learning in School, 19*(1), 19–38.

Hadad, R., Thomas, K., Kachovska, M., & Yin, Y. (2020). Practicing formative assessment for computational thinking in making environments. *Journal of Science Education and Technology, 29*(1), 162–173.

Hansen, A. K., Hansen, E. R., Dwyer, H. A., Harlow, D. B., & Franklin, D. (2016). Differentiating for diversity: Using universal design for learning in elementary computer science education. In *Proceedings of the 47th ACM technical symposium on computing science education* (pp. 376–381).

Harrison, A. G., & Treagust, D. F. (2000). A typology of school science models. *International Journal of Science Education, 22*(9), 1011–1026.

Hmelo-Silver, C. E., & Azevedo, R. (2006). Understanding complex systems: Some core challenges. *The Journal of the Learning Sciences, 15*(1), 53–61.

Hmelo-Silver, C. E., Duncan, R. G., & Chinn, C. A. (2007). Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark (2006). *Educational Psychologist, 42*(2), 99–107.

Hmelo-Silver, C. E., Jordan, R., Eberbach, C., Sinha, S. (2017). Systems learning with a conceptual representation: A quasi-experimental study. *Instructional Science,* 45(1), 53-72. https://doi.org/10.1007/s11251-016-9392-y

Hogan, K., & Thomas, D. (2001). Cognitive comparisons of students’ systems modeling in ecology. *Journal of Science Education and Technology, 10*(4), 319–345.

Jiménez-Aleixandre, M. P., Bugallo Rodríguez, A., & Duschl, R. A. (2000). “Doing the lesson” or “doing science”: Argument in high school genetics. *Science Education, 84*(6), 757–792.

Jonassen, D. H., & Hung, W. (2006). Learning to troubleshoot: A new theory-based design architecture. *Educational Psychology Review, 18*(1), 77–114.

Kafai, Y. B. (2005). The classroom as “living laboratory”: Design-based research for understanding, comparing, and evaluating learning science through design. *Educational Technology*, 28–34.

Karacalli, S., & Korur, F. (2014). The effects of project-based learning on students’ academic achievement, attitude, and retention of knowledge: The subject of “electricity in our lives.” *School Science and Mathematics, 114*(5), 224–235.

Katz, I. R., & Anderson, J. R. (1987). Debugging: An analysis of bug-location strategies. *Human-Computer Interaction, 3*(4), 351–399.

Kelly, G. J. (2013). Discourse in science classrooms. *Handbook of Research on Science Education*, 457–484.

King, A. (1998). Transactive peer tutoring: Distributing cognition and metacognition. *Educational Psychology Review, 10*(1), 57–74.

KMK [Sekretariat der StändigenKonferenz der Kultusminister der Länder in der BRD]. (2005a). BildungsstandardsimFachBiologie für den MittlerenSchulabschluss [Educational standards in biology for middle school graduation]. München/Neuwied, Germany: Wolters Kluwer. https://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2004/2004_12_16-Bildungsstandards-Biologie.pdf

KMK [Sekretariat der StändigenKonferenz der Kultusminister der Länder in der BRD]. (2005b). BildungsstandardsimFachChemiefür den MittlerenSchulabschluss [Educational standards in chemistry for middle school graduation]. Wolters Kluwer. https://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2004/2004_12_16-Bildungsstandards-Chemie.pdf

KMK [Sekretariat der StändigenKonferenz der Kultusminister der Länder in der BRD]. (2005c). BildungsstandardsimFach-Physik für den MittlerenSchulabschluss [Educational standards in physics for middle school graduation]. Wolters Kluwer. https://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2004/2004_12_16-Bildungsstandards-Physik-Mittleren-SA.pdf

Krahenbuhl, K. S. (2016). Student-centered education and constructivism: Challenges, concerns, and clarity for teachers. *The Clearing House: A Journal of Educational Strategies, Issues and Ideas, 89*(3), 97–105.

Krajcik, J., & Blumenfeld, P. (2006). Project-based learning. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 317–333). Cambridge University Press.

Krajcik, J. S., & Shin, N. (2022). Project-based learning. In R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences, (3rd ed).* Cambridge University Press.

Krell, M., Reinisch, B., & Krüger, D. (2015). Analyzing students' understanding of models and modeling referring to the disciplines biology, chemistry, and physics. *Research in Science Education, 45*(3), 367–393.

Krell, M., &Krüger, D. (2016). Testing models: a key aspect to promote teaching activities related to models and modelling in biology. *Journal of Biological Education, 50*(2).

Kyza, E. A., Constantinou, C. P., & Spanoudis, G. (2011). Sixth graders' co-construction of explanations of a disturbance in an ecosystem: Exploring relationships between grouping, reflective scaffolding, and evidence-based explanations. *International Journal of Science Education, 33*(18), 2489–2525.

Lederman, N. G. (2013). Nature of science: Past, present, and future. In *Handbook of research on science education* (pp. 845–894). Routledge.

Lee, I., & Malyn-Smith, J. (2020). Computational thinking integration patterns along the framework defining computational thinking from a disciplinary perspective. *Journal of Science Education and Technology, 29*(1), 9–18.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., ... & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*(1), 32–37.

Lee, S., Kang, E., & Kim, H. B. (2015). Exploring the impact of students' learning approach on collaborative group modeling of blood circulation. *Journal of Science Education and Technology, 24*(2), 234–255.

Lemke, J. (1990). *Talking science: Language, learning, and values*. Ablex.

Li, C., Chan, E., Denny, P., Luxton-Reilly, A., & Tempero, E. (2019). Towards a framework for teaching debugging. In *Proceedings of the Twenty-First Australasian Computing Education Conference* (pp. 79–86).

Louca, L. T., & Zacharia, Z. C. (2012). Modeling-based learning in science education: Cognitive, metacognitive, social, material and epistemological contributions. *Educational Review, 64*(4), 471–492.

Magana, A. J., & Silva Coutinho, G. (2017). Modeling and simulation practices for a computational thinking-enabled engineering workforce. *Computer Applications in Engineering Education, 25*(1), 62–78.

Martinez-Moyano, I. J., & Richardson, G. P. (2013). Best practices in system dynamics modeling. *System Dynamics Review, 29*(2), 102–123.

McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: A review of the literature from an educational perspective. *Computer Science Education, 18*(2), 67–92.

Meadows, D. (2008). *Thinking in systems: A primer.* Chelsea Green Publishing.

Mehan, H. (1979). *Learning lessons: Social organization in the classroom*. Harvard University Press.

Mittelstraß, J. (2005). AnmerkungenzumModellbegriff. In *Modelle des Denkens: Streitgespräch in der WissenschaftlichenSitzung der Versammlung der Berlin-Brandenburgischen Akademie der Wissenschaften*; Berlin-Brandenburgische Akademie der Wissenschaften.

National Research Council (NRC). (2007). *Taking science to school: Learning and teaching science in grades K-8*. National Academies Press.

National Research Council (NRC). (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.

NGGS Lead States. (2013). *Next generation science standards: For states, by states*. The National Academies Press.

Ogegbo, A. A., &Ramnarain, U. (2021). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 1–28.

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

Papert, S., & Harel, I. (1991). *Situating Constructionism. Constructionism, 36*(2), 1–11.

Pass, S. (2004). *Parallel paths to constructivism: Jean Piaget and Lev Vygotsky*. IAP.

Passmore, C., Gouvea, J. S., & Giere, R. (2014). Models in science and in learning science: Focusing scientific practice on sense-making. In *International handbook of research in history, philosophy and science teaching* (pp. 1171–1202). Springer.

Pierson, A. E., & Clark, D. B. (2018). Engaging students in computational modeling: The role of an external audience in shaping conceptual learning, model quality, and classroom discourse. *Science Education, 102*(6), 1336–1362.

Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science, 45*(5), 583–602.

Richardson, G. P. (1996). Problems for the future of system dynamics. *System Dynamics Review: The Journal of the System Dynamics Society, 12*(2), 141–157.

Riess, W., & Mischo, C. (2010). Promoting systems thinking through biology lessons. *International Journal of Science Education, 32*(6), 705–725.

Scanlon, E., Schreffler, J., James, W., Vasquez, E., & Chini, J. J. (2018). Postsecondary physics curricula and universal design for learning: Planning for diverse learners. *Physical Review Physics Education Research, 14*(2), 020101.

Schneider, B., Krajcik, J., Lavonen, J., Salmela-Aro, K., Broda, M., Spicer, J., Bruner, J., Moeller, J., Linnansaari, J., Juuti, K., &Viljaranta, J. (2016). Investigating optimal learning moments in U.S. and Finnish science classes. *Journal of Research in Science Teaching, 53*(3), 400–421.

Schneider, B., Krajcik, J., Lavonen, J., Salmela-Aro, K., Klager, C., Bradford, L., Chen, I. -C., Baker, Q., Touitou, I., & Peek-Brown, D. (2022). Improving science achievement—is it possible? Evaluating the efficacy of a high school chemistry and physics project-based learning intervention. *Educational Researcher, 51*(2), 109–121.

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

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: The Official Journal of the National Association for Research in Science Teaching*, *46*(6), 632–654.

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

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.

Shen, J., Lei, J., Chang, H. Y., &Namdar, B. (2014). Technology-enhanced, modeling-based instruction (TMBI) in science education. In *Handbook of research on educational communications and technology* (pp. 529–540). Springer.

Shin, N., Bowers, J., Krajcik, J., & Damelin, D. (2021). Promoting computational thinking through project-based learning. *Disciplinary and Interdisciplinary Science Education Research, 3*, 7.

Shin, N., Bowers, J., Roderick, S., Mclntyre, C., Stephens, L., Eidin, E., Krajcik, J., & Damelin, D. (2022). A framework for supporting systems thinking and computational thinking through constructing modeling. *Instructional Science, 50*(6), 933–960.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*(1), 142–158.

Sins, P. H., Savelsbergh, E. R., & van Joolingen, W. R. (2005). The difficult process of scientific modelling: An analysis of novices' reasoning during computer-based modelling. *International Journal of Science Education, 27*(14), 1695–1721.

Song, J., Kang, S. J., Kwak, Y., Kim, D., Kim, S., Na, J., ... & Joung, Y. J. (2019). Contents and features of 'Korean Science Education Standards (KSES)' for the next generation. *Journal of the Korean Association for Science Education*, *39*(3), 465–478.

Stave, K. A. (2002). Using system dynamics to improve public participation in environmental decisions. *System Dynamics Review: The Journal of the System Dynamics Society, 18*(2), 139–167.

Stratford, S. J., Krajcik, J., & Soloway, E. (1998). Secondary students' dynamic modeling processes: Analyzing, reasoning about, synthesizing, and testing models of stream ecosystems. *Journal of Science Education and Technology, 7*, 215–234.

Sterman, J. D. (1994). Learning in and about complex systems. *System Dynamics Review, 10*(2–3), 291–330.

Sterman, J. D., & Sweeney, L. B. (2002). Cloudy skies: Assessing public understanding of global warming. *System Dynamics Review: The Journal of the System Dynamics Society, 18*(2), 207–240.

Sullivan, F. R., & Heffernan, J. (2016). Robotic construction kits as computational manipulatives for learning in the STEM disciplines. *Journal of Research on Technology in Education, 48*(2), 105–128.

Swanson, H., Sherin, B., & Wilensky, U. (2021). Refining student thinking through computational modeling. In *Proceedings of the 15th International Conference of the Learning Sciences-ICLS 2021*. International Society of the Learning Sciences.

Türker, P. M., & Pala, F. K. (2020). The effect of algorithm education on students' computer programming self-efficacy perceptions and computational thinking skills. *International Journal of Computer Science Education in Schools, 3*(3), 19–32. https://doi.org/10.21585/ijcses.v3i3.69

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*(1), 127–147.

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.

Windschitl, M., Thompson, J., & Braaten, M. (2020). *Ambitious science teaching*. Harvard Education Press.

Wurdinger, S., Haar, J., Hugg, R., & Bezon, J. (2007). A qualitative study using project-based learning in a mainstream middle school. *Improving Schools, 10*(2), 150–161.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35.

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

Zhang, L., VanLehn, K., Girard, S., Burleson, W., Chavez-Echeagaray, M. E., Gonzalez-Sanchez, J., & Hidalgo-Pontet, Y. (2014). Evaluation of a meta-tutor for constructing models of dynamic systems. *Computers & Education, 75*, 196–217.

Zhang, N., Biswas, G., McElhaney, K. W., Basu, S., McBride, E., & Chiu, J. L. (2020). Studying the interactions between science, engineering, and computational thinking in a learning-by-modeling environment. Artificial intelligence in education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020.