International Society of the Learning Sciences

Designing for Discourse: Enacting Responsive Teaching through *Code Reflections* in a Computational Modeling Unit

Aditi Wagh, Massachusetts Institute of Technology, awagh@mit.edu
Leah F. Rosenbaum, Teachers College, Columbia University, leah@tltlab.org
Adelmo Eloy, Teachers College, Columbia University, adelmo@fablearn.net
Tamar Fuhrmann, Teachers College, Columbia University, tamarrf@gmail.com
Brendan Henrique, University of California, Berkeley, bhenrique@berkeley.edu
Paulo Blikstein, Columbia University & Instituto Tecnológico de Monterrey paulob@tc.columbia.edu
Michelle Wilkerson, University of California, Berkeley, mwilkers@berkeley.edu

Abstract: An emerging body of work in the learning sciences has examined how computational models can support teachers in responding to students' prompts, inquiry, and ideas. This work has highlighted how teachers make discursive moves in relation to computational models to support classroom discussion. In this paper, we focus on a complementary phenomenon: teachers' design of *code reflections*, or curricular modifications that deepen students' engagement with one another's code for scientific and computational sensemaking. We highlight how these code reflections advanced student discourse and how both the code reflections and discourse became more sophisticated over time, shifting towards making connections across code, behaviors, simulation outcomes, data and the scientific process being represented. We reflect on how this progression was driven by shifts in the teachers' comfort with code and computational modeling and the resources designers can offer to educators to support the development of code reflections.

Introduction

Computational modeling can help students articulate and elaborate rich, diverse ideas about how the world works (e.g., Papert, 1980; White, 1984; Wilensky & Reisman, 2006; Sengupta et al., 2021). Teachers can play an important role in leveraging students' ideas in the context of computational modeling. Indeed, recent work has documented how teachers respond to student ideas through discourse moves to help students articulate and refine their ideas using computational models (Wagh et al., 2024; Swanson et al., 2024). In this paper, we examine a complementary phenomenon: teachers' response to student ideas as expressed in code, through curricular adaptations that we call *code reflections* that are designed to engage students with the code of each other's computational models for scientific and computational sensemaking.

We draw from a design-based research project (Barab & Squire, 2004) in which we developed an agent-based modeling platform, MoDa (e.g., Fuhrmann et al., 2024; Wagh et al., 2022), that enables students to program their own models and compare them, side by side, with real-world data. The platform builds on research on domain-specific modeling environments (e.g., Kahn, 2007; Wilkerson et al., 2015) as well as work on linking real-world data with computational modeling (e.g., Blikstein, 2014; Fuhrmann et al, 2014). We draw on data from an experienced teacher's sixth grade science class in a public school on the West Coast of the US. We examine how this teacher developed curricular supports which we call *code reflections*, in response to students' ideas and models in each unit, and the kinds of whole-class discourse each code reflection prompted. By "code reflection", we mean an activity in which students closely examine code and its output as artifacts for computational and/or scientific sensemaking - in our case, reading, predicting and critiquing code. Specifically, we ask: 1. How did an experienced teacher respond to student ideas and models through the design of new activities in each unit?; and, 2. How did the nature of discourse supported by each activity shift over time? Drawing on thematic analysis (Braun & Clarke, 2012) of video data from whole-class discussions following each code reflection, teacher artifacts and interviews, and observation notes of class implementations, we illustrate how each code reflection responded to the substance of students' work and supported progressively more complex discourse that included connections across the available representations.

Conceptual background

Responsive teaching and curricular adaptations

Responsive teaching is a pedagogical approach that strives to pursue student thinking to support learning (e.g., Levin et al., 2013; Robertson et al., 2016). In general, responsive teaching involves three key ideas: 1. Highlighting the substance of students' ideas; 2. Recognizing disciplinary connections in student ideas; and, 3. Taking up and

pursuing the substance of students' ideas (Robertson et al., 2016). Robertson et al (2020) noted a gap in the literature on responsive teaching and curricular adaptations. They argued that studies on responsive teaching have primarily emphasized how teachers shape classroom instruction in response to student ideas, without an explicit consideration of the role of curricular materials in informing their decisions. On the other hand, studies of teachers' curricular adaptations have not attended to the role that students' ideas can play in motivating and guiding these adaptations. Robertson and colleagues' study (2020) brought these two bodies of work together to show how an awareness and understanding of the purpose of questions in curricula enabled teachers to be responsive to students' ideas in classroom instruction.

Prior work shows promise in the use of computational modeling to be responsive to students' ideas. For instance, proposing *responsivity* as a feature of modeling environments, Wilkerson and colleagues (2015) proposed that educators can be responsive to student ideas by modifying various aspects of the environment to enable students to pursue their own questions and ideas. This work located an educator being responsive to student ideas through the design of the tool itself. Others have studied how computational models can provide a generative context for teachers to respond to students' ideas through discourse (Swanson et al., 2024).

Computational modeling and whole-class discourse

Programming and revising computational models has gained increasing traction for how it integrates scientific learning with computational thinking practices (e.g., Sengupta et al., 2013; Weintrop et al., 2017). There has been some work on the role computational modeling can play in whole-class discourse. Overall, this work has suggested that computational modeling can play a generative role when used at the level of the whole class. For instance, Wilkerson and colleagues (2017) documented how it provided an infrastructure to support classroom-level knowledge building as different student ideas and proposals were negotiated and coordinated into a model. Others have studied how teachers can calibrate and help align students' varied ideas with the representational system of a modeling tool (Wagh et al., 2023) and balance the computational and scientific aspects of modeling through specific moves (Wagh et al., 2024). Though not involving students programming models, some work has shown how pre-built computer models served as a site for argumentation (Berland & Reiser, 2009), and that there can be variation in productive teacher roles (Hmelo-Silver et al., 2015). More recently, Swanson and colleagues (2024) showed how computational models can play a role in responsive teaching by enabling a teacher to articulate and examine students' ideas through whole class discourse.

Methods

We position this study as bridging the two reviewed lines of research. We focus both on the ways that our focal teacher's code reflections were responsive to student ideas within a computational modeling unit and on the kinds of whole-class discourse those reflections prompted. Specifically, we pursue the following questions:

1. How did an experienced teacher respond to student ideas and models through the design of new activities in each unit?; and,
2. How did the nature of discourse supported by each activity shift over time?

Context: Sixth grade science units

This paper draws on a focal grade 6 classroom across three science units designed using MoDa (Table 1). MoDa is an agent-based modeling platform that enables students to program agent-based models using domain-specific blocks and compare their models with real-world data. In Unit 1, students explored the phenomenon of ink spreading in water. Students collected and analyzed data from an experiment with dye color spreading in hot and cold water, drew paper models to explain the phenomenon, and iteratively programmed models to represent their theories, refining their models in comparison with experimental videos. In Unit 2, students explored the phenomenon of wildfire smoke spread. Students analyzed real-world satellite video data from a wildfire in CA to identify patterns in wildfire smoke shape, found variables that impact the spread of smoke, and iteratively programmed a MoDa model to match videos of low and strong wind conditions. Finally, in Unit 3, students examined the production of glucose in photosynthesis. They collected and analyzed data from an experiment with the Elodea plant, created physical models with ping-pong balls, and iteratively programmed a MoDa model to represent photosynthesis, matching the elements and ratio in the chemical equation for photosynthesis.

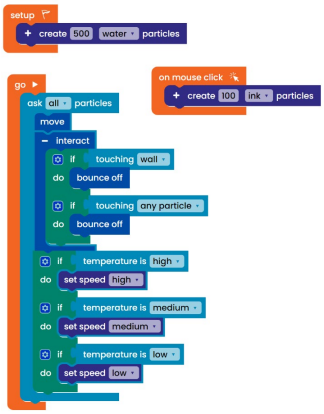
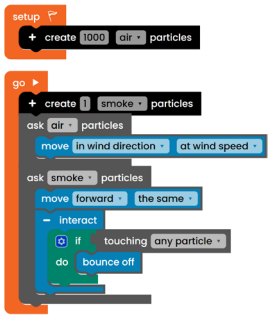
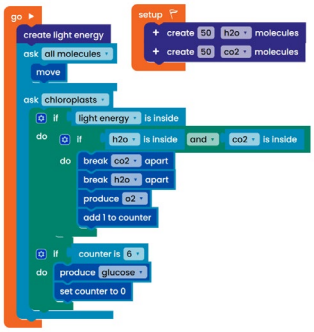
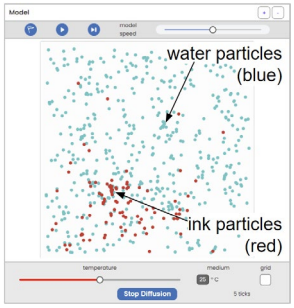
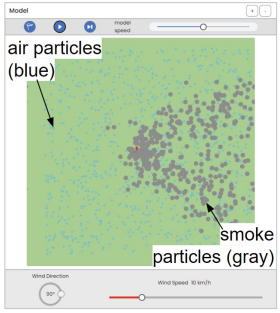
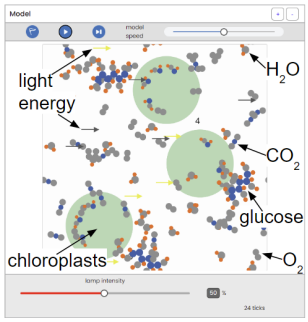
Two researchers/authors were present in the classroom during each unit implementation, primarily to provide technical support with MoDa, where needed, and to collect research data. These researchers also worked with Ms. K, the focal teacher, to prepare for and debrief from lessons, including collaboratively revising researcher-provided curricular materials.

Data collection

Data comes from three units taught a few months apart by Ms K. in a six-grade science class. Ms. K is a long-term collaborator with over a decade of experience using NetLogo models for science exploration and three years of experience guiding students in building MoDa models. She co-designed each unit with the research team and attended all professional development sessions offered (about 50 hours). This was her first year implementing three MoDa units in an academic year. Data sources consist of approximately 3 hours of video footage capturing whole-class discussions across the three units, three 45-minute teacher interviews following each unit, and the code reflections for each unit, which are detailed in the Findings and Figure 1.

Table 1

Summary of the Three Units in MoDa Used in This Study.

	U1: Ink spread in water	U2: Wildfire smoke spread	U3: Photosynthesis
Sample Code			
Sample Simulation			

Analysis

Analysis began when the authors noted the ways that Ms K responded to students' ideas and work by designing activities to enable students to reflect on each others' code for scientific and computational sensemaking. We call these designed activities *code reflections.* Using thematic analysis (Braun & Clarke, 2012), we also noted that each code reflection was unique and - by comparing the form of the code reflection to the student work that preceded it and our first-hand knowledge of the classroom activities - addressed very specific aspects of what students were doing or struggling with in class. We triangulated our interpretations with Ms. K's own notes and interview excerpts about her motivations behind the code reflections and the roles she designed them to play. Across this thematic analysis of artifacts and her interviews (Braun & Clarke, 2012), we looked for confirmatory evidence that the code reflection indeed responded to the substance of students' ideas and models.

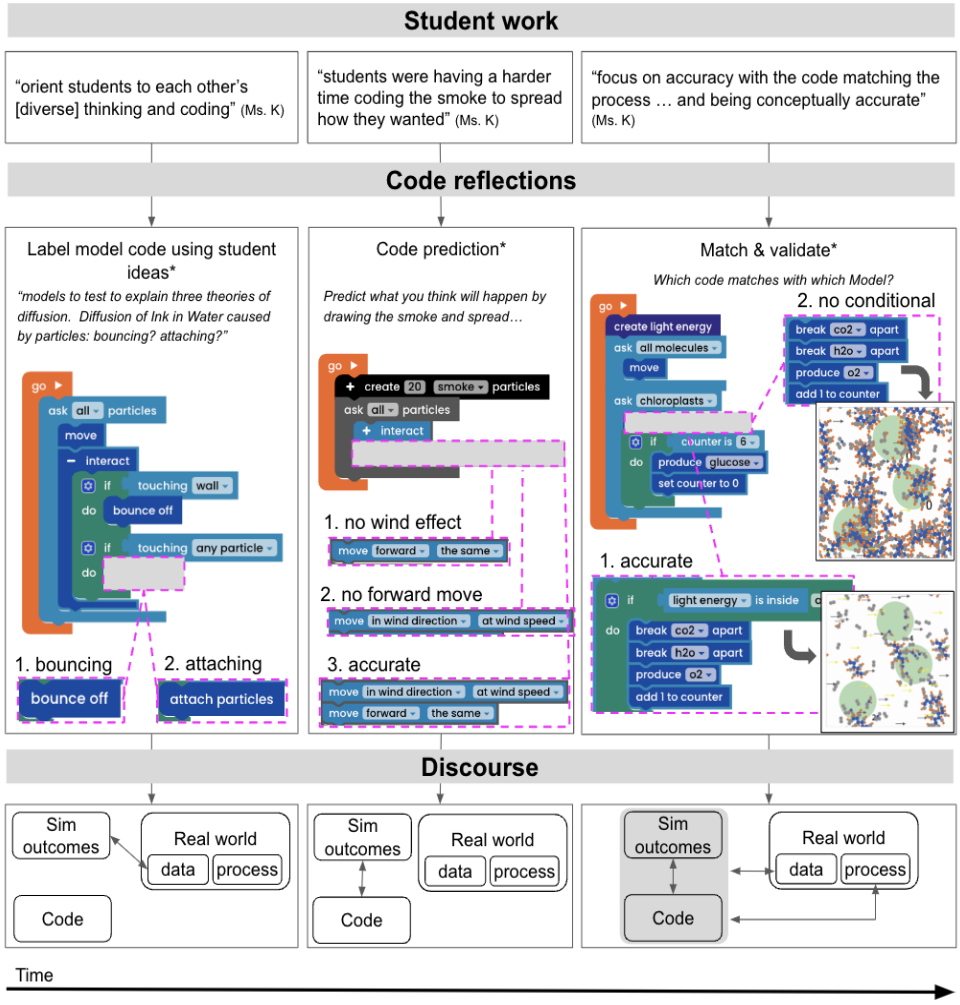
For each code reflection, we transcribed the discussion that happened immediately after it. Collectively viewing these videos and corresponding transcripts led us to notice that students were making different kinds of epistemic connections in each of the discussions. We identified the three epistemic representations (code, simulation, and data) and the target scientific process (e.g., diffusion) present in each unit. For each discussion, we identified which epistemic representations and elements of the scientific process featured most prominently in the students' and Ms K's discussion - that is, which representation(s) and scientific consideration(s) were most overtly linked and the ways those links built understanding of each constituent element.

Findings

Our analysis revealed that as Ms K developed code reflections to respond to students' ideas and work in each unit, subsequent class discourse around computational models grew more complex (See Figure 1). Specifically, in each unit, Ms K developed a unique code reflection in response to what she saw in students' ideas and work as students built computational models of the target phenomena (Figure 1). Each code reflection resulted in specific patterns in the resulting whole-class discourse in terms of the connections students made across the representations and resources available in the unit. Looking across the three units, we found that the resulting whole class discourse grew progressively more complex with each innovation. In other words, with each code reflection in each successive unit, whole class discourse shifted towards students making connections across the representations and resources within the modeling platform and unit (See Figure 1).

Figure 1

Code Reflections Developed by Ms. K Over Time, as a Response to Student Work, and Influence on Subsequent Class Discourse Only Select Models Shown for Space.



To illustrate this shift in the nature of the code reflection and the discourse it supported, we present the code reflection and following discourse. For each unit, we first describe the innovation in terms of its form and specific moves Ms K made to support students working with it. We also present excerpts from the discourse to illustrate the kinds of connections students made across the available representations and resources.

Unit 1: Labeling students' code using their ideas linked simulation outcomes to data

Code reflection: Labeling students' model code using student ideas

The code reflection in the ink spread unit enabled students to orient to each others' ideas and model code. When students proposed initial explanations for how and why ink spread in water, they came up with three ideas: ink

and water particles "attach" to each other, particles "bounce off" each other, and hot water has "more space between particles" making the ink spread more quickly. Students translated their initial ideas, represented in paper models, into model code in MoDa. The designed code reflection (Figure 1) was the teacher's way of "orienting" students to each others' models and code so students could be aware of each others' ideas and how they can be translated into code. In the post-interview, Ms K said that she was "trying to help kids make a match or connection between the theory. If you think particles attach? Then this is … the type of code that would lead to that and what are the blocks that would show attach." She said that she was "trying to help them make connections between their scientific ideas and the language." Students' ideas became labels that the class attached to the different models and model code. Rather than a one-time use, these labels became a classroom convention for how Ms K and her students described their work and thinking (e.g., how students referred to each others' models or the names students gave to their model files).

Whole-class discourse: Using labels to link simulation outcomes with data

In this class, these labels became a reference point to tag different models based primarily on their simulation outcomes. In the resulting discourse, Ms K and her students validated student observations about the experiment and the real world phenomenon with simulation outcomes from each of the model types.

Ms K initiated the whole-class discussion by telling students that "our goal here is to compare the model with the real world next to it and see if it seems to be matching" and asking them to share their thinking. Student pairs were invited to share their models. The second pair to share were Freida and Anna who had chosen to represent the attach theory. Freida and Anna's model was displayed on the projector for the class as the simulation ran. Both the program code and running simulation were visible to students. When Ms K asked the class what they noticed happening in the model, a student shared an observation from the simulation that "the ink is attaching." Ms K asked, "Does that seem to be what's happening in the real world?" Freida and Anna shared that if they were to remake this model, they would add more water to it so there was more "opportunity for ink to latch on." Ms K then opened the discussion to the class, and students argued for or against the attach theory. As they did so, they seemed to primarily compare their observations of the running simulation with their observations of the real world phenomenon or experiment. For instance:

> Calen: Well a few things for why the attachment theory I believe is correct. […] Well, one, if the dye was simply bouncing off the water molecules, yes, they would turn that color, but only in certain places where, because the dye bounce off of each other, so those places would turn into the dye color only for one second, because the particles would then leave that area.
>
> *(A couple of other students share)*
>
> Arthur: I also wanna say that it also kind of does simulate the real thing because water starts out clear and then the ink takes over and turns the water into the ink's color, which is also the attachment theory.

Both students in this excerpt argued for the attach theory based on perceived visual similarity between the model simulation and the real-world video data. Calen observed the running simulation and he argued that the bounce off mechanism would produce temporary pockets of color that bounced to other areas of the glass, which he did not see in the video. Arthur argued that the simulation resulting from Freida and Anna's attachment model looks similar to video data of the ink spread experiment, with "the ink tak[in] over and turn[ing] the water into the ink's color." Both students drew on a comparison between the data from the real world phenomenon (i.e., their observations of what the spread of dye looked like in their experiment) with the running simulation in MoDa. This code reflection of having students label programs with corresponding theories to orient to each others' thinking made it possible for students to be familiar enough with each others' ideas so they could argue for or against them. In other words, students came to recognize specific code as a (new) representation of their and their classmates' ideas.

U2: Code prediction helped link code and simulation outcomes

Code reflection activity: Drawing a prediction of a program's simulation

The code reflection in the smoke spread unit aimed to support students to read and interpret code to predict how it would play out in the simulation. Our observation notes suggested that students struggled with the idea that the code produces the simulation and that even a small change to the code can impact the simulation outcome significantly. In response, Ms K gave students four sets of program code with a screenshot of a blank simulation for each. These programs were either ones that students in the class had generated or included specific code-blocks

or arrangements of code-blocks that students had found to be confusing or challenging. Students were asked to read and interpret the program and draw their prediction of the corresponding movement of smoke particles and the pattern the program would create. In her post interview, Ms K shared that she developed this activity because "students were having a harder time coding the smoke to spread how they wanted." Her goal was to "support students to be able to accurately read code for a desired outcome to hone their code reading and coding skills." In other words, Ms. K wanted students to learn how to read and interpret how the program code for smoke particles would work.

When she facilitated the activity in class, Ms. K made specific moves to make visible student ideas and thinking and how that compared with the outcomes in the computational model. In particular, she carefully arranged materials to draw out contrasts or alignments between students' ideas and the code to help them identify these relationships. For instance, she projected students' drawn predictions on a screen on which she also displayed the MoDa simulation outcome so the two could be visually compared and contrasted. She also had students share their thinking in real-time in front of and with support from their peers as a way to build on their ideas and help them identify the role of these code blocks and how they impacted simulation outcomes.

Whole-class Discourse: Linking code and simulation outcomes

In the whole-class discourse that followed the code prediction activity, Ms K and students inferred and explained how specific program code and the blocks in it would work individually and together. For each of the four programs, Ms K invited student pairs to come to the front of the class to draw out and explain their predictions about the kind of smoke pattern the projected program code would yield.

Silas:	Well, I was just thinking it is slightly different than the other one, it says "move in the wind direction" at first and then it also says "move forward at the same time". And so I was thinking it will make a cone, like that. Like spreading out.
Ms K:	Spreading out because of that "move forward" thing *(stretches her two arms away from each other)*. Alright, should we see? *(Displays the actual pattern produced by the code)*
Student:	In the wind direction and then it chooses its interact…
Ms K:	What do we notice? Is it thicker than this one *(goes back to the previous one)*
Students:	Yeahh.
Ms K:	*(returns to this one)* So what blocks do you think are making it thicker?
Student:	The "move forward the same".
Student:	The wind.
Ms K:	The "move forward". Yeah, so somehow there is this combination of "move forward at wind direction."

The discourse in response to Ms. K's code prediction activity focused - as intended - on how specific code blocks ("move in the wind direction," "move forward at the same time") cause specific simulation outcomes ("it will make a cone," "spreading out"). To emphasize this causal relationship between code and simulation, Ms. K visually contrasted the focal simulation with that of a previous model and prompted her students to identify the code responsible for that difference ("what blocks do you think are making it thicker?"). When her students name two different code blocks ("move forward the same," "the wind"), Ms. K emphasized that code blocks might work together ("somehow there is this combination") to produce an observed simulation outcome. By highlighting patterns in simulations and code - and differences in those patterns across models - the code prediction activity helped students identify how the code blocks worked (individually and in concert) to produce specific agent-level actions and overall simulation outcomes.

In contrast to the discourse after the labeling activity, the class responded to the prediction activity by temporarily putting aside the data to focus on the interaction between code and simulation (Figure 1), regardless of each model's real-world accuracy. Whereas this may at first appear to be a tangent in the overall development of whole class discourse, the next code reflection prompted the class to unite both discursive practices, resulting in discourse that attended to all the epistemic representations the modeling platform and unit offered.

U3: Validating simulation and code linked code, simulation and data to an equation

Code reflection activity: Validate simulation and code

Finally, in the photosynthesis unit, the code reflection involved students critique whether or not a simulation represented photosynthesis, match the simulation with its corresponding program code, and then critique program

code for valid representation of the scientific process. The rationale for this design was that "Photosynthesis is a complex process. Because of this, naturally, coding to demonstrate the process is also complex. Because most students were creating code that produced glucose molecules, the next step was to focus on accuracy with the code matching the process more clearly and being conceptually accurate."

As in the code reflection in the smoke spread unit, Ms K selected intentionally contrasting cases from among student work to emphasize key relationships between the code and the simulation and how much they reflected (or not) the scientific process of photosynthesis as represented in its equation. In her selection of models to include in this set, she again included four examples that were a response to the student thinking and challenges that were visible in class. For example, one program was in response to students having an inoperable counter in their code, and two programs did not include the conditions for photosynthesis, resulting in an overproduction of glucose. Our observation notes indicate that these were all challenges that were visible in student work on the previous day. The code reflection served to elevate and make visible these challenges so the class could collectively make sense of them.

Here again, the teacher made specific design choices that served to highlight student ideas as well as support students to engage in the challenging task of mapping an equation with program code and a simulation.

Whole-class discourse: Linking code, simulation, data to an equation of photosynthesis

The match and validate design enabled students to connect between the program code and resulting simulation outcomes, connect those to the scientific process of photosynthesis, and examine how code would need to be revised to represent the process. Ms K initiated the discussion by telling the class "We're gonna look at 4 models running as we look at the 4 models, you're gonna to be thinking about, does this model show photosynthesis, does it not show photosynthesis, or kind of, but there is a problem." She opened up a pre-selected student model but deliberately hid the code so only the simulation was visible to the class. She reset the simulation three times to slow it down enough for students to see the simulation. Right away, several students chimed into the discussion - some students thought the plant wasn't producing glucose while others thought it was. One student chimed, "yes, it is making glucose, but it is making it automatically." This led to the following discussion:

Vera:	I don't think it shows photosynthesis because the light, first there is no ratio.
St:	Yes.
Vera:	And the light isn't involved in making glucose [Ms K starts writing on the board]. It is just like making glucose. I think whenever there is like, whenever oxygen and, stored, yes. And just like.
Ms K:	Do you mean carbon dioxide, or oxygen?
Vera:	Yeah, carbon dioxide.
Ms K:	Ok, can somebody build on or add to what Vera said? I agree, I don't think this models photosynthesis. Or I disagree, I think. Howard.
Howard:	I agree with Vera because I think they are not just producing glucose under Go, or something. Nothing is really breaking apart, and just think the glucose is forming without a ratio. *(A minute later)*
St:	Every tick it creates glucose. *(A minute later)*
Arthur:	I think there is actually, I think there is a line of code that says if H2O and carbon dioxide is inside, create glucose. But instead, it is constant creation of the glucose, whenever the molecules are inside.

The code reflection prompted a whole-class discussion that included available epistemic considerations and validated them against the scientific process. As they did in the code prediction activity, students identified the particular code blocks and structures that cause particular simulation outcomes ("every tick it creates glucose," "there is a line of code that says if H2O and carbon dioxide is inside, create glucose … it is constant creation of the glucose, whenever the molecules are inside"). Students also compared the code to their understanding of the scientific process of photosynthesis ("there is no ratio," "the light isn't involved," "Nothing is really breaking apart, and … the glucose is forming without a ratio."). Finally, they used both these comparisons together to make judgements about the model's validity (the code + simulation) compared to their understanding of the process ("I don't think it shows photosynthesis," "I agree"). Compared to the discourse in response to the first two innovations (Figure 1), this discussion demonstrates remarkable sophistication in students' simultaneous uses of and reasoning about the code, simulation, and scientific process in ways that reflect and build upon one another. The code reflection of having students validate the code and simulation with the scientific process represented by the equation made space for this sophisticated discourse.

Discussion

We presented work done by an experienced sixth-grade science teacher as she responded to students' ideas and models by designing *code reflections* to advance students' work and discourse. We also illustrated how these code reflections impacted whole-class discourse and supported students to make different epistemic connections across the representations and resources available in the three units (code, behaviors, simulation outcomes, data and the scientific process being represented; Figure 1).

This paper contributes to the literature on computational modeling and responsive teaching in a few ways. Much of the research on responsive teaching has studied teachers' responsiveness to students articulating ideas in-the-moment (e.g., Robertson et al., 2016) including with computational models (Swanson et al., 2024). Here, the teacher responded to students' ideas as expressed in their model code through code reflections to have them engage with each others' code for scientific and computational sensemaking. Though we see the pedagogical decision to showcase multiple ideas itself as a responsive move, the three code reflections manifested different degrees of responsiveness. The code labeling functioned to respond by making the ideas visible and durable so that other students could engage with them and the prediction activity served responsiveness by drawing out multiple code interpretations. However the validation reflection largely focused on conceptual accuracy. In the interview, this was a tension the teacher acknowledged - "Yeah, that's my only complaint with the [photosynthesis] modeling. It just felt. You know, it just converged to this one model." While the code reflection was designed by the teacher, its design was driven (and constrained) by the design of the code block library available in the unit.

We would also argue that our work motivates researchers to be responsive to teachers' existing strengths and practices as we bring new tools and practices into their classrooms. Ms K leveraged her expertise in more traditional lesson design as she adopted this new representational practice into her classroom practice. That is, though the prompts included code from students' MoDa models, the reflections took the form of paper worksheets and images projected on the class projector - more traditional forms typical in classrooms.

Finally, the code reflections both directly informed classroom discourse and served as a mechanism for the shift in discourse over time. The code reflections were developed *in response to* students' ideas and artifacts. Directly responding to students' models paved the way for the class to advance in their work and thinking. This shift also corresponded to a shift in Ms K's growing confidence and competence to develop such designs and effectively facilitate whole-class discourse around them. Just as the literature on responsive teaching calls for teacher professional development to listen for and respond to the substance of students' disciplinary ideas (e.g., Watkins et al., 2020; Ajilore et al., 2024), the work we shared here highlights the importance of supporting teachers to attend to and note the computational and scientific substance of students' ideas. In Ms K's case, this included both code-related learning (e.g., students struggling with understanding the order of execution) but also scientific ideas (e.g., how do you model to avoid overproduction of glucose). Preparing teachers to attend to these ideas and respond to them through code reflections and whole-class discourse will more broadly support teachers to adopt responsive teaching pedagogies through computational modeling.

While our paper makes contributions to the field, it comes with some limitations. Ms K's responsiveness to student ideas also played itself out in the ways in which she facilitated the whole class discourse. While we have described her pedagogical moves elsewhere (Wagh et al., 2024), here, we did not account for the ways in which these moves contributed to the shift in the epistemic connections made by students across units. In future work, we aim to examine this more closely and study how this shift co-occurred with the teachers' growing comfort and competence with computational modeling.

References

Ajilore, O., Appleby, L., Gouvea, J., Tobin, R., & Hammer, D. (2024). "I'm with Peppa": University STEM Faculty Listening to Student Reasoning. In Lindgren, R., Asino, T. I., Kyza, E. A., Looi, C. K., Keifert, D. T., & Suárez, E. (Eds.), *Proceedings of the 18th International Conference of the Learning Sciences - ICLS 2024* (pp. 1618-1621). International Society of the Learning Sciences.

Barab, S. A. , & Squire , K. (2004). Design- based research: Putting a stake in the ground . *The Journal of the Learning Sciences , 13*(1), 1 – 14. https://doi.org/10.1207/s15327809jls1301_1.

Berland, L. K., & Reiser, B. J. (2009). Making sense of argumentation and explanation. *Science Education*, *93*(1), 26–55. https://doi.org/10.1002/sce.20286

Blikstein, P. (2014). Bifocal modeling: Comparing physical and computational models linked in real time. In A. Nijholt (Ed.), *Playful learning interfaces* (pp. 317–352). Springer. https://doi.org/10.1007/978-981-4560-96-2_15

Braun, V., & Clarke, V. (2012). Thematic analysis. In H. Cooper, P. M. Camic, D. L. Long, A. T. Panter, D. Rindskopf, & K. J. Sher (Eds.), *APA handbook of research methods in psychology, Vol. 2. Research*

designs: Quantitative, qualitative, neuropsychological, and biological (pp. 57–71). American Psychological Association. https://doi.org/10.1037/13620-004

Fuhrmann, T., Salehi, S., & Blikstein, P. (2014). *A Tale of Two Worlds: Using Bifocal Modeling to Find and Resolve "Discrepant Events" Between Physical Experiments and Virtual Models in Biology*. https://repository.isls.org//handle/1/1206

Hmelo-Silver, C. E., Liu, L., Gray, S., & Jordan, R. (2015). Using representational tools to learn about complex systems: A tale of two classrooms. *Journal of Research in Science Teaching*, *52*(1), 6–35.

Levin, D., D. Hammer, A. Elby, and J. Coffey. 2013. *Becoming a Responsive Science Teacher*. Arlington: NSTA Press.

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

Pierson, A. E., Brady, C. E., & Clark, D. B. (2020). Balancing the environment: Computational models as interactive participants in a STEM classroom. *Journal of Science Education and Technology*, *29*, 101-119. https://doi.org/10.1007/s10956-019-09797-5.

Robertson, A. D., Atkins, L. J., & Levin, D. N. (2016). What is responsive teaching? In A. D. Robertson, R. E. Scherr, & D. Hammer (Eds.), *Responsive teaching in science and mathematics* (pp. 1–36). Routledge.

Robertson, A. D., Gray, K. E., Lovegren, C. E., Killough, K. L., & Wenzinger, S. T. (2020). Curricular Knowledge as a Resource for Responsive Instruction: A Case Study. *Cognition and Instruction*, *39*(2), 149–180.

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

Sengupta, P., Dickes, A., & Farris, A. V. (2021). *Voicing Code in STEM: A Dialogical Imagination*. MIT Press.

Swanson, H., Lawrence, L., Arnell, J., Jones, B., Sherin, B., & Wilensky, U. (2024). Computational models as tools for supporting responsive teaching. *Behaviour & Information Technology*, 1-18.

Wagh, A., Fuhrmann, T., Eloy, A. A. D. S., Wolf, J., Bumbacher, E., Blikstein, P., & Wilkerson, M. (2022, June). MoDa: Designing a tool to interweave computational modeling with real-world data analysis for science learning in middle school. In Proceedings of the 21st Annual ACM Interaction Design and Children Conference (pp. 206-211). https://doi.org/10.1145/3501712.3529723.

Wagh, A., Rosenbaum, L. F., Fuhrmann, T., Eloy, A., Blikstein, P., & Wilkerson, M. (2024). Toward Ontological Alignment: Coordinating Student Ideas with the Representational System of a Computational Modeling Unit for Science Learning. Cognition and Instruction, 1-32.

Wagh, A., Rosenbaum, L. F., Fuhrmann, T., Eloy, A., Bumbacher, E., Blikstein, P., & Wilkerson, M. (2023). Ontological alignment: Investigating the role of the teacher in supporting computational modeling in science classrooms. In Blikstein, P., Van Aalst, J., Kizito, R., & Brennan, K. (Eds.), Proceedings of the 17th International Conference of the Learning Sciences - ICLS 2023 (pp. 1162-1165). International Society of the Learning Sciences. https://doi.org/10.22318/icls2023.573415.

Watkins, J., Jaber, L. Z., & Dini, V. (2020). Facilitating Scientific Engagement Online: Responsive Teaching in a Science Professional Development Program. *Journal of Science Teacher Education*, *31*(5), 515–536.

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*, 127-147. https://doi.org/10.1007/s10956-015-9581-5.

Wilkerson, M., Shareff, B., Gravel, B., Shaban, Y., & Laina, V. (2017). Exploring Computational Modeling Environments as Tools to Structure Classroom-Level Knowledge Building In Smith, B. K., Borge, M., Mercier, E., and Lim, K. Y. (Eds.). (2017). *Making a Difference: Prioritizing Equity and Access in CSCL, 12th International Conference on Computer Supported Collaborative Learning (CSCL) 2017*, Volume 1. Philadelphia, PA: International Society of the Learning Sciences. https://repository.isls.org//handle/1/264.

Wilkerson, M., Wagh, A., & Wilensky, U. (2015). Balancing curricular and pedagogical needs in computational construction kits: Lessons from the DeltaTick project. *Science Education, 99*(3), 465-499.

Acknowledgments

This study is supported by funding from the National Science Foundation under Grant No. 2010413. We thank Ms. K for using MoDa in her classroom. We also thank Julia Gouvea for her thoughtful comments on the paper.