Kong, S.C., Hoppe, H.U., Hsu, T.C., Huang, R.H., Kuo, B.C., Li, K.Y., Looi, C.K., Milrad, M., Shih, J.L., Sin, K.F., Song, K.S., Specht, M., Sullivan, F., & Vahrenhold, J. (Eds.). (2020). *Proceedings of International Conference on Computational Thinking Education 2020*. Hong Kong: The Education University of Hong Kong.

Workshops and Co-design Can Help Teachers Integrate Computational Thinking into Their K-12 STEM Classes

Sally P. W. WU[1*], Amanda PEEL[2], Connor BAIN[3], Gabriella ANTON[4], Michael HORN[5], Uri WILENSKY[6]
[1,2,3,4,5,6]Northwestern University, United States
sally.wu@northwestern.edu, amanda.peel@northwestern.edu, connorbain2015@u.northwestern.edu, gabriellaanton3.2020@u.northwestern.edu, michael-horn@northwestern.edu, uri@northwestern.edu

ABSTRACT
This work aims to help high school STEM teachers integrate computational thinking (CT) into their classrooms by engaging teachers as curriculum co-designers. K-12 teachers who are not trained in computer science may not see the value of CT in STEM classrooms and how to engage their students in computational practices that reflect the practices of STEM professionals. To this end, we developed a 4-week professional development workshop for eight science and mathematics high school teachers to co-design computationally enhanced curriculum with our team of researchers. The workshop first provided an introduction to computational practices and tools for STEM education. Then, teachers engaged in co-design to enhance their science and mathematics curricula with computational practices in STEM. Data from surveys and interviews showed that teachers learned about computational thinking, computational tools, coding, and the value of collaboration after the professional development. Further, they were able to integrate multiple computational tools that engage their students in CT-STEM practices. These findings suggest that teachers can learn to use computational practices and tools through workshops, and that teachers collaborating with researchers in co-design to develop computational enhanced STEM curriculum may be a powerful way to engage students and teachers with CT in K-12 classrooms.

KEYWORDS
computational thinking, STEM education, K-12, teacher professional development, curriculum design

1. INTRODUCTION
Initiative to incorporate *computational thinking* (CT) in K-12 education face challenges on several fronts, particularly in the United States. CT education often takes place within computer science courses, which may limit access to those who traditionally take computing courses (Heinz, Mannila, & Färnqvist, 2016). Moreover, there is a dearth of K-12 teachers trained in computer science and technologies (Advocacy Coalition, 2018; Cuny, 2012).

In order to address the systemic barriers to CT education, researchers argue for the integration of CT in K-12 STEM classes (Wilensky, Brady, & Horn, 2014). Integrating CT in STEM classes can broaden access to computational practices for all students, as STEM classes are required in middle and high school. Further, students' use of computational tools has been shown to deepen learning in mathematics and science domains (e.g., Brady et al., 2016; Wilensky, 2003). Weintrop and colleagues (2016) organize computational thinking practices in mathematics and science classrooms into four strands: data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices. In this paper, we focus on *modeling and simulation* (using, modifying, and creating computational models) and *data practices* (collecting, visualizing, and analyzing data). Engaging in these CT-STEM practices can help students develop science and mathematics content understanding through authentic STEM practices used in modern science (Weintrop et al., 2016).

Integrating CT in STEM classes further addresses the shortage of teachers trained in computer science by shifting the focus to training STEM teachers in the computational tools and practices relevant to their associated fields. This shift requires both curriculum designers and teachers to reimagine classroom practices and to learn how to incorporate computational methods and tools (Ball & Forzani, 2009; Windschitl et al., 2012). We address this shift using a Design Based Implementation Research (DBIR) framework (Penuel et al., 2011) that supports teachers in professional development and integration of computationally enriched STEM units. Over multiple years of partnering with teachers and schools, our team has shifted from providing day-long professional development to ongoing teacher-driven support. Through these design iterations, we have sought to support teacher ownership, agency, and comfort in teaching with computational tools.

In the latest design iteration, we position teachers as active co-designers in modifying their existing STEM curricula to include computational tools and practices. Our approach foregrounds teachers' views on how the curriculum aligns with teaching strategies and expectations for student learning (Allen & Penuel, 2015; Coburn, 2005; Penuel et al., 2009). Researchers serve as computational experts and work alongside teachers to develop new computationally enriched STEM curricula that align with individual teacher's views and goals. The co-design process aims to (1) help teachers develop an understanding of CT and (2) empower teachers to integrate and teach CT in their STEM courses. In this paper, we present the results of a month-long professional development in which high school teachers co-design CT-STEM curricula with researchers. We investigate the research questions: (1) What did teachers learn about CT through a 4-week professional development? and (2) How did teachers integrate CT into their curriculum?

2. METHOD
To investigate our research questions, we developed the CT-STEM Summer Institute (CTSI), a 4-week professional development workshop that positioned teachers and researchers as co-designers of curriculum. Teachers and

researchers formed design teams by subject area: three biology teachers (pseudonyms: Betty, Briana, Brooke); one chemistry teacher (Carrie); three physics teachers (Penny, Peter, Philip); and one mathematics teacher (Matt). The eight participants teach high school science or mathematics in four U.S. public schools (2 urban and 2 suburban). Teachers received $1000 U.S. dollars per week of participation in CTSI and were asked to create a CT-STEM curriculum for their classroom that would be implemented in the following school year. Seven graduate students and one post-doctoral researcher were assigned to work with teachers based on their prior experience working with specific subject areas and participating teachers.

Table 1. Overview of Professional Development Activities over Four Weeks of CTSI, Organized by Day.

Week	Monday	Tuesday	Wednesday	Thursday	Friday
1	Pre-survey Introductions Demo CT Lesson	Computational Models and CT-STEM Practices	Computational Tools	Computational Tools Unit planning Reflection	Work from home
2 + 3	Work from home Review partner's work	Discuss feedback Co-design (2-3 hours)	Co-design (2-3 hours) CT-STEM Workshop	Co-design (3.5 hours) Reflection	Work from home
4	Work from home Review partner's work	Discuss feedback Co-design (2-3 hours)	Co-design (3 hours) CT-STEM Workshop	Co-design (3.5 hours) Reflection	Post-survey Post-interview Co-design (1 hour) Curriculum Showcase

Table 1 shows an overview of activities during the 4-week professional development. Teachers and researchers met in-person for 14 days from 10am-3pm, with one hour for a catered lunch.

The first week of CTSI (4 days) comprised of workshops led by the researchers. Each workshop introduced computational practices and tools by engaging teachers in lessons designed for students. Each lesson demonstrated how computational tools can engage students in CT-STEM practices while learning disciplinary content. For example, one lesson (https://tinyurl.com/IntroToCT) first asked teachers to use, modify, and debug a series of computational models that simulate how fire spreads through a forest (http://tinyurl.com/netlogofire; Wilensky, 1997) using *NetLogo*, a multi-agent programmable modeling environment (Wilensky, 1999). Next, teachers collected and analyzed 'density vs. percent burned' data using *CODAP* (https://codap.concord.org/; Common Online Data Analysis Platform), a web-based data analysis environment. Then, they posed research questions about other variables that may affect the spread of fire and discussed how scientists use such computational models. Finally, teachers reflected on the pedagogy of CT-STEM practices and how they may use computational models and/or data analysis tools with students.

In addition to NetLogo and CODAP, teachers engaged in *Unplugged* CT activities, which teach CT without computing tools (e.g., writing loops on paper), and *NetTango*, a blocks-based programming interface for exploring NetLogo Web models (Horn et al., 2014), in the context of a chemistry unit on molecular particle collisions.

The last three weeks of CTSI provided co-design time for teams of teachers and researchers to sit together as they worked on computational models and units. Teams engaged in approximately 24 hours of in-person co-design time. On Fridays and Mondays, teams worked from home and communicated via email as needed. Each team reviewed each other's work on Monday afternoons and discussed the feedback on Tuesdays. In addition, teams engaged in supplemental CT-STEM workshops that focused on CT tools or pedagogy on Wednesdays and participated in a reflection session on Thursdays. Each co-design team differed in how they collaboratively built models and curricula materials (Kelter et al., 2020).

At the end of CTSI, the teachers and researchers showcased their co-designed CT-STEM curriculum in an event open to the community: https://tinyurl.com/CTSI2019Expo. All teachers also responded to pre/post surveys and post-interviews, as described below.

2.1. Data Sources

To assess what teachers learned from CTSI (**RQ1**), the 33-item *pre/post surveys* asked teachers to rate on a 5-point Likert Scale (*1* = Strongly Disagree, *5* = Strongly Agree): their perception of CT (Adapted from Cabrera et al., 2018) and comfort with CT-STEM practices. Further, in the *post-interview*, we asked teachers what they learned from CTSI.

To assess how teachers integrated CT into their curriculum (**RQ2**), we asked teachers to describe their curriculum in the *post-interview* and examined the computational tools and practices used in their *CT-STEM curriculum*.

3. RESULTS

3.1. What Teachers Learned about CT

To address **RQ1** (what teachers learned about CT through professional development), we first analyzed teachers' ratings on the *pre-/post-survey*. Due to the small sample size, we qualitatively compare differences from pre to post. Note that Brooke did not complete the pre-survey (4.8 average across all categories on post-survey) and Philip did not complete the post-survey (4.4 average on pre-survey).

Table 2. Average Pre/Post Survey Response by Category.

	CT Value	CT in STEM	CT Integration	Modeling Practices	Data Practices	**Overall**
Pre	4.1	4.1	4.1	3.8	3.0	**3.7**
Post	4.3	4.6	4.4	4.2	4.0	**4.2**

As shown in Table 2, teachers were more likely to agree or strongly agree on all item categories on the post-survey, compared to the pre-survey. That is, after the professional development, teachers reported that they understood the role of CT in STEM education and valued CT to a greater degree. Teachers also reported higher confidence in their ability to identify and integrate computational modeling and data practices into their teaching.

Next, we analyzed the *post-interview* responses to: "What have you learned from CTSI?" We qualitatively reviewed responses of all eight teachers to identify themes mentioned by multiple teachers. Below, we present teachers' responses with the four themes underlined: computational thinking, computational tools, coding, and collaboration.

3.1.1. Computational thinking

Two teachers described learning about CT: Briana (see Section 3.1.3) and Peter. Peter described different levels of CT practices in how they affect students' thinking:

I think being able to see the different domains of computational thinking and the different levels was important. That at one level, it's just: Can you use a model? Can you change a model? Right? Can you collect data? Can you represent data? That's one level, but then can you dig in deeper? Can you change a model? Can you design a model? Can you manipulate data and represent it in different ways? Those are deeper levels that the goal is to try to push down as far as you can to get the kids' thinking, at a really deep level. So that's one thing that I've learned about computational thinking itself.

Peter learned that CT can engage students in more procedural thinking, such as using models and collecting data, as well as more deep conceptual thinking, such as changing and designing their own models. His goal now is to focus on "push[ing]" students' thinking "at a really deep level" because "the different levels [are] important."

3.1.2. Computational tools

Four teachers stated what they learned about *specific computational tools* (Peter, Matt, Philip, and Carrie). Peter and Matt listed different computational tools that they learned about and plan to use in their classroom. Additionally, Matt discussed how the computational tools can help students engage in math as professionals do:

I'd never heard of CODAP or NetLogo or NetTango or any of those. So for me, it just gave me some tools that I can use in stats and hopefully geometry to present math in a relevant way to today's learners. I think it will help me answer the question: Why are we learning this? When am I ever going to have to use this? 'Cause it'll be easy to show them, this is what actual researchers are using. 'Here's what actual statisticians are using, rather than we're using the calculator because that's what the AP exam requires you to use.'

Philip and Carrie, who had prior experience building models or implementing CT-STEM lessons, both stated that they became aware of new tools. Carrie added that she was "very excited that [she's] integrating some CODAP this year...[She] already see[s] other possible places in [her] year that [she] can use [CODAP]." Even though the workshops only aimed to help teachers integrate tools into their CT-STEM curriculum, teachers identified CT tools as resources they can use for other lessons in their classroom.

3.1.3. Coding/programming

In contrast to the four teachers above who seemed "excited" and comfortable integrating computational tools into their classrooms, three of the female teachers mentioned learning about *coding* in general because they had little or no prior experience (Betty, Penny, Briana). For example, Betty said she cannot "code anything" but learned how code works and how to explain it to her students:

I knew nothing about coding […] I cannot code anything, maybe a tiny little change I can make, but I at least see now what goes into it and I think I'll be better at explaining things to the kids.

Although Betty feels she can only make "a tiny little change" in code, another teacher Penny discussed learning "a lot" about coding by building NetLogo models for her curriculum and participating in the introductory workshops:

I never knew anything about NetLogo before and I've now learned a lot about NetLogo and modified or helped build some simulations. And that's largely my first and only exposure to coding. So that's relatively new...I thought a couple of the coolest things that we did were within the first week workshops you have for us: the forest fires simulation....that was the first thing where we really looked at the code behind it- and why aren't the trees burning? And I thought that was fun. As well as just seeing the emergent phenomena in that throwing in the same density doesn't always result in the same forest burn rates. So that was cool for me.

While Penny learned that coding was "fun" and "cool" in the first week, Briana stated that she learned to love coding in the second week as she started writing her curriculum and now wants to learn more about how to build models herself. She also mentions learning about all four themes stated across teachers (computational thinking, computational platforms/tools, coding, and collaboration):

I learned more about what computational models are, what computational thinking is. I learned how to incorporate that into my classroom and my lessons more easily. Collaboration is so important. I learned a little bit of how to do some coding and learned different modalities that can be used for different platforms that can be used for different types of analysis....the second week, my Aha moment was I think that creating models is way cooler than writing curriculum...I thought I hated the coding process. At first, I was like it's gonna be terrible, but when I actually learn the foundation/fundamentals, I was like: well this is actually really cool: how a line I write can completely change how something else works. So that was an Aha moment for me is that I would love to learn more about how to do that.

3.1.4. Collaboration

Lastly, four teachers mentioned the value of *collaboration* in their curriculum design process (Briana, Betty, Brooke, Carrie). Betty learned that "a whole team of people" contribute to constructing computational models:

I learned that the value of co-design is very important. Yeah, I'm just more comfortable with using NetLogo...I think just understanding that things have to be coded, like preferences have to be put in there. Someone put that in 'cause I'm like: how do these models know to do this? So you have to actually do some of the research ahead of time, then put it in. And you need a whole team of people. It's not- a computer programmer doesn't know the science necessarily, so you need a scientist with a computer programmer to work together. I love that. I love that idea.

Betty learned that "co-design is very important" because models involve collaborative design decisions from experts from different fields. Similarly, Brooke noted that she benefited from collaborating and brainstorming with the researcher in her team who had a different expertise:

It's just been really nice to have the time to sit down and have conversations around some of this stuff. That's giving me time to dig into the content, research more about what actually- I want it to be about think a little bit more deeply about like the alignment

of the unit itself. And that's just been really great to have [researcher] there to say: Okay, this is the idea. What might fit well? And he'll be like: 'Oh, you could do this or you could do that.' Or just that piece of brainstorming around expertise that I don't have.

In addition to brainstorming, Carrie also mentioned that "[h]aving a researcher with us the whole time was so beneficial" because she could get help on her questions right away from a collaborator sitting right next to her.

3.1.5. Summary of what teachers learned from CTSI

In sum, teachers generally learned more about CT after CTSI. Some of them learned about computational tools and practices that they can integrate into their classroom. Other teachers with limited CT experience learned coding so that they can engage in and explain CT to their students. Further, multiple teachers mentioned collaboration, which supported them in building and integrating computational tools and practices into a CT-STEM curriculum.

3.2. How Teachers Integrated CT

To assess how teachers integrated CT into their curriculum (**RQ2**), we analyzed how teachers used computational practices and tools in their CT-STEM curricula. We first describe their curriculum below and then discuss their use of CT-STEM practices (summarized in Table 3):

1. *Experimental Design and Computational Thinking*: 8-day AP Biology unit that uses a physical lab, CODAP, NetTango, and NetLogo to conduct experiments on animal behavior, further described below (Betty)
2. *Evolution Part II: Natural Selection (Darwin's Finches and The Case of the Rock Pocket Mouse)*: 20-day Freshmen Biology unit that uses CODAP and NetLogo models to collect and analyze data on the mechanisms of natural selection (Briana)
3. *Climate Change in the Great Lakes*: 10-day Environmental Science unit that uses Unplugged activities, CODAP, and NetLogo models to investigate various environmental factors and make sense of climate change models (Brooke)
4. *Energy in Chemical Reactions*: 13-day Chemistry unit that uses NetLogo and CODAP to explore changes in energy when bonds break and form during chemical reactions (Carrie)
5. *Charge Interactions*: 8-day Physics unit that uses a physical lab, CODAP, NetLogo, and PhET simulations to explore the behavior of charges in electricity and magnetism, further described below (Penny and Peter)
6. *1-D Kinematics Motion Maps*: 3-day Physics unit that uses NetLogo and NetTango to analyze and draw velocity in kinematics motion maps, building on Philip's 1-D Kinematics NetLogo model, further described below (Penny and Peter)
7. *1-D Kinematics and Newton's Laws*: six Physics lessons that use CODAP, NetTango, and NetLogo to collect and analyze data through writing formulas and generating graphs on kinematics and Newton's Laws, implemented throughout the fall semester (Philip)
8. *Descriptive Statistics*: 8-day AP Statistics unit using Python notebooks and Unplugged activities to generate formulas, data tables, and plots that describe various real-world datasets (Matt)

Table 3. CT-STEM Practices Targeted in Curriculum

	Curricular Unit (see Section 3.2)							
	1	2	3	4	5	6	7	8
Modeling and simulation practices								
Using computational models (CMs) to understand a concept	x	x	x	x	x	x	x	x
Using CMs to find and test solutions	x	x	x			x	x	
Designing CMs	x				x			
Assessing CMs	x	x	x	x	x		x	
Constructing CMs	x	x			x			
Data practices								
Collecting data	x	x	x	x	x	x	x	
Manipulating data	x	x	x	x	x		x	x
Analyzing data	x	x	x	x	x		x	x
Visualizing data	x	x		x	x	x	x	x
Creating data	x	x	x	x	x	x		

The descriptions of CT-STEM curriculum show that all teachers integrated several computational tools into their curricula to teach disciplinary content. In addition, Table 3 shows that all CT-STEM curricula targeted multiple CT-STEM modeling and data practices. To better understand how teachers integrated computational practices and tools, we present three example curricula (#1, #5, #6) below.

Biology. Betty, with her co-design partner, developed *Experimental Design and Computational Thinking* (#1) for her AP Biology course. She described it as: "really about scientific design and inquiry." In the unit, students design experiments to find the preferred habitat conditions of the *pill bug (rolypoly)*. Betty decided that students start with a physical lab experiment using two connected chambers, one damp and one dry. The students place 10 pill bugs and observe change in population of the two chambers over time. After the physical experiment, students then explore, modify and recreate the animal behavior experiment digitally using NetLogo and NetTango models.

Betty also explained that her unit engages students in multiple CT-STEM data practices: "the kids learn how to set up a controlled experiment, how to collect data, how to make graphs, and it's also where we start to teach them how to analyze some of that data." She integrated these data practices with the CT-STEM practice of using models:

[My class uses] the computational model to learn about the importance of sample size because we only get to use 10 rolypolies and then when we do Chi Square, we don't always get good answers. And then we looked it up, they're like: oh, you need at least 30, for your sample size...So with the model, they can say: oh, what happens if we have 20 rolypolies, 40 rolypolies?

Betty wanted students to not only use models but modify them based on a physical lab: "[students] are now also learning how to change the model. So the first model just has wet and dry, and then in the second activity, they actually changed the code and add their variable, like the one that they tested in class." Specifically, Betty wanted students to learn "that the model is actually coded by a human, based on things that actually happened in real life," as she herself learned at CTSI (see Section 3.1.4). Her integration of NetTango block-based programming makes this design decision particularly salient: "[students] build their chamber using NetTango. Then they put the rolypolies in and all the

rolypolies escape because they didn't tell them to stay within the chamber."

Although Betty expressed that she "cannot code anything" (see Section 3.1.3), her CT-STEM curriculum is the only unit that integrates all modeling and data practices into science content (see Table 3) and forefronts CT in its title.

Importantly, after Betty taught this unit in the fall, classroom observations and an interview suggests that this unit helped students learn science content and engage in CT-STEM practices because Betty discussed coding and CT in context of disciplinary science content, as a result of the professional development (Peel et al., 2020).

Physics. Peter and Penny, who work at the same school, developed two units together for their general Physics classes. With their co-design partners, they designed *Charge Interactions* (#5), which focused on "electrostatics: electric charge, Coulomb's law, electric fields" (Peter), and a short unit on *1-D Kinematics Motion Maps* (#6).

The electrostatics unit first asks students to engage in physical lab experiment with sticky tape and then explore a NetLogo library model on electrostatics (Sengupta & Wilensky, 2005), which was modified with researchers to fit the curriculum. Penny described the unit as primarily focused on the model and how the code works:

Most of it is around the simulation and specific questions asking them to observe particular behaviors or how things happen using their prior knowledge to try to explain why those are things that are happening. And then a few questions asking them to look at the code and, fine, where did we program in that the electron should repel from each other? Like where did we program in that the conductor's color is gray. Could you change that?

Then, students use CODAP to understand Coulomb's Law, as Peter explained: "If we really want them to come up with Coulomb's law, which is our goal, then you have to keep one thing constant and vary another. And CODAP lets you do that really quickly. So that's why we chose that." Finally, students examine a PhET simulation of charges.

Penny and Peter finished their first unit in Week 3, and then modified Philip's 1-D Kinematics NetLogo model for the motion maps unit (#6). Peter saw this short unit as a way to help students dynamically see changes in velocity: "[students] don't often see the map being drawn, as something moves. I think that the simulation that we put together does that and sort of bridge that gap between what we want them to see and what they actually see." The unit also asks students to build their own motion map using NetTango, as Peter explained: "The NetTango thing is a way to help kids gain more control over making a motion map…they have that ownership of the whole process and I think they'll be able to internalize what's going on better."

As of this writing, Penny and Peter have not yet implemented their Charge Interactions unit, but classroom observations of students engaging with the 1-D Kinematics Motion Maps unit showed that both teachers encouraged students to not only understand the science content, but to "explore the code" and "try to break the model."

4. DISCUSSION

Results from our qualitative study suggests that engaging high school STEM teachers in workshops and co-design of CT-STEM curricula in a 4-week professional development can help them develop an understanding of CT and integrate CT into their classroom. We are particularly encouraged by the fact that although these eight teachers already valued CT at the beginning of the workshop because they chose to participate in the professional development, all teachers reported even more favorable perceptions of CT and greater confidence in integrating it into their classroom at the end of the professional development. Teachers shared in post-interviews that they learned not only about CT and computational tools for their classroom, but also about coding in general and the value of collaboration in the co-design process. Due to the relatively recent emergence of CT in STEM for K-12 teachers, particularly in the United States, this work takes one step towards understanding where teachers may need particular support when learning about CT and how to help teachers integrate CT into their classroom practices.

Our analysis of co-designed curriculum showed all teachers were able to integrate multiple computational tools that engage their students in CT-STEM practices. Teacher interviews and classroom observations show that teachers designed and implemented activities that reflect what they personally learned about coding, computational tools, and CT during the professional development. For example, Betty learned that computational models involve design decisions made by people and thus engaged her students in designing computational models where they write code for the behaviors that they expect to see. Further, because Penny found it "fun" and "cool" to see the code behind a model to understand how it works, she encouraged her students to similarly explore and break the code.

Taken together, these findings suggest that teachers benefited from both parts of our professional development: workshops in Week 1 and co-design in Weeks 2-4. Particularly, learning about specific computational tools and how to use them in the context of disciplinary content was important for four of the eight teachers, who reported being "excited" about integrating the tools into their classrooms. However, three of our teachers had little experience with coding and may not have the ability to integrate new computational tools into their classroom without the additional support provided in Weeks 2-4. At the end of the professional development, these three teachers reported learning to be comfortable with code and one teacher, Briana, even learned to love coding in the second week when she began working side-by-side with researchers to co-design curriculum. Moreover, multiple teachers viewed researchers as valuable thinking partners with expertise in CT. Hence, co-design may be an effective way to help teachers in integrate CT into their curriculum, particularly those with little or no CT experience. This finding aligns with prior work which showed that teachers' confidence in CT and ability to reach their curricular goals grew over a multi-week process of working with researchers as co-designers (Wu et al., 2020). We propose that additional research support integration of CT in K-12 by positioning teachers not only as learners of CT in workshops or

trainings, but as co-designers and collaborators who can augment existing STEM disciplinary content with CT in their classroom.

This work has the potential to engage more K-12 teachers and students in computational practices and tools by integrating CT into existing K-12 STEM classrooms. Through one summer professional development, teachers were empowered to develop and implement eight computationally enhanced STEM curricula for up to three weeks in mathematics and science classrooms. Our observations of these classrooms showed that the teachers talked about their experience during the 4-week professional development and leveraged what they learned about CT to help students become more comfortable with CT and engage in CT-STEM practices. Additional professional developments will help us identify what factors contribute to our success, beyond those specific to our eight teachers. This will help us scale this work to a larger population using in-person and online support on CT integration. By helping more teachers understand CT and computational tools, we can empower K-12 STEM teachers to engage their students in authentic scientific practice while also broadening participation in computing.

5. REFERENCES

Advocacy Coalition. (2018). *2018 State of Computer Science Education*. Retrieved October 8, 2019, from https://advocacy.code.org/

Allen, C. D., & Penuel, W. R. (2015). Studying Teachers' Sensemaking to Investigate Teachers' Responses to Professional Development Focused on New Standards. *Journal of Teacher Education, 66*(2), 136-149.

Ball, D., & Forzani, F. (2009). The Work of Teaching and the Challenge for Teacher Education. *Journal of Teacher Education, 60*(5), 497-511.

Brady, C., Orton, K., Weintrop, D., Anton, G., Rodriguez, S., & Wilensky, U. (2016). All Roads Lead to Computing: Making, Participatory Simulations, and Social Computing as Pathways to Computer Science. *IEEE Transactions on Education, 60*(1), 59-66.

Cabrera, K., Morreale, P., & Li, J. J. (2018). Computer Science+ Education: An Assessment of CS Professional Development. *Journal of Computing Sciences in Colleges, 33*(3), 141-147.

Coburn, C. E. (2005). Shaping Teacher Sensemaking: School Leaders and the Enactment of Reading Policy. *Educational Policy*, *19*(3), 476-509.

Cuny, J. (2012). Transforming High School Computing: A Call to Action. *ACM Inroads, 3*(2), 32-36.

Heintz, F., Mannila, L., & Färnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. *Proceedings of 2016 IEEE Frontiers in Education Conference (FIE)*, 1-9.

Horn, M. S., Brady, C., Hjorth, A., Wagh, A., & Wilensky, U. (2014, June). Frog Pond: A Codefirst Learning Environment on Evolution and Natural Selection. *Proceedings of the 2014 conference on Interaction Design and Children*. ACM, 357-360.

Kelter, J., Peel, A. M., Bain, C., Anton, G., Dabholkar, S., Aslan, U., Horn, M. & Wilensky, U. (in press). Seeds of (r)Evolution: Constructionist Co-Design with High School Science Teachers. *Proceedings of Constructionism 2020.*

Peel, A. M., Dabholkar, S., Anton, G., Wu, S. P. W., Horn, M. S., & Wilensky, U. (in press). A case study of teacher professional growth through co-design and implementation of computationally enriched biology units. *Proceedings of 14th International Conference of the Learning Sciences (ICLS) 2020*. Nashville, TN.

Penuel, W. R., Fishman, B. J., Cheng, B. H., & Sabelli, N. (2011). Organizing Research and Development at the Intersection of Learning, Implementation, and Design. *Educational Researcher, 40*(7), 331-337.

Sengupta, P. and Wilensky, U. (2005). *NetLogo Electrostatics model.* Retrieved December 1, 2019, from http://ccl.northwestern.edu/netlogo/models/Electrostatics

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. (1997). *NetLogo Fire model*. Retrieved December 1, 2019, from http://ccl.northwestern.edu/netlogo/models/Fire

Wilensky, U. (1999). *NetLogo*. Retrieved December 1, 2019, from http://ccl.northwestern.edu/netlogo/

Wilensky, U. (2003). Statistical Mechanics for Secondary School: The GasLab Multi-agent Modeling Toolkit. *International Journal of Computers for Mathematical Learning*, *8*(1), 1-41.

Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering Computational Literacy in Science Classrooms. *Communications of ACM*, *57*(8), 24-28.

Windschitl, M., Thompson, J., Braaten, M., & Stroupe, D. (2012). Proposing a Core Set of Instructional Practices and Tools for Teachers of Science. *Science Education*, *96*(5), 878-903.

Wu, S.P.W., Anton, G, Bain, C, Peel, A.M., Horn, M.S. & Wilensky, U. (2020). Engage teachers as active co-designers to integrate computational thinking in STEM classes. Presented at NARST Annual International Conference (NARST 2020). Portland, Oregon.