Back to Computational Transparency: Co-designing with Teachers to Integrate Computational Thinking in Science Classrooms

Connor Bain, Gabriella Anton, Michael Horn, and Uri Wilensky
connorbain@u.northwestern.edu, gabby.anton@gmail.com, michael-horn@northwestern.edu, uri@northwestern.edu
Northwestern University

Abstract: Integrating computational thinking (CT) in the science classroom presents the opportunity to simultaneously broaden participation in computing, enhance science content learning, and engage students in authentic scientific practice. However, there is a lot more to learn on how teachers might integrate CT activities within their existing curricula. In this work, we describe a process of co-design with researchers and teachers to develop CT-infused science curricula. Specifically, we present a case study of one veteran physics teacher whose conception of CT during a professional development institute changed over time. We use this case study to explore how CT is perceived in physics instruction, a field that has a long history of computational learning opportunities. We also discuss how a co-design process led to the development of a lens through which to identify fruitful opportunities to integrate CT activities in physics curricula which we term computational transparency–purposefully revealing the inner workings of computational tools that students already use in the classroom.

Introduction

There are increased efforts to incorporate the NGSS practice of computational thinking (CT) in K-12 education (Grover & Pea, 2013). However, initiatives to integrate CT in K-12 education face challenges on several fronts. Much of CT education takes place within computer science courses, which limits access to those who traditionally take computing courses (Margolis & Fisher, 2003; Grover & Pea, 2013). Moreover, there is a dearth of K-12 teachers trained in computer science and technologies (State of Computer Science Education, 2018). In order to address the systemic barriers to CS education, Wilensky, Horn, and colleagues argue for the integration of CT in K-12 STEM classes (Wilensky, Brady & Horn, 2014). They characterized CT practices of STEM professionals to develop a taxonomy of CT-STEM practices for K-12 education (Weintrop et al., 2016). Integrating CT in STEM classes aligns education with modern scientific practices and thus engages students in epistemologically authentic science inquiry as emphasized in NGSS. It also broadens access of computational practices to all students, as STEM classes are required in middle and high school. Moreover, students' exploration of computational tools has been shown to deepen learning in mathematics and science domains (e.g., Blikstein & Wilensky, 2009; Sengupta et al., 2013; Wilensky & Reisman, 2006). 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 (Wilensky, Brady & Horn, 2014). This shift requires designers and teachers to reshape classroom pedagogy and how teachers learn these new computational practices and tools (Ball & Forzani, 2009; Lampert et al. 2013, Windschitl et al., 2012).

In this paper, we explore a possible role of computational thinking within the high school physics classroom. Building upon the work of constructivist and constructionist scholars, we describe how computational models can be used to "unbox" the mechanisms in traditional mathematical models while at the same time obscuring many useful computational ideas at play. Finally, we present a case study of how an experienced physics teacher's conceptions of computational thinking in his classroom changed over the course of a co-design-centered professional development program. Specifically, we describe a number of instances where the teacher saw computational transparency, in this particular case revealing the innerworkings of computational models, as a valuable resource for CT learning.

Using computation as a way of investigating physics

There has been a rich history of studying the intricacies of physics education and ways of understanding the physical world within the learning sciences. The intuitive understandings people develop through observation and interaction with the world around them provide rich windows into exploring how knowledge is constructed and leveraged in formal education. These intuitions, or phenomenological primitives (p-prims), are the building blocks used to explain phenomena around us, ranging from why there are seasons to how gravity influences the toss of a ball (see diSessa, 2007). These efforts to understand cognitive processes behind reasoning about and learning in physics are often paired with the design of computational tools to support deeper learning. diSessa's original work

identifying p-prims and theorizing about a 'knowledge in pieces' approach to cognition translated into the design of a reconstructable computational medium, Boxer, that leverages a spatial metaphor for relating objects or processes, and concretizes functions by making these visible and manipulable (diSessa & Hall, 1986; diSessa, 1991). This was another step toward designing to support the sort of computational literacy for all that Seymour Papert (1980) had argued for by supporting learners' exploration of math and science concepts through their own personal computational constructions.

More contemporary computational tools have targeted physics learning more directly. Researchers have examined the symbolic forms and representational systems within programming languages that lend themselves to scaffolded and productive cognition within physics (Sherin, 2001; Guzdial, 2006; Sherin, diSessa, & Hammer, 1993). Other researchers focused on modeling phenomena, like in the GasLab and CTSiM programs, through which learners can visualize, conceptualize, and explore the mechanisms underlying physics concepts more easily than with the traditional mathematical models (Wilensky, 1999a; Basu et al., 2016). These computational programs and tools function to restructurate physics learning by presenting content in new representational forms (Wilensky & Papert, 2010). In traditional physics courses, concepts are explored through mathematical models, which blackbox the mechanisms acting within the system. In shifting to computational mediums that provide learners with the opportunity to control or manipulate aspects of the system, the mechanisms acting within the system can be exposed and students have the opportunity to more deeply understand the phenomena of study (Wilensky, 1999a). The use of these computational tools to understand STEM content has been shown to engage learners in rich exploration of computation while developing deep content knowledge in physics and STEM more broadly (Brady et al., 2017; Blikstein & Wilensky, 2009; Sengupta et al., 2013).

Despite the availability of deep computational tools like the ones described above, many traditional physics classrooms use these models not as transparent computational artifacts but as blackboxed computational simulations of the real world. By transparent, we mean its inner workings can be seen and leveraged for developing an understanding (Resnick et al., 2000). While using these computational models is a step in the direction of engaging students in computational thinking practices, a more powerful step would be for students to be deeply engaging with models as computational artifacts to be analyzed and understood in their own right just as scientists do (Weintrop et al., 2016). In the following sections, we describe a co-design process with one high school physics teacher who wished to integrate computational thinking into his existing simulation-based curricula. We layout our progress toward two larger research questions: How is computational thinking perceived in physics instruction and how can co-design support the negotiation of the role of computational thinking in science classrooms?

The Computational Thinking Summer Institute

This project is part of a larger agenda of Design Based Implementation Research (DBIR) framework (Fishman et al., 2013) that supports teachers in professional development and integration of computationally enriched science units. Through three 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 science curricula to include computational tools and practices. Our approach foregrounds teachers' views on how the curriculum aligns with goals for students, teaching strategies, and expectations for student learning (Means et al., 2001; Allen & Penuel, 2015; Coburn, 2005). Researchers work closely with teachers as computational experts 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-STEM practices and (2) empower teachers to feel confident in teaching CT-STEM.

The summer institute took place in four weeks during the summer, for five hours a day. Teachers were on campus for fourteen days and were expected to work off-site for the remaining six days. The time on campus was spent in workshops and co-design sessions. Teachers worked on their units from home on the days they were offsite, with email access to the design teams. The first week of the summer institute was designed to provide participating teachers with an introduction to computational thinking, pedagogies of practice, and exposure to potential computational tools to leverage in curricular design. Teachers and researchers participated together as learners in various computationally enriched lessons previously taught in high school courses. Teachers and researchers then reflected on these experiences as learners and analyzed these enactments from a pedagogical perspective. Other lessons focused on introducing a suite of computational tools, including NetLogo (Wilensky, 1999b), NetTango (Horn & Wilensky, 2011), CODAP (Finzer, 2016). The teachers participated in two additional skills-based workshops for deeper exploration of a selected computational tool, and attended two workshops led by the principal investigators of the project on the computational nature of modern science and the role of computation in restructuring education (Wu et al., 2020).

After the first week, the majority of the summer institute was dedicated to co-design, with time allocated for feedback and group reflections. Each teacher was paired with at least one member of the research team as a

co-designer. The structure of these working relationships was flexible, and the working relationship of each design team emerged naturally based on the needs of the teachers. Some teachers focused on the design of curriculum and their co-design partners focused on developing computational models or tools (Kelter et al., 2020). Other teachers dove into modifying and designing computational models themselves and leveraged their co-design partners for just-in-time help and for feedback (Authors, to appear). The structured feedback sessions throughout the weeks provided opportunities for teachers to receive feedback from other content, pedagogical, and computational experts at the institute.

Using computation as a way of investigating physics

The institute included nine teacher-participants across physics, chemistry, biology, statistics, earth sciences, and material sciences. These teachers worked with nine members of the project team each with training in the learning sciences in addition to disciplinary knowledge in the subject-team to which they were assigned. In this paper, we focus on a case study around a single physics teacher, Nathan (pseudonym). We selected Nathan as a case study to showcase the negotiation of the roles of computational models in the classroom and how the meaning of computational thinking could evolve through the institute. Specifically, we unpack how Nathan began to reframe his computational simulations as transparent computational models. Leveraging Yin's case study method and approach to triangulating data (2004), we make use of pre-survey and interview data to capture the teacher's confidence and conception of computational practices and compare audio recordings of design sessions and field notes of classes to construct an evolution of his conceptualization of the role of computational thinking in the classroom.

Case

Nathan has been teaching Physics and Engineering for 31 years and has had ongoing association with his school's makerspace. Nathan brought considerable personal experience with computational thinking. Throughout the years, he has developed numerous NetLogo agent-based models to integrate into his lessons, in addition to incorporating Logger Pro, Excel, PHeT Simulations, and Geogebra, among other computational tools. His experience with computational tools and programming was self-taught and he expressed confidence in his abilities with computational thinking and practices. He self-reported moderate and high confidence in his abilities to define, identify, adapt, create, engage, and help teacher-peers and students with computational modeling and computational data practices, as reported in his pre-survey prior to the summer institute. Nathan elected to participate during his final year of teaching for "idealistic" purposes, saying in his post-interview:

> I just feel like there are certain ideas in physics that should be able to be learned by anybody. So I'm just like, there must be something about this concept that we're doing that they just can't get. And I just wonder what it is...But I feel like everybody at some level should understand what we're saying by these things and somehow we're just not providing an opportunity for them. The way they learn to get it, the way we're presenting it. And I just feel like there's more that we can always do a better job and that sort of thing.

Moreover, while he mentioned content as one learning goal for students, he described his broader goals as "empowering" his students to use physics to reach whatever goals they have and to apply these concepts to different fields. As such, Nathan approached the CTSI with personal goals of improving his approach to teaching in order to more effectively share knowledge with his students, and, ultimately, be more successful in empowering them to leverage physics knowledge in their lives.

Co-design process and negotiation of computational transparency

Nathan participated in the first three weeks of the program, experiencing the week of CT-intensive workshops and two weeks of co-design. The co-design sessions were spent in a larger physics design team with two partnered physics teachers from another local high school and two researcher co-designers (referred to as co-designers), as well as smaller breakout co-design sessions where Nathan partnered with one or both researchers to get directed feedback and support. Nathan's approach to building computational thinking activities was directly related to his definition of "computational thinking" coming into the institute: "using models to mimic natural systems as much as possible...to investigate natural systems by altering variables and analyzing outcomes." This approach was evident in a number of Nathan's NetLogo models that he had used in his class the previous year. Since Nathan already had these models built, one of the two co-designers asked Nathan to walk him through the models and

what he used them to teach. One such model (see Fig. 1) was a 1-D Kinematics model where students would modify the initial velocity and acceleration of a car and observe its motion via a Position vs. Time graph.

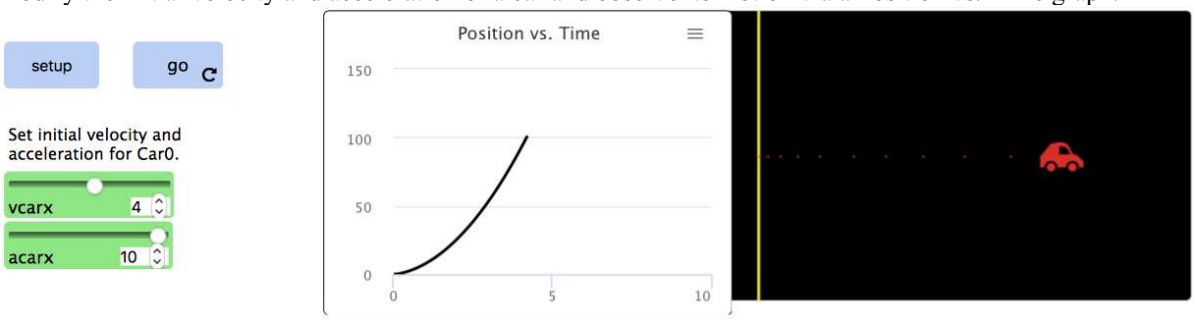

Figure 1. One of Nathan's original models from the previous year. The model aims to help students understand how initial velocity and acceleration of the car affect the shape of the Position vs. Time graph (e.g., with an acceleration of 0, the graph will be linear; a positive acceleration will cause a curved graph like the above).

Nathan designed the model such that manipulations of the two variables (e.g., green sliders) could result in the changes he wanted students to observe in the plot. However, Nathan found mixed results in his implementation of the model, particularly when it came to extrapolating to Velocity-Time, and Acceleration-Time graphs noting that:

> You know, they just go home and there's the graph and then, [it's just a] memorization thing of all that. "The graph did this 'cause a car goes like this...Oh, that's what you get when the cart does that." And so they tried to start memorizing that. And that doesn't help because we get real confused [when it comes to the math].

The co-designer decided to push Nathan to use this as an opportunity to include more CT-practices to accompany the model–to have the students write the code that generates the plots in question. This refocuses the activity toward facilitating an understanding of the shapes of these different plots, but also about un-blackboxing the calculations behind plots themselves. Using CODAP, Nathan set up the model such that it recorded in a table the current time and current position of the car every 0.1 seconds. The students were then asked to write the formulae for Velocity and Acceleration. CODAP stores its data in tables which can have additional columns added that are a "Formula" or the result of some operation on the other columns in the data table. Students are asked to write a Formula for a new column called Velocity with a formula of `(Position - prev(Position)) / (Time - prev(Time)`. Then students are asked to create a similar formula for which they define as "the change in velocity over time" just as velocity "is the change in position over time." Ultimately, the goal of the new model with CODAP integration is the same: students are asked to engage in actual data manipulation rather than just having data automatically analyzed for them. By directly linking the plotted values to the mathematical definition, the aim was to connect the conceptual understandings of velocity and acceleration to the shape of the graph. In this way, Nathan saw a clear pathway to integrate CT to the benefit of learning the underlying physics by asking students to actually *build* the tools they needed to analyze data and not just have a world to *manipulate*. For the better part of the second week, Nathan focused on integrating CODAP and specifically formula writing, into his existing models. This bottom-up data analysis techniques became a common theme in his lessons.

Using programming as a way of scaffolding open-ended exploration

However, Nathan was still hesitant to expose students to the actual underlying code of the models he built. He viewed them as fixed simulations of the real world, not unlike the extensive repository of scientific models available from Colorado's PhET Project (Perkins et al., 2006). Nathan hesitated to have students "look into the blackbox" of how the models themselves were implemented. While he reported having previously tried asking students to look at the code, he expressed a concern that depending on any given student's ability to persist, some of his students would show interest while others would quickly hit various roadblocks. This previous experience was reinforced by Nathan's students expressed desire to have explicit instructions on assignments rather than open-ended experiments. Nathan specifically mentioned "[he] wanted every student to get something out of the experience and not just flail around and leave with nothing." This was reflected in Nathan's designed activities where students were given very little freedom to design experiments or models and instead were asked to solve for or answer questions with a single correct answer.

For example, in one model Nathan had students experiment with the effect of forces in two directions on the motion of an object (see Figure 3). However, rather than having the students designing experiments in order to discover the effect the force had on the motion, Nathan had specific values that students were to test using the model. They repeated the same experimental procedure for five different sets of values. One of the co-designers in the team pushed back on this "over" scaffolding, however Nathan was worried that without any sort of experimental design scaffolding, the students would become overwhelmed with the tasks of both designing and then performing the experiment.

Again, the researchers saw this as another opportunity to integrate a CT practice, particularly designing automated data collection protocols. Nathan had previously expressed interest in using NetTango, a blocks-based dialect of NetLogo that allows a teacher to design domain-specific blocks for students to construct models with, in his models, but was not sure how to start. To convince Nathan that this might be a good place to integrate such a tool, the co-designers and Nathan discussed designing "experiment blocks" that would allow students to design an experiment by varying sets of parameters of the model. Students would be creating scripts that would automatically run the model with certain parameters being varied and others being held constant and then collect the movement data so that they could compare across the different runs (see Figure 3). After designing the blocks, Nathan was on-board, replying "That sounds great...but how do we do it?" Working together with the co-designers, Nathan created these blocks and integrated them into the unit. Nathan saw this as an opportunity to get students to program without having to introduce students to the formal aspects of programming. Instead, students could drag and drop blocks together, simultaneously thinking about experimental design but also about the algorithms behind the model. It is this latter design affordance that Nathan focused on in his later designs.

This implementation subtly foregrounds some previously hidden elements of the simulation. For instance, varying a variable is no longer explicitly a learner driven process and instead controlled by a learner authored script. Additionally, students need to interact with certain simulation variables (e.g., pos1x or the x-position of Lander 1) in order to understand how to create stop conditions for their automated experiments. While this activity allows for a more open-ended, yet scaffolded, exploration of the model, it still does not ask students to construct or inspect any of the underlying code that expresses the physics of the model. The model is still mostly meant as an object through which to study motion.

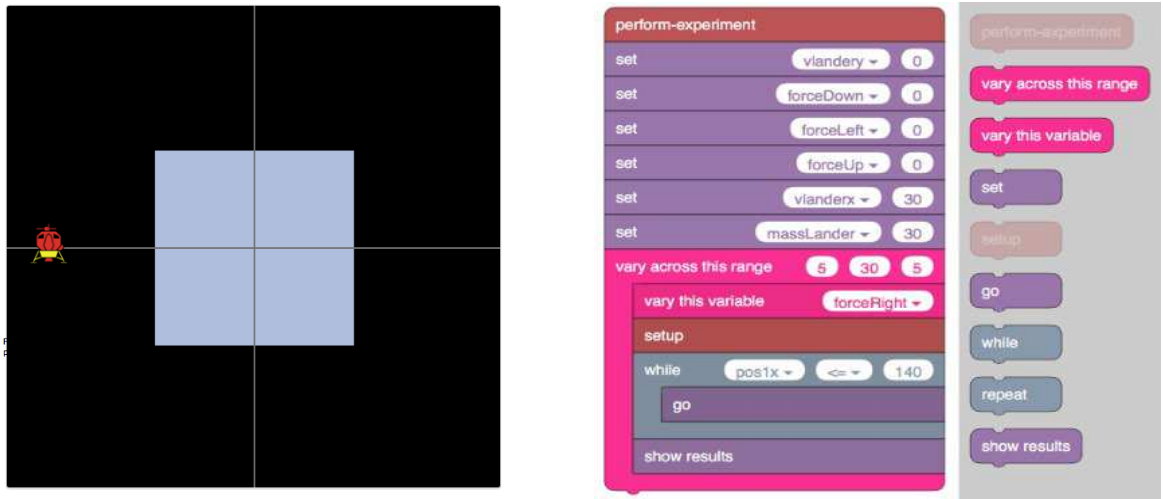

Figure 2. On the left, we see Nathan's XY-Forces model where a spaceship in a vacuum starts with an initial velocity and then experiences certain magnitudes of X and Y-forces, depending on the students' selections, when it enters the blue box. On the right, we see the blocks used to structure experiments in the model. This particular set of blocks will test the effect of varying the `forceRight` variable across a range of 5 Newtons to 30 Newtons and show the different motion trajectories of the lander.

Letting students glimpse into the blackbox of the model

While that marked the end of Nathan's time at the institute, he continued to develop new activities for his unit on Kinematics and Newton's Laws. Several weeks later, Nathan emailed the researchers about a new model directly related to the earlier model featured in Figure 4. He had modified the model to address "side-by-side kinematics problems," in essence, problems where you are comparing the motion of two objects. However, rather than just have students interact via the usual variable manipulations (i.e. sliders), Nathan had foregrounded the structure of the code in various NetTango blocks-spaces (see below).

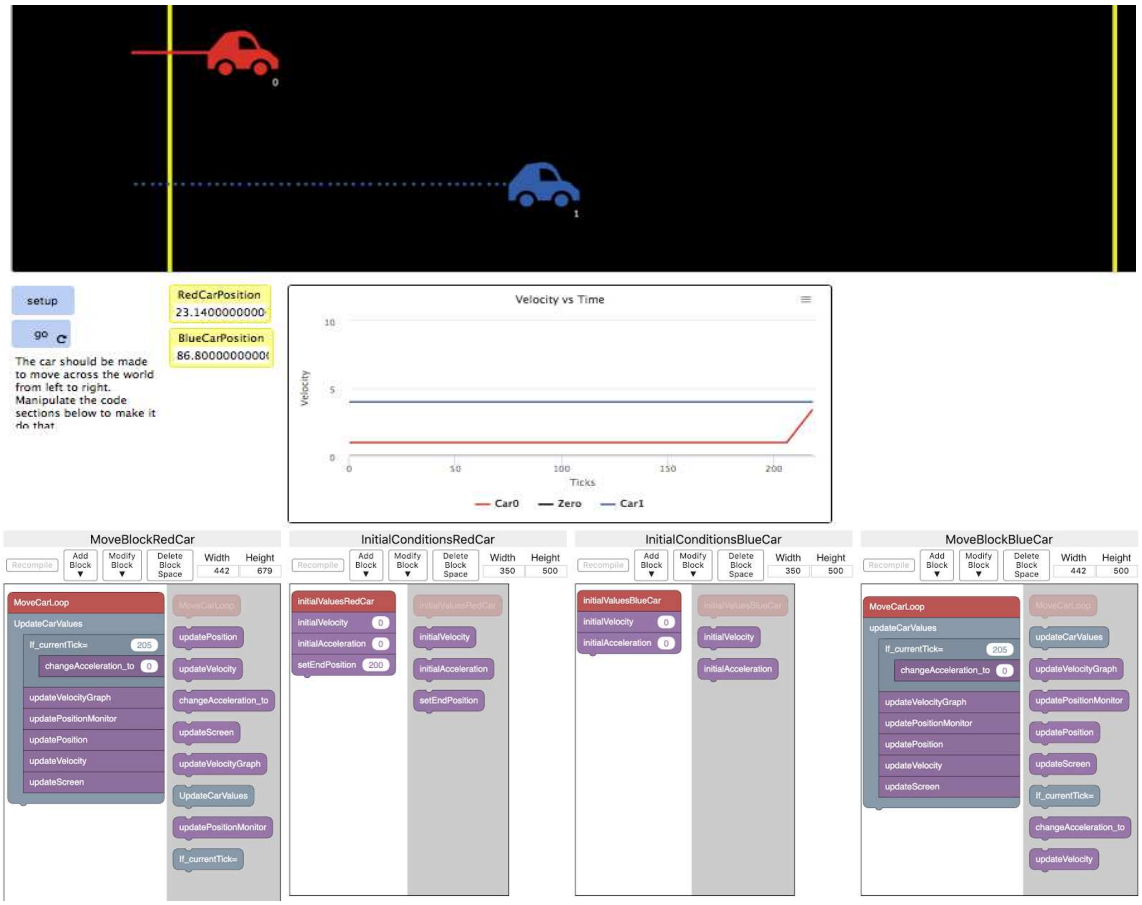

Figure 3. Above are the four different sets of blocks that Nathan designed for his students to control two virtual cars (one red and one blue). Each car has a set of blocks to set the initial conditions and a set of blocks to control its movement. Students used these blocks to recreate given motion graphs that Nathan assigned to each group. This activity asked students to reference the given plots in order to recreate the cars' motion.

This NetTango model is much more complicated than the previous model. While students ultimately were only required to interact with the blocks by changing the parameters of the blocks and not the structure or flow, Nathan saw this as an opportunity to make "transparent" the actual mechanisms by which the model operated. This was a clear move from his earlier hesitation to asking students to look into the text-based code behind the model. Nathan was using the blocks as scaffolding for his students to become aware of the code behind the model. To his surprise, several students during the implementation of the unit did change the structure of the code successfully allowing the cars to move in ways that Nathan had not originally envisioned. Not only did Nathan immediately tell his co-designers about this instance but he also shared the development with the class indicating that Nathan may have seen this an empowering moment for students in his classroom. Perhaps more importantly, Nathan did not simply see this model as an opportunity to improve physics knowledge on side-by-side problems, but also saw it as an opportunity to have students see and interact with the code behind the model. No longer were these just simulations that were prebuilt for students to use, but they were objects that could be played with, manipulated, and fundamentally modified by students. When asked to define computational thinking in an interview after the institute, Nathan described it in much broader terms than before: "It's like we're going to try to investigate this phenomenon [lets] write a program to see if we can model it and then see what kind of outcomes coming out of that that we weren't maybe sure of or something different than we would have thought."

Discussion and conclusion

This case study describes our experiences of co-designing computational thinking activities to be integrated into a traditional high school physics classroom. While physics classrooms already use a number of computational tools and simulations, we argue that these tools are mostly used as a means to an end–not as objects to understand in their own right. Nathan's initial definition of computational thinking, "using models to mimic natural systems as much as possible...to investigate natural systems by altering variables and analyzing outcomes," showed

evidence of this sort of framing. Even with Nathan's extensive technological experience and expertise and despite the fact that he had specifically designed a number of simulations himself for use in his classroom, he initially did not see the need to foreground the more computational aspects of the models as learning experiences in his classroom. However, this is not to say that he did not value computational thinking in his classroom as he had, on prior occasions, tried to engage students in modeling activities without much success.

Our first research question asked how is computational thinking perceived in physics instruction? While Nathan demonstrated mastery and understanding of many different computational tools, he did not expect the same level of understanding from his students. It was sufficient for students to be able to use the computational tool or simulation in order to learn the physics content. Additionally, he did not seem to view computational tools as opportunities to promote exploration or open-ended inquiry due to scaffolding concerns. As such, Nathan's computational activities tended to be very scaffolded. While we did not see Nathan dramatically change his attitude towards overall scaffolding of his activities, we did see shifts in Nathan's willingness to foreground computational aspects of his models, even if they conflicted with his 31 years of teaching experience. In addition to asking students to write functions to calculate measurements like velocity and acceleration, Nathan asked students to create their own graphs via CODAP and saw the blocks-based NetTango as a way of foregrounding computer code without overwhelming students.

Our second research question asked how can co-design support the negotiation of the role of computational thinking in science classrooms? We see this co-design as the main driver of Nathan's change in the way he presented activities to students. During these co-design sessions, it became evident that Nathan was concerned about exposing students to the inner workings of computational tools because he viewed programming as something *some* students picked up quickly while others struggled, reducing the overall physics learning experience. Through co-design, the team was able to integrate computational thinking practices within Nathan's existing framework of scaffolding for his students by using computational transparency: taking these technological tools that students took for granted and asking them to glimpse inside. However, this presented a an issue for the researchers on the co-design team: could we design an open-ended exploratory model to be implemented in Nathan's class? Yes. However, what classroom implications would that sort of design directive have on Nathan's attitude towards CT and how he framed the activities in the classroom.

Time and again we have seen teachers struggle with the task of identifying parts of their existing curricula that might be ripe for integrating computational thinking practices. While physics classrooms already use various technological tools in their curricula, there is untapped CT-potential in asking students to understand the computational assumptions and power lying beneath these tools. As such, we see designing for computational transparency as a useful idea that might help other physics teachers conceptualize how to integrate computational thinking activities in their classrooms. There is a long history of physics education and learning sciences research that asks students to look behind the blackbox of the "equation" to understand physics and we argue that this history suggests students could look behind the blackbox of the "simulation" to understand computational thinking practices and physics simultaneously. By asking teachers to identify already existing computational activities in their classrooms and then thinking about how unboxing those activities may engage students in CT practices, we provide a sort of curricular lens through which to focus teachers' efforts on integrating CT. While this is certainly not an all-encompassing strategy for integrating CT in physics, we see this as an "on-ramp" for physics teachers looking to engage students in authentic computational thinking practices.

References

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. M. (2009). The work of teaching and the challenge for teacher education. *Journal of teacher education*, *60*(5), 497-511.

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.

Blikstein, P., & Wilensky, U. (2009). An atom is known by the company it keeps: A constructionist learning environment for materials science using agent-based modeling. *International Journal of Computers for Mathematical Learning*, *14*(2), 81-119.

Coburn, C. E. (2005). Shaping teacher sensemaking: School leaders and the enactment of reading policy. *Educational policy*, *19*(3), 476-509.

diSessa, A. A. (2007). An Interactional Analysis of Clinical Interviewing, COGNITION AND INSTRUCTION, 25:4, 523-565, DOI: 10.1080/07370000701632413

Finzer, W. (2016). Common online data analysis platform (CODAP). *Emeryville, CA: The Concord Consortium.[Online: concord. org/codap]*.
Fishman, B. J., Penuel, W. R., Allen, A. R., Cheng, B. H., & Sabelli, N. O. R. A. (2013). Design-based implementation research: An emerging model for transforming the relationship of research and practice. *National society for the study of education*, *112*(2), 136-156.
Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38-43.
Guzdial, M. (1994). Software-realized scaffolding to facilitate programming for science learning. *Interactive learning environments*, *4*(1), 001-044.Lampert, M., Franke, M. L., Kazemi, E., Ghousseini, H., Turrou, A. C., Beasley, H., ... & Crowe, K. (2013). Keeping it complex: Using rehearsals to support novice teacher learning of ambitious teaching. *Journal of teacher education*, *64*(3), 226-243.
Horn, M. & Wilensky, U. (2011). NetTango [Computer Software]. Evanston, IL: Center for Connected Learning and Computer Based Modeling, Northwestern University.
Kelter, J., Peel, A., Bain, C., Anton, G., Dabholkar, S., Aslan, U., Horn, M., & Wilensky, U. (2020). Seeds of (r)Evolution: Constructionist Co-Design with High School Science Teachers. *International Constructionism Conference*. Dublin, Ireland.
Margolis, J., & Fisher, A. (2002). *Unlocking the clubhouse: Women in computing*. MIT press.
Means, B., Penuel, W. R., & Padilla, C. (2001). *The connected school: Technology and learning in high school*. Jossey-Bass, Inc., 989 Market St., San Francisco, CA 94103-1741.
Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc..
Perkins, K., Adams, W., Dubson, M., Finkelstein, N., Reid, S., Wieman, C., & LeMaster, R. (2006). PhET: Interactive simulations for teaching and learning physics. *The physics teacher*, *44*(1), 18-23.
Resnick, M., Berg, R., & Eisenberg, M. (2000). Beyond black boxes: Bringing transparency and aesthetics back to scientific investigation. *The Journal of the Learning Sciences*, *9*(1), 7-30.
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.
Sherin, B. L. (2001). A comparison of programming languages and algebraic notation as expressive languages for physics. *International Journal of Computers for Mathematical Learning*, *6*(1), 1-61.
Sherin, B., diSessa, A. A., & Hammer, D. (1993). Dynaturtle revisited: Learning physics through collaborative design of a computer model. *Interactive Learning Environments*, *3*(2), 91-118.
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. (1999a). GasLab—An extensible modeling toolkit for connecting micro-and macro-properties of gases. In *Modeling and simulation in science and mathematics education* (pp. 151-178). Springer, New York, NY.
Wilensky, U. (1999b). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.
Wilensky, U., & Papert, S. (2010). Restructurations: Reformulations of knowledge disciplines through new representational forms. *Constructionism*
Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering computational literacy in science classrooms. *Commun. 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., Anton, G., Bain, C., Peel, A., Horn, M., & Wilensky, U. (2020, March). Engage Teachers as Active Co-Designers to Integrate Computational Thinking in STEM Classes. In proceedings of *National Association of Research on Science Teaching*. March, 2020. Portland, Oregon, U.S.A.
Yin, R. K. (2004). *The case study anthology*. Sage.
2018 State of Computer Science Education. (2018). Retrieved from https://advocacy.code.org/

Acknowledgments

This work was made possible through support from the National Science Foundation (CNS-1138461, CNS-1441041, DRL-1020101, DRL-1640201, DRL-1842374, DGE-1842165) and the Spencer Foundation (#201600069). Any opinions, findings, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding organizations.