This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

Balancing Curricular and Pedagogical Needs in Computational Construction Kits: Lessons from the DeltaTick Project

Abstract. To successfully integrate simulation and computational methods into K-12 STEM education, learning environments should be designed to help educators maintain balance between (a) addressing curricular content and practices and (b) attending to student knowledge and interests. We describe DeltaTick, a graphical simulation construction interface for the NetLogo modeling environment designed to make computational model construction a more *accessible* and *responsive* part of science and mathematics curricular activities through domain-specific, customizable construction libraries. With DeltaTick, learners assemble and re-assemble predefined sets of "behavior blocks" to build simulations that represent a particular domain of study. When needed, blocks can be added, adjusted or replaced to better reflect student knowledge, interests, or questions. We present coding analyses and vignettes from *DeltaTick* enactments in middle and high school classrooms to illustrate ways these features allowed learners to explore core curricular ideas, while at the same time accommodating unexpected student or classroom needs and pursuits. From these findings we posit two design principles, *curricular example space* and *levels of responsivity*, for computational modeling environments intended for classrooms. We argue that this design approach can bring into better alignment the complex relationships between modeling toolkits, student knowledge, curricula, and teacher supports in K-12 computational modeling activities.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

Balancing Curricular and Pedagogical Needs in Computational Construction Kits: Lessons from the DeltaTick Project

There are increasing calls to integrate computational modeling and simulation into K-12 science and mathematics education. Interacting with and constructing simulations allows learners to explore the specific mechanisms that underlie a phenomenon of interest (Sherin, 2001; White & Frederiksen, 2005; Wilensky & Reisman, 2006), to engage in inquiry (de Jong & van Jollingen, 1998; Kali & Linn, 2007; Windschitl, 2000), and to interact with scientific and mathematical ideas that might otherwise be difficult or inaccessible (Kaput, 1994; Roschelle, et al., 2000; Wilensky & Resnick, 1999). It can also expose learners to professional practices in STEM fields, where computational methods are changing what it means to generate intuition and conduct experiments (Borwein & Bailey, 2011; Nersessian, 2009).

These calls come at a time when educators are already asked to balance a number of curricular and pedagogical goals. Policy documents and standards emphasize core content and key scientific practices in science and mathematics (CCSS, 2010; NGSS, 2012; NRC, 2012). At the same time, they need to attend and respond to students' own knowledge, and engage them in making sense of and evaluating one another's ideas (Berland & Reiser, 2009; Duncan, Rogat & Yarden, 2009; Franke & Kazemi, 2001; Levin, Hammer & Coffey 2009; Sherin & van Es, 2005). These goals can be in tension, as educators balance curricular learning goals with attending to students' own intellectual pursuits. Navigating this tension is a fundamental characteristic of science and mathematics classroom practice (Ball & Bass, 2000; Dewey, 1904; Hammer, 1997; Metz, 1997; Simon, 1995). This has led many to emphasize the important and complementary roles modeling toolkits, student knowledge, curricula, and teacher support in computational modeling activity (Louca & Zacharia, 2012; Roschelle, Knudsen, & Hegedus, 2010; VanLehn, 2013).

In this paper, we describe *DeltaTick*[i] (Wilkerson-Jerde & Wilensky, 2010), a visual block-based construction interface for NetLogo (Wilensky, 1999b) that allows students to construct simulations for specific domains of study, and allows educators to adapt blocks according to student needs. Using in-

depth vignettes and video coding analyses, we report ways learners used DeltaTick to explore core curricular topics, and ways we adapted explorations for pedagogical purposes. From these findings, we propose two principles, *curricular example space* and *levels of responsivity*, for the design of computational environments for classroom integration. These principles shed light on new ways to address the persistently complex relationships between modeling toolkits, student knowledge, curricula, and teacher supports in K-12 computationally-based modeling activities.

Background and Motivation

Simulation and modeling is core to contemporary mathematics and science (NRC, 2010; 2012). Computational methods help practitioners generate intuition (Bailey & Borwein, 2011), represent and test scientific theories (Grimm et al., 2005), and reveal hidden connections across ostensibly disparate domains (Goldstone & Wilensky, 2008; Sabelli, 2006). Importantly, it is not just access to and use of simulations that are driving these changes, but also the process of building them (Wilensky, 2003).

Given the central roles that computational modeling and simulation play in STEM professional practice, it is not surprising that educators have also begun to explore its pedagogical potential (Clark et al., 2009; Hoyles & Lagrange, 2010; NGSS, 2012). Much of this work has focused on *using* pre-constructed simulations and models as visualizations (Gilbert, 2005), experimental tools (Buckley et al., 2004; Zacharia, 2007), or sites for inquiry (Edelson, 2001; Ketelhut, Nelson, Clarke & Dede, 2010; Williams & Linn, 2002). In this project, we focus on the role that *constructing* computational simulations can play in middle and high school STEM classrooms: as a way for students themselves to express, test, and refine their ideas about how the world works.

Constructionism and Programming to Learn: Building From Student Ideas

There is a long history of integrating programming into mathematics and science education, starting with Constructionist (Harel & Papert, 1991; Kafai, 2006; Papert, 1980) tools for K-12 learners such as LOGO. These environments, inspired by Constructivist theories of learning (Ackermann, 2001;

Piaget, 1952), are based on the premise that constructing *external* physical or digital artifacts can facilitate knowledge construction. The argument is that by constructing such artifacts using primitive rules that connect to their existing ways of navigating the world (such as physical movement), learners' prior knowledge can be reorganized, debugged, and built upon to generate new ideas.

Studies reveal that engaging in programming can help learners reason about the mechanisms and relationships that underlie key phenomena (Blikstein & Wilensky, 2009; Parnafes & Disessa, 2004; Sherin, 2001; Wilensky, 1995), better connect with their experience of the world (Ackermann, 1996; Sengupta & Wilensky, 2009; Sherin, diSessa, & Hammer, 1993), and develop symbolic and generalizable ways of describing ideas (Noss, Healy & Hoyles, 1997; Papert, 1993; Wilensky & Reisman, 2006). Recent calls to incorporate computational literacy (diSessa, 2001) and computational thinking (NRC, 2010; Wing, 2008) into K-12 education emphasize the utility of ideas from computer science for thinking about problems broadly across domains, as well as deeply for particular topics.

But there have been challenges to integrating programming into K-12 science and math explorations. Many programming environments are domain-general, and use primitive elements that might not easily map to complex scientific or mathematical topics (Louca & Zacharia, 2008; Schwartz, 2007). This allows significant flexibility in what students can construct, but may also require significant support for users to build complex models (Harvey, 2010; Hmelo-Silver, 2006) and take longer than matched activities for the same content (Xiang & Passmore, 2010). A common response to the need to scaffold particular learning goals under this time pressure is to employ more curricular structure, which can leave less space for students to pursue their own interests.

We agree that learning programming as a general literacy is essential (Wilensky, 1995; Sondahl, Wilkerson-Jerde & Wilensky, 2011). However, complementary research is needed to design for the immediate pedagogical and curricular needs of classrooms. Several newer generations of computational construction environments have started to address some of these concerns.

New Paradigms to Support Simulation and Modeling in the Math and Science Classroom

Designers and researchers have explored ways to make computational modeling toolkits better suited for complex curricular explorations. Some environments offer primitives that foreground specific classes of phenomena, such as geometry (Logo) or emergent phenomena (NetLogo). To address the difficulties teachers and students experienced related to the syntax and structure of programming, environments including ToonTalk (Kahn, 1998), Alice (Conway & Pausch, 1997), StarLogo TNG (Klopfer, Yoon, & Um, 2005), Scratch (Resnick et al., 2009), SimSketch (Bollen & van Joolingen, 2013), Tern Tangible (Horn & Jacob, 2007; Bers & Horn, 2009); and NetTango (Horn & Wilensky, 2012) have introduced visual paradigms that make it easier for students to know what instructions they have available to use, illustrate how different computational primitives fit together, and eliminate syntax errors. Other environments such as Stagecast Creator (Smith, Cypher, & Tesler, 2000), the Simulation Creation Toolkit (Basawapatna, Repenning, & Lewis, 2013), or SiMSAM (Wilkerson-Jerde, Gravel & Macrander, 2013) allow learners to visually demonstrate how objects should move, relate, and interact.

To make complex curricular explorations more accessible, other construction kits such as Agentsheets (Repenning & Sumner, 1995), MathSticks (Noss et al., 1997) and Behavior Composer (Kahn, 2007) include blocks to represent high-level structures that only make sense for a particular problem. For example, constructing simulations of disease spread in Behavior Composer might involve primitives such as ‘avoid’ or ‘infect’. Other environments provide existing, ‘broken’, or partly constructed simulations for a given content area. Examples include the NetLogo Models Library (Wilensky, 1999c) microworlds such as ChanceMaker (Pratt, 1998) constructed in the Boxer environment (diSessa & Abelson, 1986; diSessa, 1997), and Kynigos’ half-baked microworlds (Kynigos, 2007). Rather than having students start construction from scratch, these approaches invite them to explore curricular targets through modification, repair and extension.

While such approaches have made complex content more accessible for computational exploration, they have also reduced the flexibility students have to construct, test, and revise their own ideas through simulation. Pre-constructed simulations offer students less opportunity to hypothesize about, create and test their own models. Also, high-level primitives may restrict access to particular aspects of a phenomenon, and may not be aligned with students' own understandings of a domain.

Moving Forward: Contributions of the Current Work

Rather than focusing on one or the other–creating highly flexible languages or making complex curricular content more accessible–we take a different approach. *DeltaTick* leverages the accessibility and focus of block-based paradigms and domain-specific programming libraries to provide learners with construction units that reflect the "simplest elements" (Bamberger, 1996, p. 34) that make up a domain of interest[ii]. However, we recognize that these "simplest elements" may differ across contexts, and that educators themselves are most likely to understand which elements can best serve the needs and interests of a given group of learners. Therefore, DeltaTick also allows designers or educators to add or customize construction blocks themselves, and permits users to access and change the low-level code assembled from those blocks to explore a phenomenon in more depth.

We have been iteratively refining and studying DeltaTick for the past five years in middle and high school classrooms with different curricular foci. In this paper, we analyze ways in which particular design features of DeltaTick allowed us to address curricular and pedagogical needs as they emerged in our work. We derive from these experiences two general-purpose principles, *curricular example space* and *levels of responsivity*, for the design of computational modeling toolkits for math and science classrooms. A major contribution of this work is that it reconceptualizes the role of computational modeling toolkits vis-à-vis student knowledge, curricular supports, and pedagogy, and offers principles and examples to support this reconceptualization. We argue that such a reconceptualization can address

persistent issues in the STEM education literature regarding the appropriateness and nature of computational modeling activities for K-12 classroom settings.

The DeltaTick Construction Kit

DeltaTick is a block-based drag-and-drop interface for the NetLogo agent-based modeling environment. NetLogo is a simulation construction environment based on LOGO (Papert, 1980) that allows users to program collections of computational micro-level *agents* (such as molecules in a gas or animals in an ecosystem). The program can then be executed to observe how agents act and interact to produce a system-level outcome of interest, such as air pressure or predator-prey oscillations (Wilensky, 2003; Wilensky & Reisman, 2006; Wilensky & Resnick, 1999).

The standard NetLogo interface includes three tabs: One that provides access to the text-based "Code" of a model, one provides access to interface elements to "Run" and explore the model, and one provides more "Info" (including hints for educators, technical details, and references to relevant publications). DeltaTick replaces the "Info" tab with a new tab called "Build" (Figure 1)[iii]. Video of DeltaTick in action has been included as Supplementary Materials for this article[iv].

To create a simulation using DeltaTick, users load at least one *Behavior Library*: a specially formatted text file that creates domain-specific blocks that can be assembled to create simulations. Users then add a species (a "breed", in NetLogo parlance) to their simulation with the "Add Species" button. This creates a window that represents a collection of simulated agents. The window includes options to give the species a name, shape, color, and assign traits. Users can define how agents of the species will behave by adding combinations of purple "behavior" blocks and green "conditional" blocks, which will be executed once per "tick" or single simulation execution. Users can also add graphs to their simulation by using orange "quantity blocks. As blocks are assembled in the DeltaTick interface, underlying NetLogo text code is dynamically constructed, and can be accessed by clicking on the "Code" tab.

[INSERT FIGURE 1 HERE]

Figure 1 shows a simulation involving two different species, "tortoises" and "hares", built within *DeltaTick*. The simulation starts with 20 tortoises, and 40 hares. Both species perform some of the same behaviors: they "wander" (move randomly about the visuospatial world), and each has a chance to reproduce. Both also have a chance of dying once they reach a certain age, since purple "die-with-chance-of [n]%" behavior blocks are placed inside of green "if age-more-than [n]?" conditional blocks. However, the parameters of these behaviors are different for the two different species. Tortoises wander more slowly, reproduce less frequently, and live longer than hares. And, if hares find any tortoises in the same location as themselves ("if any-here? [tortoises]"), they wander an additional step[v]. The "graph of [population]" window adds a plot of hare and tortoise populations to the simulation interface.

Figure 2 illustrates how traits are used to represent within-species variation (Wagh & Wilensky, 2012). Users select a trait for a species, and define how that trait will vary in the species at the start of a simulation. Once a trait is added, a corresponding trait block appears that can be used with relevant behaviors. In Figure 2, 100 hares will be created, 50 with a vision range of 1 patch, and 50 with a vision range of 2 patches. When the simulation runs, each hare will approach and eat grass within its vision range. When a hare reproduces, its offspring inherit its vision.

[INSERT FIGURE 2 HERE]

Mode of Inquiry

In this paper, we explore the degree to which specific design features of DeltaTick allowed us to (1) focus student explorations of targeted materials and activities (curricular goals), and (2) modify or extend those activities in-the-moment in response to student needs and interests (pedagogical goals). We approach our analysis as an interpretive, multi-site case study (Yin, 2009) focused on the use of DeltaTick as a modeling toolkit for classroom-based curricular exploration. We present and analyze vignettes drawn from data collected during three middle and high school classroom enactments to provide a rich

description of use. We situate these vignettes by reporting results from complementary coding analyses we conducted to explore broader patterns of DeltaTick use across participants and enactments.

Utilizing a case study methodology within a design-based research context allows us to explore DeltaTick in order to develop and revise theory about how modeling toolkits can be designed for and used in classroom settings (Khan, 2008). Indeed, over the course of our work with DeltaTick, we discovered a need to modify our conjectures about what role it would serve within classroom activity based on our experiences. Our original intention was to design an environment and associated materials that would make computational construction activities productive and accessible in classroom settings. However, we often found ourselves revising the environment in response to immediate pedagogical needs, and began to recognize the potential of these unexpected patterns of use.

Data Sources

Our data are drawn from two design-based (Brown, 1992; Cobb, et al., 2003; Collins, Joseph, & Bielaczyc, 2004) curriculum development and research projects across three sites. The projects used DeltaTick to engage learners in constructing, testing, and debugging simulations of natural selection and population dynamic systems, respectively. We worked in classroom settings to engage students in exploring key curricular structures and relationships through multi-day, guided simulation construction activities. Students did not have prior experience with DeltaTick or the NetLogo modeling environment. Table 1 provides details regarding the schools and classrooms where our studies took place.

[INSERT TABLE 1 HERE]

Our focus was on small group patterns of interaction, to inform subsequent iterations of our software and curricula. Therefore we collected video of student small-group work synchronized with captures of their on-screen activity using Camtasia (TechSmith, 2010). We also collected video of whole-classroom activity (Derry et al, 2010), digital artifacts including behavior libraries we constructed and

revised for each classroom enactment, students' simulation constructions, collaborating teacher reflections, and researcher journal notes and reflections.

Case Selection

We selected vignettes from our enactments that illustrate how DeltaTick was used to support curricular exploration and accommodate students' pedagogical needs. To illustrate curricular exploration, we present excerpts from video of small-group curricular activity from one Population Dynamics (high school) and one Natural Selection (7th grade) study. These vignettes (curricular examples C1 and C2) were drawn from different study sites to illustrate typical *DeltaTick* use across grade levels and domains of inquiry. Together, they illustrate how students sometimes used *DeltaTick* by incrementally adding blocks to "build up" an inquiry into a core phenomenon, or used it to "break down" a complex simulation by identifying and reflect on its basic underlying components.

To illustrate pedagogical accommodation, we describe three instances in which we leveraged features of DeltaTick in unexpected ways to accommodate student needs, interests, and questions (pedagogical examples P1, P2, and P3). We selected these instances to illustrate the breadth of adaptations that can be supported by DeltaTick: from small edits to behavior libraries, to the development of new libraries that support extensions or new investigations, to micro-adaptations that are made directly in existing NetLogo code.

Coding Analysis

To establish the representativeness of the vignettes we present within our broader corpus of data, we also conducted a coding analysis of video collected across studies. Here we present the details of this analysis only briefly; more information about data selection and coding is available in Appendix A. *Investigating Curricular Needs*. First, we were interested in exploring the degree to which learners in our studies were able to use DeltaTick to construct and explore computational models that illustrated curricular target ideas. Since our data are from different studies (see Table 1), we restricted this analysis

to focus on intervals of time across studies that could be meaningfully compared. We analyzed video that captured student group interactions during these intervals, synchronized with their on-screen activity (TechSmith, 2010). This included six group episodes from Population Dynamics Study 1, eleven group episodes from Population Dynamics Study 2, and ten group episodes from the Natural Selection study.

For each episode, we marked to the nearest minute each interval of time during which students were actively engaged in model construction. For each of these intervals, we identified whether students (a) successfully constructed a simulation that exhibited the behavior specified in their curriculum materials, and (b) actively engaged in exploring the causal chains that underlie that behavior. We also marked whether (c) students' model construction behavior could be best described as "building up" or "breaking down". By "building up", we mean students added or adjusted blocks in their simulation incrementally as they worked on constructing a target model or making sense of the phenomena under investigation. By "breaking down", we mean students removed blocks or modified parameters for existing blocks in order to deconstruct the mechanics of a simulation in order to make sense of or create a model of a target phenomenon.

Investigating Pedagogical Needs. Second, we were interested in the degree to which design features of DeltaTick did, or could have, supported educators in making adjustments for students and student groups depending on their pedagogical needs and interests. Since this was initially unexpected and less frequent than students' use of the tool for meeting curricular goals, we did not restrict the scope of our analyses to particular intervals of time in this case. Instead, we analyzed all available data including group-level video, research notes, and behavior library files from classroom enactments.

Across these data sources, we documented each unique instance where educators (a) adapted existing behavior libraries or blocks, (b) created new behavior libraries or blocks, or (c) worked with students to directly edit NetLogo code. We also documented opportunities where we found clear *potential* for such adaptations that we did not pursue, for example when learners themselves suggested new

behavior blocks or modifications to existing blocks to one another while working with the environment, or directly viewed or edited NetLogo code in an effort to modify their model without a facilitator.

Findings Part I: Using DeltaTick to Conduct Curricular Investigations

The majority of participating groups successfully created models that illustrated curricular concepts, and engaged substantively in investigations using those models, using DeltaTick. We found that 22 (81%) of the 27 groups we analyzed constructed at least one model that reflected intended target ideas, and 17 (60%) of them explicitly engaged with the conceptual content of the problem, rather than simply treating the problem as a "puzzle" (Löhner et al., 2005, p. 458) by adjusting blocks and parameters without considering their underlying causal linkages (Table 2).

[INSERT TABLE 2 HERE]

We note that these findings represent patterns of model construction after only one class period of working with the DeltaTick tool. This is notable, since most studies report that it takes learners several weeks to learn a given modeling tool well enough to construct one's own (VanLehn, 2013), and that both these curricular topics (Population Dynamics and Natural Selection) are challenging, as we describe further below. In the following sections, we describe both curricular interventions in more detail, and provide detailed vignettes to illustrate patterns of successful use.

Modeling Natural Selection

Aditi and Uri (the second and third authors of this paper) have been using DeltaTick with middle school students to model micro-evolutionary change on a project called EvoBuild (Wagh & Wilensky, 2012). The main goal of EvoBuild is for students to explore how distributions of genetic variations in a population change over time due to natural selection and genetic drift (NGSS, 2013; NRC, 2012) through a series of model building activities.

We have pilot-tested the natural selection activity with students in a seventh grade science class in a large public middle school in Chicago. This middle school is one of the largest in the district, with a

socioeconomically diverse student population. Here, we focus on an excerpt featuring Andy and Nikhil, two students who were working together on the activity. The blocks for this activity had been designed to enable exploration of how the distribution of a trait, speed, in a prey species would shift over generations due to selection pressures from predators.

Example C1: Andy and Nikhil Explore Variation and Natural Selection. Andy and Nikhil started by adding two species called "Steve" and "Buddernaug" to their model. They added a "speed" trait to the "Buddernaug" species so that 20% of the Buddernaug population would be "very-slow", "slow", "medium", "fast" and "very-fast". They added the "move" behavior to both species and programmed steves to eat a buddernaug whenever they were physically next to it.

At this point, Andy and Nikhil decided to run their simulation. Given the rules assigned, Buddernaugs would each move at different fixed speeds until a Steve eats them. However, on running their model, they noticed an unexpected outcome.

1 [*Model begins*]

2 Nikhil: Yay!

3 Andy: There's a bunch of dots [*Steve agents*] eating them! (*Laughing*)

4 [*All the Buddernaug in the model get eaten*]

5 Nikhil: Now there's nothing left.

6 Andy: Oh wait--I forgot to put "reproduce." [*Looks at Nikhil*]

7 Nikhil: Oh yeah.

8 Andy: Well let's go put some reproduce.

Upon running the model, Andy and Nikhil noticed that the prey all died out (Lines 3-5). This led Andy to realize that the prey would need to reproduce in order to survive past one generation (Line 6). He went back to the Build tab, and added a reproduce behavior to both the predators and prey. They decided

to set the chance for reproduction to 100% for both species because they "didn't want to lose all of them" again.

Upon running the simulation, Andy noticed a missing rule. This led him to continue building up the simulation by adding a "reproduce" behavior. This instance of building up by adding a "reproduce" behavior is important for a couple of reasons. First, regardless of whether their motivation was to study variation across generations or simply to sustain the model over time, Andy and Nikhil explicitly noted reproduction as a core mechanism for their simulation. Second, as we will see below, this specific addition helped Andy and Nikhil bridge ideas of change at the level of individual behaviors, such as dying or eating prey (Line 3), to the idea of change in traits at the level of the population's characteristics later on in their investigation.

Prompted by their activity worksheet, Andy and Nikhil continued building up their model by adding a histogram to track how the distribution of the speed trait in prey changed over time (Figure 3a). This highlighted systematic differences in what buddernaug speed traits survived multiple generations.

9 [*Andy adds a histogram block on the Build tab, clicks on the Run tab, hits setup, notices*
10 *the histogram on the model interface, and starts running the model*]
11 Andy: Ah, this is kind of cool.
12 Nikhil: Well, that's sort of weird .. this one. *[One of the variations die out from the*
13 *population*]
14 Andy: Ah, one species died out.

On running their model, Nikhil and Andy were surprised to see one bar become smaller and then die out (Lines 12-14). Andy suspected that it was because there were too many predators, so he lowered the number of predators and ran the model again.

[INSERT FIGURE 3 HERE]

This time, Andy and Nikhil closely monitored the histogram to track the distribution of speeds as the model ran. They were surprised to see some speeds dying out from the population even after they had lowered the number of predators in the model. As they continued exploring, they eventually came to expect that certain variations would always die out, and turned their attention to *which* variations consistently survived, as seen in the excerpt below.

15 Andy: The slow ones are winning.
16 Andy: Oh my god, the dots [*Predators in their model are shaped like dots*] are so evil.
17 They're destroying them. Look at that. Nom, nom, nom. *[Laughing]*
18 Nikhil: Only two speeds remain. The fastest and the slowest. Fastest going to lose?
19 Andy: Let's see if they all die out because they're the only ones to eat. Can they
20 reproduce fast enough? Yeah, they-there's too many of the slow ones. I'm
21 surprised the slow ones -- *[Andy decides to increase the number of predators to*
22 *100]*

This time, Nikhil and Andy ran the model until only two variations remained in the population, but mostly slow ones (Lines 40, 46; Figure 3b). Presumably based on their previous model runs, Andy and Nikhil began to anticipate that the slow ones would survive (Line 33-35), though Andy was still surprised by this outcome (Line 37).

It is telling that Andy and Nikhil continued to be surprised that the slow members of the Buddernaug species survived at the *population* level: From an intuitive, individual's point of view, we might expect faster creatures to be able to more easily "escape" their predators than slower ones. However, in this simulation, creatures moved randomly and lacked an ability to sense a predator in order to "escape" more readily with speed. Andy and Nikhil's surprise, then, is likely indicative that they were working to break down the simulation to make sense of it: to connect an intuitive sense of predation

behavior – that faster is better for survival – to the multigenerational population-level patterns they observed in the histogram (even if the connection between these two ideas was not yet clear to them).

Andy and Nikhil began to work to make sense of this discrepancy: that is, to make sense of how and why the slow buddernaugs were surviving. They became curious to know if all the slow buddernaugs would die out because they were the only source of food, or if they would reproduce fast enough to remain in the simulation (Lines 35-38).

Later in the activity, Andy and Nikhil reasoned about the patterns they were noticing to a researcher in the class:

25 Andy: Certain ones are selected; the other ones just die off because they're hunted.
26 Nikhil: Yeah. The medium is sort of going good because it's neither too fast nor too
27 slow but then the fast are dying off because they run into the predators but
28 while the slow, they don't, because they just well, now the fast ones died, umm,
29 the slow one just like reproduced, making more slow ones than fast, see.

Pointing to the changing histogram while the model was running, Andy explained that some variations of bugs were getting selected while others were getting eaten by the predators (Line 49-50). Nikhil further noted a specific *individual* mechanism through which this *population* level pattern emerged: that the fast ones ran into predators and died, while the slow ones did not, and were hence able to reproduce to make more slow ones (Line 51-55).

In this vignette, Nikhil and Andy explored a core content idea, natural selection, through their investigation. Building up their simulation using content-specific primitives and breaking it down by debugging the model and tweaking parameters provided them with a window into a surprising outcome of this evolutionary mechanism. It also offered tools to help them make sense of it by connecting their expectations of *individual* level behavior – predation and death – to *population* level behavior across trait

variation and multiple generations. Andy and Nikhil were also able to pursue additional questions of interest by proposing and testing theories in their simulation, making explicit the mechanisms at work. This led them to a key learning goal, the differential distribution over time of speed in a prey species.

Overall, we found that 9 out of the 10 participating groups who worked on natural selection activities engaged in some kind of "building up", while 8 out of 10 engaged in some kind of "breaking down". Over the course of the activity, 9 out of the 10 participating groups successfully constructed simulations that exhibited the target curricular phenomenon, differential distribution of speed over multiple generations in the prey species. Four of the 10 groups explicitly engaged in the chain of causal interrelationships that make up this target phenomenon, like Andy and Nikhil did in this vignette.

Modeling Population Dynamics

Michelle and Uri have used DeltaTick to explore high school students' thinking and learning about emergent patterns of mathematical change in population dynamic systems (Wilkerson-Jerde & Wilensky, 2010). Our learning objectives were to engage students in recognizing key population growth patterns, and developing systematic ways to predict how different combinations of behaviors and interactions contribute to patterns of population change over time.

Typically, population growth patterns are represented mathematically with algebraic expressions that represent the collective influence of individual-level behaviors and interaction such as mating, birth, death, predation and competition. These parameters build on top of and influence one another. For example, the common Malthusian or exponential model of population growth can be constructed using a general rate of change for the population ($P(t)=P_0e^{rt}$), or a combination of birth and death rates ($P(t)=P_0e^{(b-d)t}$). It can be combined with carrying capacities or other environmental influences (the Verhulst model includes carrying capacity K: $P(t) = \frac{KP_0e^{rt}}{K+P_0(e^{rt}-1)}$.) Multiple populations can be modeled by relating parameters among one another to reflect predation, competition, or other interactions such as the Lotka-Volterra equations, which relate predator-prey dynamics (as cited in Wilensky & Reisman, 2006).

Using DeltaTick, we designed a library that includes sets of behaviors and conditions (constraining factors) that roughly correspond to each of the mathematical parameters that may or may not be included in a typical algebraic model of population. For example, patterns of growth or decay that approximate exponential models can be generated by giving a population rules to reproduce or die with a given probability, and patterns approximating logistic growth are generated by introducing spatial constraints on reproduction behaviors. One species of agent can be programmed to interact with another (see Appendix B for more information).

We used the population dynamics behavior library with 10th, 11th, and 12th graders in mathematics and science classes at two urban public high schools. One of our main goals was to get students to consider how different individual and population-level factors work *in interaction,* rather than just additively or in sequence, to generate a pattern of interest. For example, when space or age constraints are introduced to a simulation, the population may grow, shrink, or stabilize differently depending on the size or age distribution of the population over time. We were interested in how students made sense of these dynamics on both a behavioral and mathematical level.

Example C2: Mayra and Jessica Explore Interacting Causes for Population Growth. In this excerpt, AP Biology students Mayra and Jessica were tasked with finding combinations of behaviors that generate a pattern of relative stability in a population. Despite the fact that they could have used many possible combinations of behaviors to generate this pattern (many in the class, for example, added blocks that gave individual members of the population equal probabilities of birth and death to "even out" overall change in the population), Mayra and Jessica began by assembling a somewhat complex set of blocks – a 1% probability of reproduction for all members of the population, and a conditional 5% probability of death when population members were between the ages of 30 and 70. This combination of behaviors worked reasonably well to produce relative stability in the simulation, with only a small decrease relative to the total population. They decided to adjust the simulation to further stabilize the population:

1 Jessica: Let's do it with probability of .02 and make the death rate decrease it.

2 Mayra: They shouldn't die that much.

Jessica changed the values in all three of the boxes: she increased the probability of reproduction for all members of the population from 1% to 2%, narrowed the age limits in the green "if age-between?" box from 30-70 to 65-70, and decreased the probability of death for population members within this range of ages to 2% (Figure 4a). While they did not explicitly articulate why they made these particular decisions, one reasonable explanation is that they were working to moderate the drop in population they observed in their prior simulation by reducing the possibility for death (Lines 1, 2). After making these adjustments, Jessica executed the simulation for a second time. Figure 4b shows that the new simulation rules created dramatic exponential growth, which surprised the students. While the simulation ran, Jessica and Mayra continued to read prompts from their worksheet that encouraged them to consider what were the driving factors behind the pattern their simulation was producing.

[INSERT FIGURE 4 HERE]

3 Jessica: Oh, they're increasing.

4 Mayra: That's because they're also reproducing with a higher rate.

5 Mayra: [*Reads prompt aloud*: The overall rate of change in a population reflects a
6 number of different factors. In your model, what different factors are affecting
7 the rate of change?] Well, the probability of reproduction, because we
8 increased it at one point.

9 Jessica: And the death rate.

10 Mayra: The death rate. Especially with the different ages. So like...

11 Jessica: I think that reproduction affects it more than death.

12 Mayra: Yeah, for sure. Death rate, not that much. Because, it's going to be steady, you

13 know?

Initially, Jessica was surprised that the population increased so much more than it did in the prior simulation (Line 3). Mayra associated the pattern of growth with the increase from 1% to 2% in the probability that members of the population could reproduce (Line 6). When considering the question of which factors are influencing the pattern they are observing, both Jessica and Mayra argued that reproduction was affecting the pattern *more* than death. However, the modifications Jessica and Mayra introduced to the *death* behavior were more significant quantitatively than the changes they made to the *birth* behavior. They had severely reduced the number of population members who had a probability of dying from an interval of 40 years to only 5, and also cut the probability with which this population could die by 3%. Despite these changes, and both students' explicit acknowledgment of the death rate (Lines 11, 12), they still describe the system as dominated by one influence rather than the interaction of many.

With more prompting from their worksheets, Mayra and Jessica switched back to the "Build" interface and re-inspected the blocks they used to construct the simulation. This time, they began to attend more to the simultaneous interaction of factors that influenced the population level over time.

14 Mayra: Um...[*Reads prompt aloud*: how much does each factor affect the rate] Oh, we
15 just did that. Birth rate, I think, affects it more.
16 Jessica: Mmhmm.
17 Mayra: But then again, we also made this one slower. So they both kind of affect the
18 same...yes, but I mean, I think overall birth rate.
19 Jessica: Isn't the birth rate and the death rate the same? Did you make it the same for this
20 one?
21 Mayra: No.
22 Jessica: I thought you made them both .02.

23 Mayra: I thought I made one .025, [*Mayra switches back to "Build" tab to expose*
24 *simulation blocks*] but also I changed the ages. Yea, remember? I changed the
25 ages. Like from 65 to 75.

Here, Mayra and Jessica engaged in important content and practices related to population modeling by reflecting upon an existing model's components as laid out in the *DeltaTick* interface. When they encountered some of the basic mathematical patterns used in population modeling – relative stability caused by balanced inputs, and exponential growth, they began to make sense of how *combinations* of input behaviors can produce those patterns in new or unexpected ways by inspecting and comparing relative proportions of inputs and outputs in the simulation rules. There is also evidence that like the case of Andy and Nikhil, Mayra and Jessica made sense of these topics by connecting to other relevant knowledge: for example, they used age ranges that made sense given human lifespans. This balance of curricular focus and pedagogical connection made it possible for Mayra and Jessica to formulate commonsense rationales for what they are observing, and check their rationale against the resources they had available to them through the DeltaTick environment.

Overall, we found that 14 out of 17 the participating groups who completed the population dynamics activities engaged in "building up" strategies to create their population dynamics models, while 9 groups engaged in "breaking down" in a manner similar to Mayra and Jessica. Thirteen of the 17 groups successfully constructed curricular target models, simulations that exhibited specific patterns of birth, decay, relative stability, and/or logistic-like growth. Twelve of the 17 groups meaningfully engaged in an explicit discussion of the causal and quantitative interrelationships that underlied these patterns, like Mayra and Jessica in this vignette.

Findings Part II: Using DeltaTick to Address Pedagogical Needs

While the examples above reflect the ways in which curricular libraries have permitted students to explore target patterns and interactions for a given topic of interest, we have also had experiences where those libraries had to be refined in unexpected ways, in response to the particular knowledge, experiences, or interests of the students we are working with. As noted earlier, we did not explicitly design for this at first, but found that rather than trying to *eliminate* these differences, we used DeltaTick to *respond* to them in situation-specific ways. When we returned to our data to systematically document instances where we made adaptations or there was notable potential for adaptations to address student needs or concerns, we found that opportunities to leverage DeltaTick features to refine activities in response to student needs and interest emerged persistently across enactments (Table 4).

[INSERT TABLE 4 HERE]

We found instances in both studies where we as facilitators adapted blocks in response to what we noticed in the classroom, and where students volunteered new behaviors or ways of measuring their models that were, or could have been, added to the existing behavior libraries with which they were working. In the high school population dynamics study, we also found instances where facilitators or learners accessed and modified underlying NetLogo code in ways that were directly motivated by our own efforts to respond to student needs and help support their explorations. Here we present three specific vignettes (indicated in Table 4 with superscript) that illustrate how the flexibility built into DeltaTick allowed us to (1) respond to unexpected patterns in student behavior, (2) provide extensions to allow students to pursue their own projects and questions, and (3) enable students to identify and modify the basic assumptions of a given library.

Example P1: Responding to Unexpected Patterns in Student Behavior through Adaptation

In DeltaTick, it is relatively easy to remove or rename blocks in a behavior library quickly by editing the text library file. Although these types of quick modifications might seem trivial at first, we

have found them to be essential for focusing student attention, and recasting behaviors available in the library in ways that better connect to students' ways of describing the phenomena they are exploring. Here we share a vignette that illustrates how seemingly minor and straightforward changes to a curricular behavior library file helped circumvent unanticipated issues and better connect to student knowledge.

On Day 1 of the Natural Selection study, Aditi was scheduled to co-teach consecutive sections of a science class on the same day. The first activity had been designed to introduce students to DeltaTick and to have them build a simple model to investigate how the distribution of speed (a trait labeled as "step-size" in the library) in a prey species would change over time due to selection pressures (much like the investigation described in C1). Though this activity required students to use only one trait in their simulation, "step-size", the accompanying behavior library included three additional traits: vision, hearing-range, and body-size. These were included to serve as examples of traits students could explore later, and the behaviors in the library did not include parameters to work with them.

When students in the first class started working on their model, Aditi noticed two unanticipated issues. First, many students tried to use all available traits in their simulation, and found it confusing that the behaviors in the library did not include explicit parameters for them. For instance, after adding the trait "vision" to their predator species, some students tried to use the vision block in the behavior, "chase" to enable a predator to chase a prey on spotting it. When they could not find a parameter for vision in the behavior, they assumed that the construction environment would automatically specify the trait for them. Because this was the first time that students were using DeltaTick, this obfuscated some students' understanding of the construction environment and subsequently, the goal of the activity. Second, Aditi noticed that several students wanted to add prey species that could swim or fly, and were dissatisfied with the name of the trait "step-size".

Recognizing both these issues as relevant to how students were making sense of the construction environment and activity, Aditi modified the corresponding EvoBuild behavior library in the 10 minutes

before the next section. She changed the trait name of "step-size" to "speed" (Figure 5) and removed the additional traits by deleting their code segments. In subsequent sections that day, students focused on one trait to more quickly learn how the environment worked, and observe patterns in the changing distribution.

[INSERT FIGURE 5 HERE]

Example P2: Supporting Student Investigation through Extensions

DeltaTick's design also allows new blocks or extensions to be made available for students within the structure of an existing library. During the first population dynamics enactment using DeltaTick, [Name removed] noticed students discussing or mentioning to facilitators other types of phenomena such as disease spread that could be modeled using the mathematical patterns and behavior blocks they were already exploring. She noticed that many of these recommendations could be accommodated relatively easily with only a few additions or modifications to the existing population dynamics library.

In a subsequent enactment of population dynamic activities, Michelle asked students more formally to make recommendations about what sorts of extensions they would like to explore – this time, with the goal of providing them enhanced libraries on the final day. Suggestions included explorations of the spread of disease (29% of students), predator/prey dynamics (16%), the effects of birth control and sex/gender dynamics on population growth patterns (10%), employment dynamics (10%), and the influence of environmental factors on population and reproduction (8%). Some recommendations were especially relevant to students' lives and current events, such as a suggested "Oil Spill Library" to explore the effects of the infamous *Deepwater Horizon* oil rig accident in the U. S. Gulf Coast, which had happened only a few months before. Figure 6 shows the Spread of Disease library, one of the libraries we were able to provide based on their suggestions. Those blocks that were added to the original population dynamic library students had worked with are highlighted in the figure.

[INSERT FIGURE 6 HERE]

Given the time constraints of the activity, students were only able to explore these newly constructed libraries during their last day of the *DeltaTick* enactment. However, given that the libraries were familiar in that they were simply extensions of those students had already used, students were comfortable beginning these new, interest-driven investigations. At the very least, extending their investigations got students thinking about how the ideas they were exploring might be relevant to a larger collection of topics than what we had time to explore in class.

Example P3: Uncovering and Modifying a Library's Lower-Level Assumptions

While Examples 1 and 2 illustrate DeltaTick's flexibility through the modification of behavior libraries, there were also times when students' questions were best addressed by allowing them to "look inside" and directly edit the NetLogo code generated by DeltaTick blocks. One example of this occurred during Michelle's population dynamics work, when one student group noticed an interesting pattern in their simulation which they believed was related to the age of population members.

Vince and Joey had constructed a simulation in which agents between the ages of 15 and 55 had a probability of reproducing, and agents over the age of 65 had a probability of dying. However, when the simulation was run, agents reproduced immediately, and the population level stabilized after 15 time units had passed. Vince and Joey were surprised because they expected that no agents would reproduce until at least 15 time units had passed, and no agents would die until at least 65 time units had passed. As he inspected the blocks, Vince began to question how individual agents' age (which should be controlling when they reproduce or die) was initialized and used in the simulation.

1 Vince: What? But like when they reproduce, I don't think this program takes into account
2 like when they reproduce, there's a new, like it starts at 0, you know? Like there's
3 a new life.
4 Joey: So you think they reproduce and stay at 15?

5 Vince: Yea

6 Joey: So like if a 15 year old has a baby the baby comes out at 15?

7 Vince: 15. I don't know. But that's what it seems like.

What Vince and Joey were struggling with was an artifact of one of the major design tradeoffs made in constructing the population dynamics library. We included "behind the scenes" code so that at the beginning of each simulation, individual agents would start with a random age between 0 and 50. This was because we wanted students to consider population as a system that involved multiple, stochastic, distributed events that *accumulated* to create a pattern of interest, rather than what might result if the population acted as one. We also expected that starting a population with heterogeneous ages would be more aligned with students' expectations for how a population should behave.

The unexpected patterns Vince and Joey were observing were a byproduct of the specificities of our initial random distribution of ages. When the simulation started, all population members were between the ages of 0 and 50; and therefore many of them were immediately old enough to reproduce. Starting 15 time units into the simulation execution, the oldest of the population members finally became "eligible" for probabilistic death, at age 65 as Vince and Joey had assigned. They were confused by this pattern and its apparent disconnection from the rules used to construct the simulation, and wondered whether the age of newly born agents might have something to do with the confusing pattern. The students decided to run the model again, stepping through each time unit slowly to confirm sure that they were observing birth and death at unexpected times.

8 Vince: I just don't get why they're reproducing right away.

9 Joey: Just do steps for 15 years, and see if they're reproducing. [*Clicks the "step"*

10 *button in the simulation interface.*] See they're already reproducing, and they

11 should only be two years old.

12 Vince: Wait, are they?

13 Joey: Yea, because that's [*indicating graph*] a change, so it went up two.

Vince and Joey consulted classroom facilitators Michelle and Josh, another research assistant. Michelle told them that the simulation begins with a random distribution of ages 0-50. Josh stayed with them and opened the simulation code to show where this distribution was defined, and how they could edit it so that all agents started at age 0 (Figure 7).

[INSERT FIGURE 7 HERE]

This modification would have been impossible to make from the block-based interface. It also would not have made sense to edit the entire library for future classes, as Aditi had done in Example 1. It was a response that made sense only given the particular simulation Vince and Joey had decided to build and explore, and helped them maintain ownership of and investment in their exploration. After simplifying some of the simulation's underlying code, Vince and Joey could re-focus on disentangling aspects of the simulation's behavior that they found confusing and unexplained (the earlier immediate increase in population at time 0; the dramatic shift in population dynamics at year 14) from dynamics that they expected to be tightly connected to the blocks they used to construct the model.

Discussion

This study is based on our own use and refinement of DeltaTick and its particular design features. Here, we explore how the lessons we have learned can inform the design of K-12 modeling toolkits more broadly, and shed light on the role they can play within the broader classroom learning ecology.

Design Principles for K-12 Computational Modeling Toolkits

Based on our analysis of the ways in which DeltaTick supported curricular and pedagogical goals, we posit two general principles, *Curricular Example Spaces* and *Levels of Responsivity,* for the design of computational construction environments intended specifically for classroom use. Table 2

illustrates how the vignettes presented above link to and inform the design principles we are introducing. Below, we describe each in further detail.

[INSERT TABLE 2 HERE]

Curricular Example Spaces

As we reviewed in the Background, a major decision in the design of computational modeling toolkits involves the "granularity" of building blocks. Based on the ways we found students were using *DeltaTick*, we propose the notion of a *curricular example space* to help guide this decision. We use the term *curricular example space* to refer to a space of possible simulations that together, provide examples of the core dynamics of a curricular domain. For example, in the Population Dynamics behavior library (Example C2), the available primitives supported a curricular example space that included a collection of important mathematical population growth models – including exponential-like growth and decay, logistic-like growth, and relative stability.

To determine what construction blocks are appropriate for a given domain of study, we suggest that designers first consider this curricular example space as representing the set of simulations learners should be able to construct. Ideally, this example space should be sufficiently complex that it produces dynamics that are surprising or unexpected to learners. Then, designers should decompose those examples into component elements that when used in isolation can also generate working simulations whose behaviors are relatively straightforward. This accommodates both approaches we have identified in students: "building up" simulations by incrementally adding one or two construction units at a time and making sure they make sense (as in Example C1), and "breaking down" complex simulations that are already constructed to understand how and why those simulations generated surprising or unknown behavior (as in Example C2). It also allows more complex target investigations to be mapped to simpler behaviors that are accessible and familiar to students. Appendix B illustrates this approach in more detail using the Population Dynamics Library as an example.

Levels of Responsivity

Research in science and mathematics education has shown that effective classroom practice requires teachers to attend to and respond to their students' thinking (Carpenter, Fennema, Peterson, Chiang, & Loef, 1989; Duncan, Rogat & Yarden, 2009; Franke & Kazemi, 2001; Levin, Hammer & Coffey 2009; Sherin & van Es, 2005). This work has focused primarily on patterns of discourse – that is, how teachers orchestrate student discussions and establish disciplinary norms in the classroom. However, there is growing interest in exploring responsive *curricula* or how pedagogical materials can be developed to support teachers' attention to student knowledge, questions, and interests, and encourage refinement or changes to materials based on their findings (Berland & Hammer, 2012; Gallas, 1995). DeltaTick is an example of how computational tools might serve such a responsive role. We use the term *responsivity* to refer to the potential for educators to modify aspects of a computational learning environment – in this case, a behavior library, specific blocks, or even code itself – in direct response to student thinking and interests. We distinguish *responsivity* as enacted by educators themselves from *responsiveness* of a technology, where computational environments adjust based on user input (such as in intelligent tutoring systems or adaptive testing applications).

The case of DeltaTick extends the notion of responsive teaching to the design of computational modeling toolkits, as meditators of curriculum and discourse in the classroom environment. We are finding that such toolkits need not, and should not, be a static fixture of classroom curricula or pedagogy. They can be modified in response to student needs in the same way that teachers' responses to students and use of curricular materials are. Furthermore, our experiences with DeltaTick suggest that the potential connections and conflicts between students' own understandings and the tools they use to express and share those understandings might unfold across particular levels of analysis. This is why we add to the notion of responsivity the idea of *levels* – specific units of analysis that might link to different ways of noticing and responding to students. For example, we found that we responded to student questions about

underlying model assumptions and mathematical detail by accessing and modifying code directly, responded to interests by extending existing behavior libraries, and worked to more directly connect to learners' existing ways of thinking about the phenomena by editing exiting blocks to use new or different language.

Reconceptualizing the Role of Modeling Toolkits Within the K-12 Classroom Ecology

Modeling toolkits are typically described as relatively static elements within a dynamic classroom ecology, requiring specially aligned curricula, supports, and pedagogies to be used effectively (Fretz, et al. 2002; Krajcik et al., 1998; Louca & Zacharia, 2012; VanLehn, 2013). While we agree that effective modeling activities require such alignment and support, our findings and resultant principles suggest a reconceptualization of how this alignment can be achieved. Rather than ossified classroom artifacts that must be 'worked around', modeling toolkits themselves can be sufficiently flexible that they, like curricula or teacher attention, can be adapted over time to accommodate given curricular and pedagogical needs. Reconceptualizing modeling toolkits as flexible rather than static components of the broader classroom learning ecology can help address a number of persistent issues in the literature on K-12 computational modeling for STEM education.

Issue 1: Usability and Accessibility of Modeling Toolkits

While there is general consensus that computational modeling activities hold promise for K-12 STEM education, some researchers question whether the modeling paradigms available today are best suited for K-12 learners. Some have cited the grain size of primitives as a problem: for example, some argue that learners might struggle to connect the abstract representations used in system dynamics or concept mapping tools to particular domains (Doerr, 1996), others that tools which foreground low-level interactions such as agent-based environments may require too much coordination to construct complex models (Hmelo-Silver & Azevedo, 2009). Others have noted that the representational type and level of specificity available for a given modeling paradigm might constrain or preclude student explorations

(Louca & Zacharia, 2008; Louca, Zacharia, Michael, & Constantiou, 2011). Visual paradigms often do not enable a high degree of specificity and hence allow learners to explore a problem space and model the entities and interactions that define a system, but limit their ability to specify and test relations and variable parameters. Conversely, text-based tools often require precision from the outset, which may preclude learners' ability to get started in a modeling task (Löhner, Van Joolingen, & Savelsbergh, 2003). To address these issues, some have advocated for the design of modeling toolkits that include one or more "meso-levels" (Schwartz, p.169, 2007) of primitives, or sets of linked models (Fredericksen & White, 2002), that allow learners to explore the same phenomenon at multiple levels of analysis. Others have suggested that different paradigms may be more appropriate at different times during the modeling process (Löhner et al., 2003; Wilkerson-Jerde, Gravel & Macrander, 2014).

Grain size and specificity, which are often pre-defined and fixed by the notational system and primitives in computational toolkits and contribute to their accessibility and utility in the classroom, become *fluid* within environments such as DeltaTick. We were able to make adaptations such as adding or removing adjustable quantitative parameters (like the ones text or formula-based tools require) from a behavior block to allow more or less specificity when needed by students, or consolidate or decompose behaviors into more or less granular primitives to give students access to different levels of analysis. When working with flexible toolkits, the issues raised above become ways to focus one's attention on student knowledge and respond accordingly over time, rather than issues that constrain or preclude successful exploration.

Issue 2: Supporting Meaningful Curricular Investigations

Concerns have also been raised about the degree to which modeling toolkits adequately connect to or support meaningful exploration of curricular goals. Some studies have found that when working with modeling toolkits, learners may not focus on a system's underlying structure and instead focus on reproducing target patterns (Löhner et al. 2005). Or, they may add components to their model arbitrarily

in an effort to generate the desired effect without considering the actual phenomenon being modeled (Hogan & Thomas, 2001; Löhner et al., 2003; Metcalf et al., 2000). In an effort to strengthen curricular connections, some tools compare learners' models to expert solutions (Bravo, van Joolingen, & de Jong, 2006), and/or use domain-specific toolkits (Sengupta et al., 2013). But while these approaches better align with curriculum, they may further limit students' ability to pursue their own questions or extensions. For example, Hansen and colleagues used a 3D astronomy-specific toolkit and successfully increased student performance on spatially related astronomical concepts (2004a). However, they found that limitations of the tool prevented learners from exploring further questions, and may have interfered with their learning of concepts not directly related to the modeling environment (2004b).

When we designed DeltaTick, our original intention was to connect to curricula by designing block libraries that featured natural language representations of particular domains. Despite this attempt to make curricular connections accessible and explicit through careful *a priori* design, we still found ourselves making modifications *in situ* to clarify those connections, or to permit learners to pursue new questions and extensions. Flexible modeling toolkits yield opportunities to make modifications in ways that align with already established techniques for helping learners clarify connections between a language and a domain such as negotiating what a given notation in their model should represent, or creating their own names for primitives (VanLehn, 2013). They allow such modifications to be enacted and evolved within the toolkit itself.

Issue 3: Teacher Attention and Support

Finally, research has problematized the critical role of teachers and facilitators in supporting classroom computational modeling activities. To enact computational modeling activities, teachers must help learners become adept with the software, make progress in constructing models, and make connections between the model and their existing knowledge (Doerr, 2007; Webb, 1994). Furthermore, they must help students understand the broader goals of modeling as a scientific practice in itself. Louca,

Zacharia, and Constantinou (2011) explored how teachers and toolkits influence what *modeling frames* learners adopt—that is, whether they focus on (1) describing a phenomenological system, (2) operationalizing its underlying processes, or (3) constructing algorithms to define the relationships between objects and processes in the system. They found that the modeling toolkit they used, StageCast, helped foreground certain modeling frames and shifts among frames, while others required additional teacher support. They argued that these complementary roles of the teacher and toolkit were critical, since learners needed to maintain balance across these different frames in order to iterate and make progress in the modeling activity.

Flexible and adaptable modeling toolkits re-conceptualize the complementary roles of the teacher and toolkit as interconnected and overlapping by enabling the enactment of teacher roles *through* the toolkit. For instance, as classroom facilitators, we found ourselves addressing many of the roles identified in the literature: helping learners use DeltaTick effectively, identifying ways they could shift modeling frames in order to improve, test, or revise their models, and encouraging them to focus on new or different relationships in the systems they were modeling. However, we did this not only through direct interaction with students, but also through direct modifications to DeltaTick toolkit. In environments where curricular alignment and responsivity are explicitly considered, teachers' roles are directly supported by, and can be directly enacted through, the modeling toolkit itself.

Future Work

Our findings and discussion point to a number of areas for future work, and have provided useful frameworks to guide that work. First, we must better understand how to support educators in learning how to adapt flexible construction toolkits like *DeltaTick*, and to recognize when it is appropriate to do so. One aspect of this involves helping teachers develop the computational and technological competency required to edit library files, modify program code, and recognize those student interests and requests that can be reasonably pursued within the infrastructure of a given library. Another involves helping teachers

recognize what productive student practices look like within these construction environments, and how to respond to and foster those practices – both outside of the computational tool as well as through direct edits and customizations to the tool itself.

While these technological and pedagogical competencies are not yet typical expectations for teachers, knowledge of the relationship between technological tools, pedagogy, and the content and practices of a discipline is an increasingly important and unique aspect of teacher practice (Koehler & Mishra, 2006). In fact, a review of professional development programs suggests that allowing teachers to make their own customizations to tools and curricula can support effective long-term adoption of technology-mediated inquiry tools into classroom practice (Gerard et al., 2011). Environments such as NetLogo and others have made significant progress in reducing the barrier to programming entry for both students and teachers (Rodger, et al, 2009). And growing support for the development of technological and pedagogical competencies is buttressed by an increasing recognition by the educational community that computational modeling plays an important role not only pedagogically, but also as a fundamental component of scientific practice (NRC, 2010; NGSS, 2013).

Second, the themes that have emerged from this work can set the stage for a deeper exploration of how technological tools can build on and support student knowledge, curricular materials, and teacher attention *in situ* and in interconnected ways, rather than only *a priori* and in complementary ways. We found ourselves revising the DeltaTick environment in distinct ways based on what we noticed in students. We responded to students' ways of describing phenomena in the system they are exploring by modifying existing computational components (as in example P1, where the name of a block was revised), encouraged proposed extension investigations by adding more construction blocks or creating new libraries (as in example P2, where new libraries were developed), and helped them interrogate *underlying assumptions* of a given computational model (as in example P3, where students worked with facilitators to edit specific age-related parameters in the text code of the simulation). It would be

interesting to study the extent to which these types of modifications are needed for different domains or model frames. Similarly, it would be interesting to explore the extent to which classroom teachers attend and respond to these different types of needs.

Conclusion

We have argued that when designing computational modeling toolkits for K-12 mathematics and science classrooms, it is critical to reflect on, and find ways to help educators navigate, the tension between *curricular* goals that emphasize particular topics and content areas within mathematics and science, and *pedagogical* goals such as building upon and responding to students' own knowledge, needs, and intellectual pursuits. While many computational modeling toolkits have focused on one or the other of these needs, our experiences with DeltaTick suggest that it is possible to do both, by designing libraries that represent curricular domains as *example spaces* that students could reconstruct and explore, and by offering *layers of responsivity* that allow educators to ensure that libraries connect to students' own knowledge, interests, and needs. In classroom enactments, students used DeltaTick to successfully construct and explore computational models in multiple complex domains of study; and we as facilitators were able to adapt the toolkit in response to students' immediate and longer-term needs. These findings and principles point to a reconceptualization of modeling toolkits as components of a classroom ecosystem that, like curricula or teacher attention, can be adapted over time to accommodate given curricular and pedagogical needs. This reconceptualization highlights new ways modeling toolkits can be used to better align with, build on, and support student knowledge, curricular materials, and teacher attention during classroom computational modeling activities.

References

Ackermann, E. (1996). Perspective-taking and object construction: Two keys to learning, In Y. Kafai & M. Resnick (Eds.) *Constructionism in Practice: Designing, thinking, and learning in a digital world* (pp. 25-37). Mahwah, N.J.: Lawrence Erlbaum Associates, Inc.

Ackermann, E. (2001). Piaget's constructivism, Papert's constructionism: What's the difference? Retrieved October 2011, from http://learning.media.mit.edu/publications.html

Bailey & Borwein. (2011). Exploratory experimentation and computation. *Notices of the American Mathematical Society, 58*(10), p. 1410-1419.

Ball, D. L., & Bass, H. (2000). Interweaving content and pedagogy in teaching and learning to teach: Knowing and using mathematics. In J. Boaler (Ed.) *Multiple perspectives on the teaching and learning of mathematics* (pp. 83-104). Westport, CT: Ablex.

Bamberger, J. (1996). Turning music theory on its ear do we hear what we see; do we see what we say? *International Journal of Computers for Mathematical Learning, 1*(1), 33-55.

Basawapatna, A., Repenning, A., & Lewis, C. (2013). The simulation creation toolkit: An initial exploration into making programming accessible while preserving computational thinking. *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, 501-506.

Berland, L. K., & Hammer, D. (2012). Students' framings and their participation in scientific argumentation. In M. S. Khine (Ed.), *Perspectives on scientific argumentation* (pp. 73-93). Springer Netherlands.

Berland, L. K., & Reiser, B. J. (2009). Making sense of argumentation and explanation. *Science Education, 93*(1), 26-55.

Bers, M. U., & Horn, M. S. (2009). Tangible programming in early childhood: Revisiting developmental assumptions through new technologies. In I. R. Berson & M. J. Berson (Eds.), *High-tech tots: Childhood in a digital world*. Greenwich, CT: Information Age Publishing.

Blikstein, P., & Wilensky, U. (2009). An atom is known by the company it keeps: A constructionist learning environment for materials science using multi-agent simulation. *International Journal of Computers for Mathematical Learning, 14*(1), 81 – 119.

Bollen, L., & van Joolingen, W. R. (2013). SimSketch: Multi-agent simulations based on learner-created sketches for early science education. *IEEE Transactions on Learning Technologies, 6*(3), 208-216. doi:10.1109/TLT.2013.9

Brown, A. L. (1992). Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences, 2*(2), 141-178.

Carpenter, T. P., Fennema, E., Peterson, P. L., Chiang, C.-P., & Loef, M. (1989). Using knowledge of children's mathematics thinking in classroom teaching: An experimental study. *American Educational Research Journal, 26*(4), 499–531.

[CCSSO] Council of Chief State School Office. (2010). *Common core state standards for mathematics.* Retrieved July 2013 from http://www.corestandards.org/assets/CCSSI_Math Standards.pdf

Chandrasekharan, Nersessian, & Subramanian. (2013). Computational modeling: Is this the end of thought experiments in science? In M. Frappier, L. Meynell, & J. R. Brown (Eds). *Thought experiments in philosophy, science and the arts* (pp. 239-260). London: Routledge.

Clark, D., Nelson, B., Sengupta, P., & D'Angelo, C. (2009). Rethinking science learning through digital games and simulations: Genres, examples, and evidence. In M. Honey & M. Hilton (Eds). *Learning Science through Computer Games and Simulations*. National Academy of Sciences, Washington, DC..

Cobb, P., Confrey, J., Disessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational Researcher, 32*(1), 9.

Collins, Joseph, & Bielaczyc. (2004). Design research: Theoretical and methodological issues. *Journal of the Learning Sciences, 13*(1), 15-42.

Conway, M. J. & Pausch, R. (1997). Alice: easy to learn interactive 3D graphics. *ACM SIGGRAPH Computer Graphics*, *31*(3), 58-59.

diSessa, A. A. (1997). Open toolsets: New ends and new means in learning mathematics and science with computers. In E. Pehkonen (Ed.), *Proceedings of the 21st Conference of the International Group for the Psychology of Mathematics Education* (Vol. 1). Lahti, Finland, 47-62.

diSessa, A. A. (2001). *Changing minds: Computers, learning and literacy*. Cambridge, MA: MIT Press.

diSessa, A. A., & Abelson, H. (1986). Boxer: a reconstructible computational medium. *Communications of the ACM, 29*(9), 859-868.

Derry, S., Pea, R., Barron, B., Engle, R., Erickson, R., Goldman, R., Hall, R., Koschmann, T., Lemke, J. L., Sherin, M. G., & Sherin, B. L. (2010) Conducting video research in the learning sciences: Guidance on selection, analysis, technology and ethics. *Journal of the Learning Sciences, 19*(1), 3-53.

Dewey. (1904). The relation of theory to practice in education. In C. A. McMurry (ed.), *The relation of theory to practice in the education of teachers* (Third Yearbook of the National Society for the Scientific Study of Education, Part I). Bloomington, IL: Public School Publishing.

Duncan, R. G., Rogat, A. D., & Yarden, A. (2009). A learning progression for deepening students' understandings of modern genetics across the 5th–10th grades. *Journal of Research in Science Teaching, 46*(6), 655-674.

Edelson, D. C. (2001). Learning-for-use: A framework for the design of technology-supported inquiry activities. *Journal of Research in Science Teaching, 38*(3), 355-385.

Franke, M. L., & Kazemi, E. (2001). Learning to teach mathematics: Focus on student thinking. *Theory into practice, 40*(2), 102-109.

Gallas, K. (1995). *Talking Their Way Into Science: Hearing Children's Questions and Theories, Responding with Curricula.* New York, NY: Teachers College Press.

Gerard, L. F., Spitulnik, M., & Linn, M. C. (2010). Teacher use of evidence to customize inquiry science instruction. *Journal of Research in Science Teaching, 47*(9), 1037-1063.

Gilbert, J. K. (2005). *Visualization in science education.* Dordrecht, the Netherlands: Springer.

Goldstone, R. L., & Wilensky, U. (2008). Promoting transfer by grounding complex systems principles. *Journal of the Learning Sciences, 26*(1), 465 – 516.

Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W. M., Railsback, S. F., DeAngelis, D. L. (2005). Pattern-oriented modeling of agent-based complex systems: Lessons from ecology. *Science, 310*(5750), 987-991.

Hammer, D. (1997). Discovery learning and discovery teaching. *Cognition and instruction*, *15*(4), 485-529.

Harel, I. E., & Papert, S. E. (1991). *Constructionism*. Westport, CT: Ablex Publishing.

Harvey, B., & Mönig, J. (2010). Bringing "No ceiling" to scratch: Can one language serve kids and computer scientists? *Proceedings of Constructionism 2010,* 1-10.

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

Horn, M. S. and Jacob, R. J. K. (2007). Tangible Programming in the Classroom with Tern. *Proceedings of CHI'07 ACM Human Factors in Computing Systems (CHI Trends Interactivity)*, ACM Press.

Horn, M. S., & Wilensky, U. (2012). NetTango: A mash-up of NetLogo and Tern. In T. Moher (chair) and N. Pinkard (discussant), When systems collide: Challenges and opportunities in learning technology mash-ups. Symposium presented at the annual meeting of the American Education Research Association. Vancouver, British Columbia, Canada, April 13 – 17.

Hoyles, C., & Lagrange, J. B. (Eds.). (2010). *Mathematics education and technology: rethinking the terrain: the 17th ICMI study* (Vol. 13). Dordrecht, the Netherlands: Springer.

De Jong, T., & Van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research, 68*(2), 179-201.

Kafai, Y.. (2006). Constructionism. In K. Sawyer (Ed.), *Cambridge handbook of the learning sciences* (pp. 35-46). New York: Cambridge University Press.

Kahn, K. (1998). ToonTalk—An animated programming environment for children. *Journal of Visual Languages & Computing, 7*(2), 197-217.

Kahn, K. (2007). Building computer models from small pieces. In *Proceedings of the 2007 Summer Computer Simulation Conference*. 931-936.

Khan, S. (2008). The case in case-based design of educational software: A methodological interrogation. *Educational Technology Research & Development, 56*(4), 423-447.

Kali, Y., & Linn, M. C. (2007). Technology-enhanced support strategies for inquiry learning. In J. M. Spector, M. D. Merrill, J. van Merrienboer, & M. P. Driscoll (Eds.), *Handbook of Research on Educational Communications and Technology: A Project of the Association for Educational Communications and Technology* (3rd Edition, pp. 145-162).

Kaput, J. (1994). Democratizing access to calculus: New routes to old roots. *Mathematical Thinking and Problem Solving,* 77-156.

Ketelhut, D. J., Nelson, B. C., Clarke, J., Dede C. (2010). A multi-user virtual environment for building and assessing higher order inquiry skills in science. *British Journal of Educational Technology*

Klopfer, E., Yoon, S., & Um, T. (2005). Teaching complex dynamic systems to young students with StarLogo. *Journal of Computers in Mathematics and Science Teaching, 24*(2), 157-178.

Levin, D. M., Grant, T., & Hammer, D. (2012). Attending and responding to student thinking in science. *The American Biology Teacher, 74*(3), 158-162.

Levin, D. M., Hammer, D., & Coffey, J. E. (2009). Novice teachers' attention to student thinking. *Journal of Teacher Education*, *60*(2), 142-154.

Louca, L. T., & Zacharia, Z. C. (2008). The use of computer-based programming environments as computer modelling tools in early science education: The cases of textual and graphical program languages. *International Journal of Science Education, 30*(3), 287-323.

Manz, E. (2012). Understanding the codevelopment of modeling practice and ecological knowledge. *Science Education, 96*(6), 1071-1105.

Metz, K.E. (1997). On the complex relation between cognitive developmental research and children's science curricula. *Review of Educational Research. 67* (1), 151-163.

Metz, K. E., Sisk-Hilton, S., Berson, E., & Ly, U. (2010). Scaffolding children's understanding of the fit between organisms and their environment in the context of the practices of science. In *Proceedings of the International Conference of the Learning Sciences*.

[NCTM] National Council of Teachers of Mathematics. (2000). *Principles and Standards for School Mathematics.* Reston, VA: NCTM.

[NRC] National Research Council. (2007). *Taking science to school: Learning and teaching science in grades K-8.* Washington, D.C.: The National Academies Press.

[NRC] National Research Council. (2010). *Report of a workshop on the scope and nature of computational thinking.* Washington, D.C.: The National Academies Press.

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

[NGSS] Next Generation Science Standards. (2013).

Noss, R., Healy, L., & Hoyles, C. (1997). The construction of mathematical meanings: Connecting the visual with the symbolic. *Educational Studies in Mathematics, 33*(2), 203-233.

Papert, S. (2004). Keynote Speech. In E. McKay (Ed.), *Proceedings of the International Conference on Computers in Education (ICCE)*. Sydney: University of Sydney. http://vimeo.com/9092144

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning, 1*(1), 95-123.

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

Papert, S. (1993). *The children's machine: Rethinking school in the age of the computer.* New York: Basic Books.

Parnafes, O., & Disessa, A. (2004). Relations between types of reasoning and computational representations. *International Journal of Computers for Mathematical Learning, 9*(3), 251-280.

Piaget, J. (1952). *The origins of intelligence in children.* New York: International Universities Press.

Pratt, D. (1998). The co-ordination of meanings for randomness. *For the learning of mathematics, 18*(3), 2-11.

Repenning, A., & Sumner, T. (1995). Agentsheets: A medium for creating domain-oriented visual languages. *Computer, 28*(3), 17-25.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., … Silverman, B. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60-67.

Rodger, S. H., Hayes, J., Lezin, G., Qin, H., Nelson, D., Tucker, R., Lopez, M., Cooper, S., Dann, W. & Slater, D. (2009). Engaging middle school teachers and students with alice in a diverse set of subjects. In *ACM SIGCSE Bulletin* (Vol. 41, No. 1, pp. 271-275). ACM.

Roschelle, J. M., Pea, R. D., Hoadley, C. M., Gordin, D. N., & Means, B. M. (2000). Changing how and what children learn in school with computer-based technologies. *The Future of Children, 10*(2), 76-101.

Sabelli, N. H. (2006). Complexity, technology, science, and education. *Journal of the Learning Sciences, 15*(1), 5.

Schwartz, J. L. (2007). Models, simulations, and exploratory environments: A tentative taxonomy. In. R.A. Lesh, E. Hamilton, & J.J. Kaput (Eds.), *Foundations for the Future in Mathematics Education* (161-172). Mahwah, NJ: Lawrence Eribaum Associates.

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, M., & van Es, E. (2005). Using video to support teachers' ability to notice classroom interactions. *Journal of technology and teacher education, 13*(3), 475-491.

Sherin, B., diSessa, A., & Hammer, D. (1993). Dynaturtle revisited: Learning physics through collaborative design of a computer model. *Interactive Learning Environments, 3*(2), 91-118.

Simon, M. A. (1995). Reconstructing mathematics pedagogy from a constructivist perspective. *Journal for Research in Mathematics Education 26*(2), 114-145.

Simpson, Hoyles, & Noss. (2005). Designing a programming-based approach for modelling scientific phenomena. *Journal of Computer Assisted Learning, 21*(2), 143-158.

Smith, D. C., Cypher, A., & Tesler, L. (2000). Programming by example: Novice programming comes of age. *Communications of the ACM, 43*(3), 75-81.

TechSmith Corporation (2010). Camtasia [Computer Software].

Wagh, A., & Wilensky, U. (2012). Evolution in blocks: Building models of evolution using blocks. In C. Kygnios, J. Clayson, & N. Yiannoutsou (Eds.), *Proceedings of the Constructionism 2012 Conference* (pp. 549 – 554), Athens, Greece.

White, B., & Frederiksen, J. (2005). A theoretical framework and approach for fostering metacognitive development. *Educational Psychologist, 40*(4), 211-223.

Wilensky, U. (1995). Paradox, programming and learning probability. *Journal of Mathematical Behavior, 14*(2), 253 – 280.

Wilensky, U. (1999). NetLogo Models Library [Computer Software]. Center for connected learning and computer-based modeling. Evanston, IL: Northwestern University. Available from http://ccl.northwestern.edu/netlogo/models/.

Wilensky, U. (2003). Statistical mechanics for secondary school: The GasLab modeling toolkit. *International Journal of Computers for Mathematical Learning, 8*(1), 1 – 41.

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep or a firefly: Learning biology through constructing and testing computational theories—An embodied modeling approach. *Cognition & Instruction, 24*(2), 171 – 209.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems perspective to making sense of the world. *Journal of Science Education and Technology, 8*(1), 3 – 19.

Wilkerson-Jerde, M., Gravel, B., & Macrander, C. (2013). SiMSAM: An integrated toolkit to bridge student, scientific, and mathematical ideas using computational media. *Proceedings of the International Conference of Computer Supported Collaborative Learning (CSCL 2013)* (Vol. 2, pp. 379 – 381), International Society of the Learning Sciences.

Wilkerson-Jerde, M.,&Wilensky, U. (2010). Restructuring change, interpreting changes: TheDeltaTick modeling and analysis toolkit. In J. Clayson & I. Kalaˇs (Eds.), *Proceedings of the Constructionism 2010 Conference*, Paris, France, p. 97.

Williams, M., & Linn, M. C. (2002). WISE inquiry in fifth grade biology. *Research in Science Education, 32*(4), 415-436.

Windschitl, M. (2000). Supporting the development of science inquiry skills with special classes of software. *Educational technology research and development, 48*(2), 81-95.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. doi:10.1145/1118178.1118215

Xiang, L. & Passmore, C. (2010). The use of an agent-based programmable modeling tool in 8th grade students' model-based inquiry. *Journal of the Research Center for Educational Technology, 6*(2), 130-147.

Zacharia, Z. C. (2007). Comparing and combining real and virtual experimentation: an effort to enhance students' conceptual understanding of electric circuits. *Journal of Computer Assisted Learning, 23*(2), 120-132.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

Endnotes

[i] The name DeltaTick was chosen to reflect the environment's emphasis on describing the behaviors each agent performs within one tick or iteration of simulated time, and the resulting quantitative change or delta that results from the aggregation of those behaviors.

[ii] Bamberger contrasts the "simplest elements" for a novice in a domain with the "smallest elements" that make up the normative description of that domain. For example, structures such as melody, rhythm, and repetition might more "simply" describe a musical composition for a novice, rather than the specific pitch and duration of individual notes used in formal musical notation.

[iii] DeltaTick, like NetLogo, is open source. The open source project is available at http://bit.ly/DT_github, and packaged software and additional resources can be accessed at http://ccl.northwestern.edu/deltatick/.

[iv] DeltaTick has been through several rounds of development and refinement. Some of the screenshots featured in this paper are of older versions of DeltaTick, and may not look exactly as we describe here.

[v] The hares are aware of how important it is to stay one step ahead of tortoises (Aesop, as cited in Pinkney, 2000).

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

	Grade Levels	School (% Low SES)	Classes	Topics Explored	Duration	# Student Participants
Population Dynamics	10th, 11th, 12th	Public Urban Senior High (37.3%)	2 Science	Exponential & Logistic Growth, Relative Stability, Constraints	Two 100 minute sessions	44
		Public Urban Senior High (98.4%)	4 Math		Four 50 minute sessions	42
Natural Selection	7th	Public Suburban Junior High (47.1%)	4 Science	Genetic Variation, Predation, Natural Selection	Three 40 minute sessions	46

Table 1. Overview of *DeltaTick* implementation across studies.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

	EPISODES	**TARGET MODEL**	**CAUSAL REASONING**	**BUILDING UP**	**BREAKING DOWN**
POPULATION DYNAMICS	17	13	12	12	7
NATURAL SELECTION	10	9	4	9	8
TOTAL	27	22 (81%)	16 (60%)	21 (78%)	15 (56%)

Table 2. Results of coding analysis for students' degree of curricular engagement using DeltaTick.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

	POPULATION DYNAMICS	NATURAL SELECTION
RESPONSIVITY: ADAPTATION	✓ Facilitator added new "reproduce" and "die" blocks between studies to make models easier to construct without specific parameters.	✓ Facilitator removed blocks to help focus students' explorations. (Case P1) ✓ Facilitator renamed "step-size" block to "speed".
REPONSIVITY: EXTENSION	✓ Facilitator elicited and received several student recommendations for extensions, implemented some on last day of activity. (Case P2) ✓ One group spontaneously proposed introducing additional spatial constraint types. • 2 groups (1 at each site) spontaneously proposed adding sex/gender dynamics. • 4 groups (1 at first site, 3 at second site) spontaneously proposed adding infection behavior.	• 1 group proposed a measurement of death. • 1 group proposed a new "chase" behavior block.
RESPONSIVITY: LOWER-LEVEL	✓ 1 student group worked with a facilitator to edit NetLogo code and introduce infection behavior. ✓ 1 group worked with a facilitator to edit NetLogo code to modify the initial conditions of the model. • 1 group independently edited NetLogo code to introduce infection behavior. • 2 groups questioned the underlying behavior of construction blocks ("wander" and "only-if-this-much-space") in ways that could be explored by accessing NetLogo code.	n/a

Table 3. Instances during classroom enactments when facilitators used features of DeltaTick to respond to pedagogical needs (indicated with a checkmark), and instances we identified as potential further evidence of the potential for responsivity (indicated with a bullet).

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

CASE	DESCRIPTION	SUMMARY OF LESSONS LEARNED	DESIGN PRINCIPLES
C1	Exploring Variation and Natural Selection (Andy & Nikhil)	"Building Up" – Students use *DeltaTick* to incrementally build and explore curricular phenomena.	Curricular Example Space
C2	Exploring Interacting Causes for Population Growth (Mayra & Jessica)	"Breaking Down" – Students use *DeltaTick* to identify and explore factors contributing to a complex simulated curricular phenomenon.	Curricular Example Space
P1	Changing Blocks	Facilitator refines the collection of available blocks to streamline student investigations.	Curricular Example Space
		Facilitator modifies a trait label to more closely reflect to student interpretations.	Responsivity: Adaptation
P2	Library Extensions	Facilitator adds new blocks to existing library to expand space of investigation	Curricular Example Space
		Blocks added in direct response to student proposals for topics.	Responsivity: Extension
P3	Editing Text Code (Vince & Joey)	Text code refined to accommodate specific student questions during an investigation.	Responsivity: Lower-Level

Table 4. Summary of vignettes, lessons learned, and mapping to design principle.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

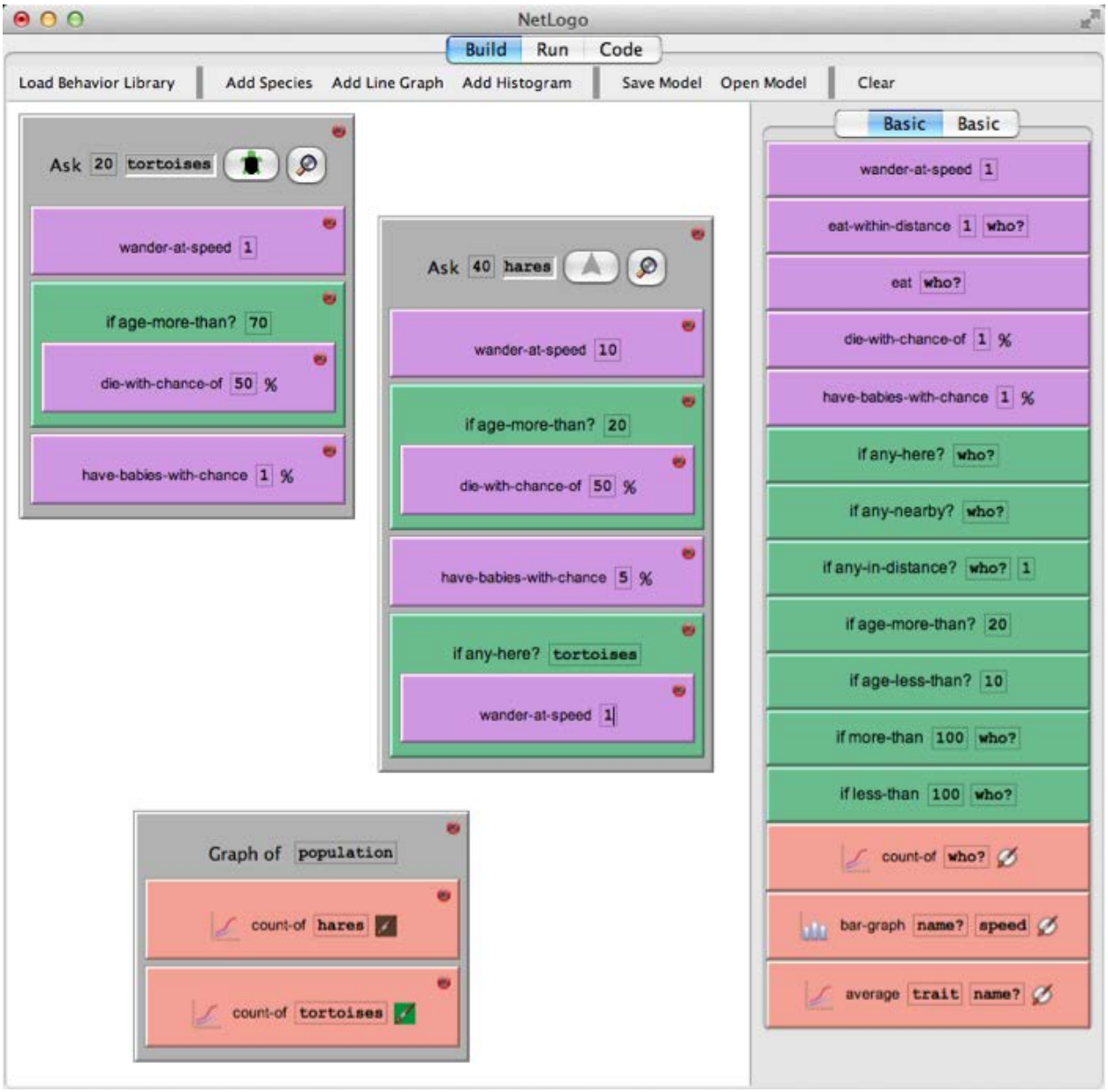

Figure 1. The *DeltaTick* interface. Simulations are constructed by assembling different combinations of blocks within the appropriate species window.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

Figure 2. The trait window (top) allows users to define within-species variation for each species trait. Once a within-species variation for a trait is introduced, a yellow trait block is added to the library (bottom) and can be used with behaviors.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

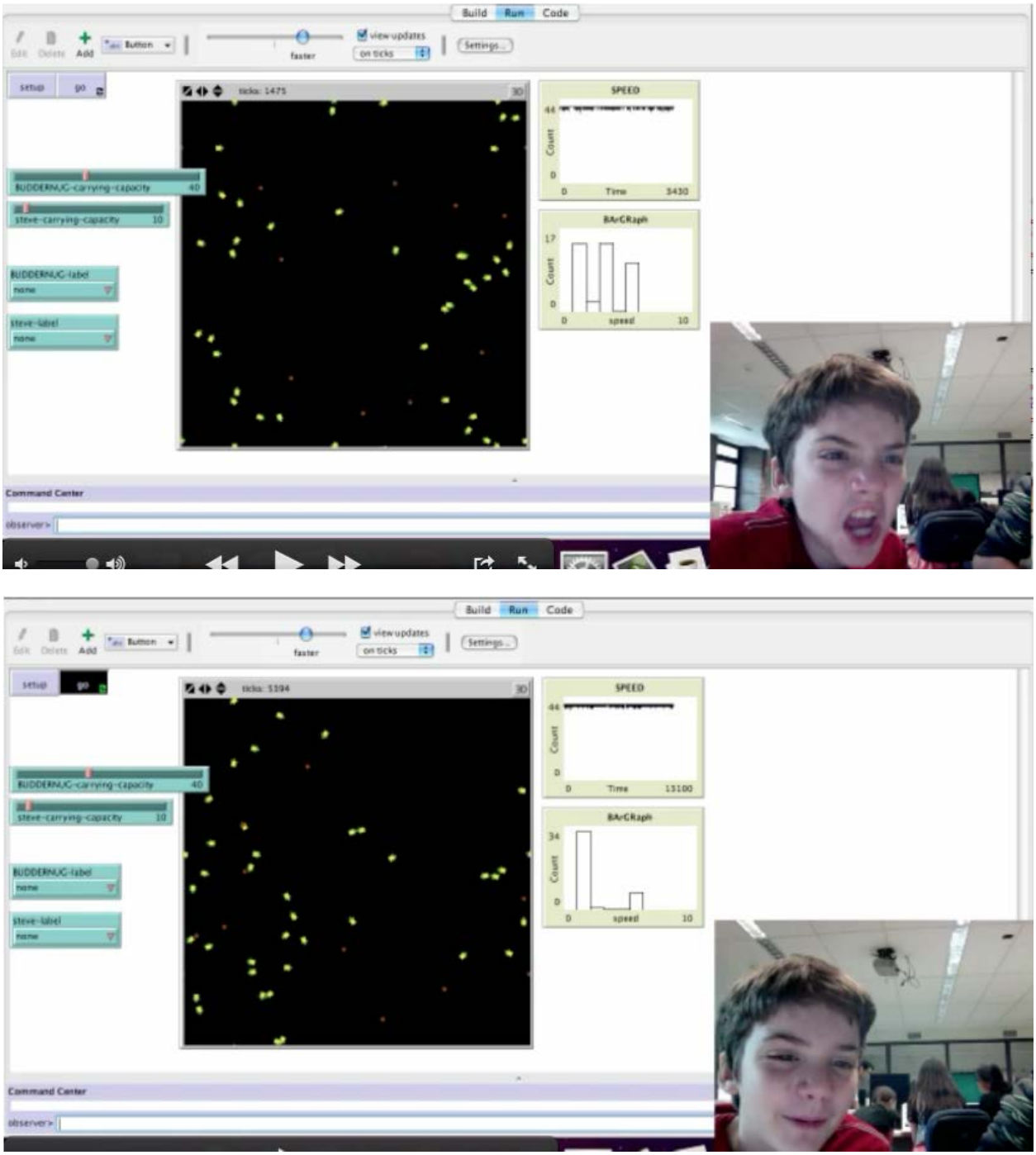

Figure 3. Andy and Nikhil monitor the distribution of speed as it changes from including a variety of speeds (histogram in top screenshot) to many slow and only a few fast "buddernaug" (histogram in bottom screenshot) in their simulation.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

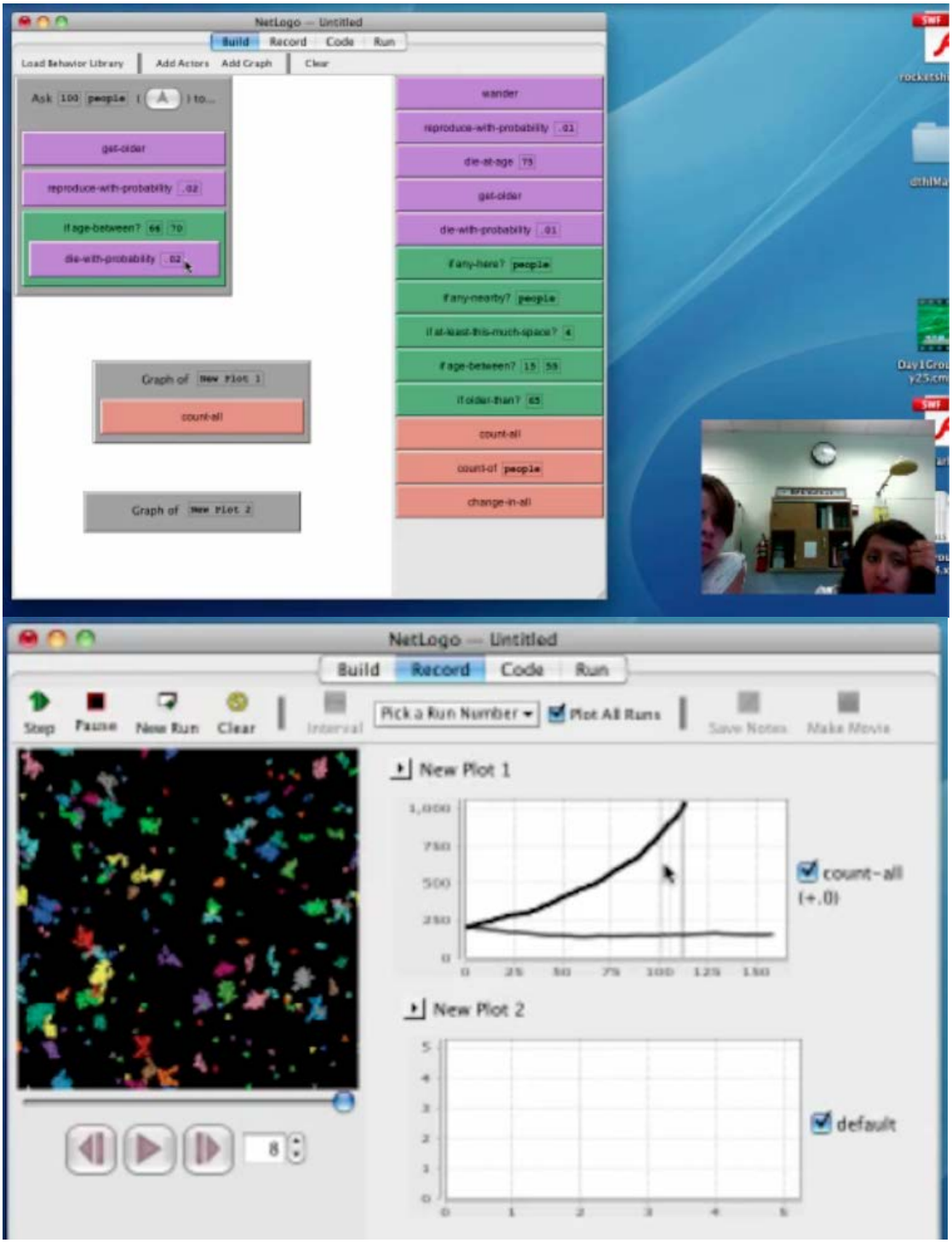

Figure 4. (top) Mayra and Jessica revise their original simulation. (bottom) The new simulation produces a more dramatic pattern of growth (in plot, black represents current simulation population levels, and grey represents prior simulation run).

```
<trait name="step-size">
    <setupReporter>random 9</setupReporter>
    <variation name="very-fast" value="5" setupNumber="25"></variation>
    <variation name="fast" value="4" setupNumber="15"></variation>
    <variation name="medium" value="3" setupNumber="30"></variation>
    <variation name="slow" value="2" setupNumber="20"></variation>
    <variation name="very-slow" value="1" setupNumber="10"></variation>
    <message>What is their speed?</message>
    <mutateCode>
    <probability-slider name="prob-slider" default="50"></probability-slider>
    if random-float 100 &lt; prob-slider [set trait]
    </mutateCode>
</trait>
```

```
<trait name="speed">
    <setupReporter>random 9</setupReporter>
    <variation name="very-fast" value="5" setupNumber="25"></variation>
    <variation name="fast" value="4" setupNumber="15"></variation>
```

Figure 5. The "step-size" block was modified to display "speed" in just a few minutes by replacing the trait name in XML code (original top, changed version bottom).

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

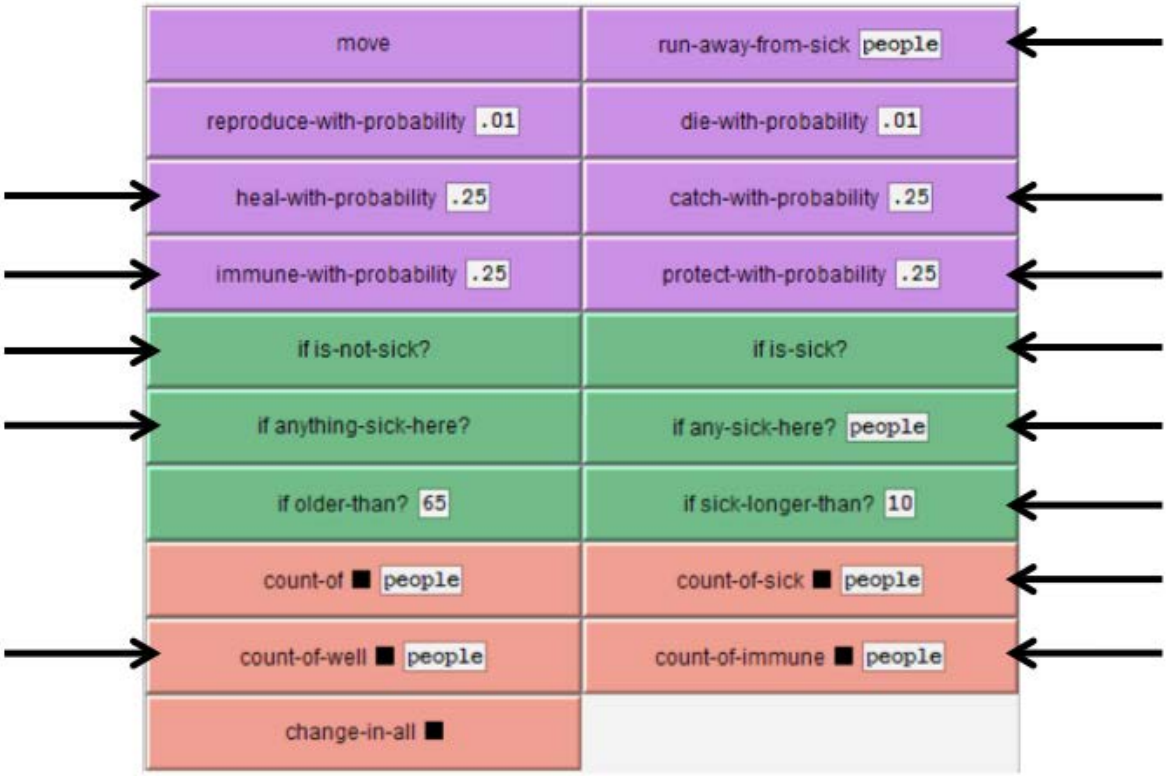

Figure 6. A modified population dynamic library created to enable students to model the effects of disease spread. Arrows point to blocks that were added to the existing library.

This is the accepted manuscript version of the article: Wilkerson-Jerde, M. H., 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. doi: 10.1002/sce.21157

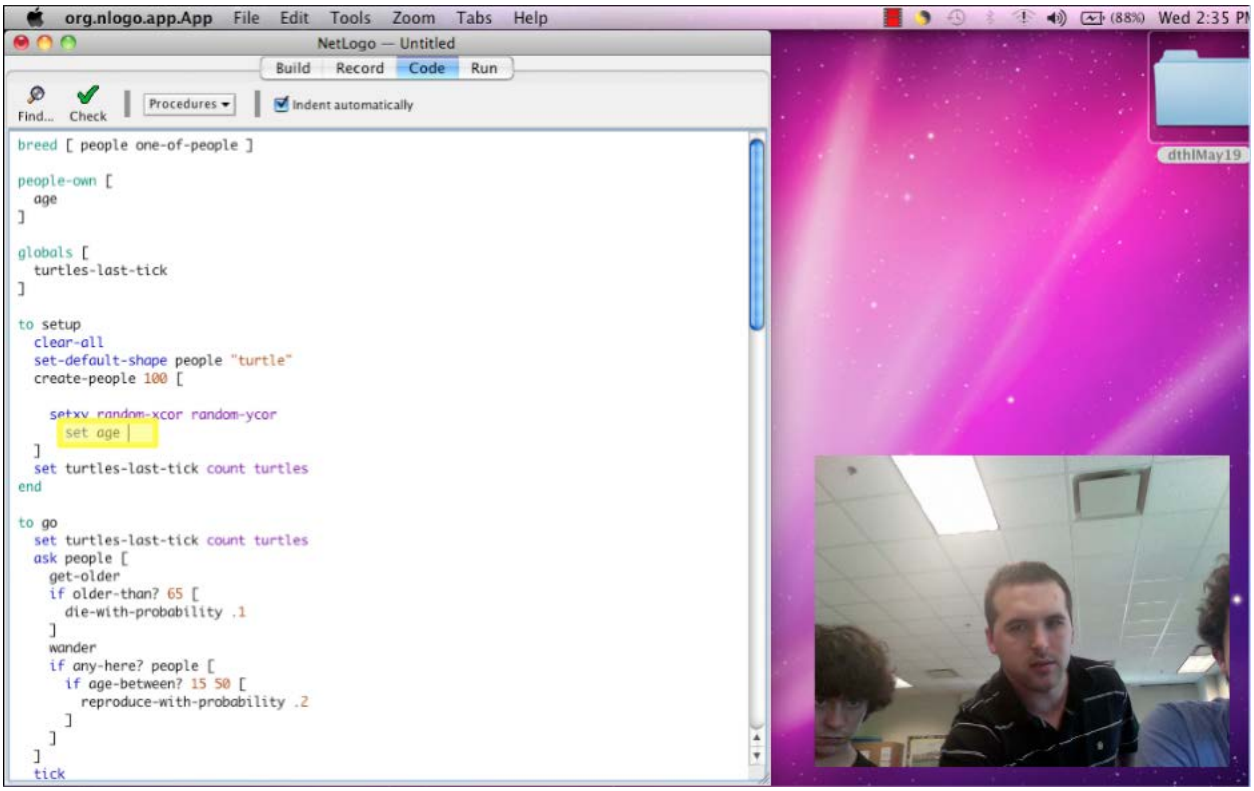

Figure 7. A facilitator helps a student group directly modify NetLogo code (changing the initial age of people from "random 50" to "0", highlighted here) to test a theory about the influence of age on population patterns they are observing in their model.

Appendix A.

Here we describe our data selection and coding procedures, and present examples of border cases we encountered while analyzing students' use of DeltaTick. Our goal is to help make explicit our reasoning processes while coding so that the implications of the numbers we report are clear, in the spirit of Hammer & Berland (2013).

Data Selection Procedures. In order for our analyses to be comparable across different enactments (which lasted different amounts of time and involved different instructional sequences), we developed the following selection criteria for data to explore curricular goals. We identified intervals of time from each of the three classroom enactments that (a) constituted one complete, designed lesson from each of the curricula, (b) engaged students in small group model construction and exploration with DeltaTick, (c) asked students to construct one or more simulations that would exhibit a particular target outcome aligned with curricular goals, and (d) did not specify which blocks or parameters students should use to build the models (e.g. was not a tutorial). We eliminated any videos that captured less than ½ the total duration of the planned lesson (usually as a result of technical difficulties; this was 30 minutes of video in both Population Dynamics enactments, and 23 minutes in the Natural Selection enactment). All remaining videos of student group work using the DeltaTick tools during these times were included in our analysis. These videos represented the second half of Day 1 for Population Dynamics Study 1, Day 3 for Population Dynamics Study 2, and Day 2 for Natural Selection Study 1. In all three cases, students had not been working with the DeltaTick environment for more than one hour total at the beginning of the episodes analyzed.

Definitions of "Target Model" for each Curricular Focus. For each study, we coded the number of participant groups who constructed the "target model" intended in each curriculum. In the Natural Selection study, a model was coded as a target model when multigenerational population change resulting from natural selection was evident in the model. This required having certain behaviors in the model (like

reproduce, die or eat) as well as including a graph to represent change in the population over time. If a model did not include a graph, but the video of the student group showed evidence that they had noticed patterns of population change resulting from the behaviors they had coded in, we coded this successfully producing the target model.

In the Population Dynamics study, a model was coded as a target model if it exhibited one of a number of behaviors included in the task prompt. These prompts required participants to construct models using birth, death, and constraint elements (more than one behavior had to be included in the simulation) that generated exponential-like growth or exponential-like decay at a 5% rate, relative stability in the population for at least 100 ticks, logistic-like growth, or rapid decay followed by relative stability. All models constructed by students in the population dynamics activities included graphs.

Examples of "Causal Reasoning", "Building Up", and "Breaking Down" During Model Construction. Finally, for each study, we coded the number of participant groups who engaged in reasoning about the underlying causal relationships exhibited by (or missing from) their models as they constructed them. We did this to address concerns about the degree to which students constructed models by "jigsaw puzzling" (Löhner et al., 2003), or adding and removing model components only to reproduce desired effects, rather than substantively considering what those components mean in terms of the actual system being modeled.

Table A1. Examples and non-examples of "Causal Reasoning".

	Population Dynamics Study	**Natural Selection Study**
Target Model & Causal Reasoning	Group adds and/or adjusts behaviors and parameters to reflect actions and values that would be reasonable for human populations. Group discusses changes to model behaviors or parameters in terms of their influence on individual agent behavior within the simulation.	Group notices and describes how an individual's trait is influencing its chances of survival and reproduction, and consequently a change in its population.

Target Model, No Causal Reasoning	Group achieves target by comparing and adjusting the difference between numerical parameters only, without connecting parameters to the behaviors they influence.	Group builds a model that depicts natural selection patterns, but does not notice and describe the pattern.
No Target Model or Causal Reasoning	Group achieves target, but continues to adjust parameters because each data point is not exactly the same as in the target example. This indicates a failure to consider the probabilistic element of the simulation (and reproduction more generally). Group does not achieve target model.	Group does not notice or describe how the speed trait distribution or average speed of a species changed/stayed constant over time.

We identified students' model construction approach as "building up" when they started with a simple or easily understood model, and incrementally added or refined their model in an effort to complicate it, adjust it toward the intended target, or understand how the new behavior or parameter would influence the system. In general, when participants are "building up", they have a clear expectation for how their modification will affect the model. Within each specific curricular focus, examples of "building up" are:

Table A2. Examples, borderline cases, and non-examples of "building up".

	Population Dynamics Study	**Natural Selection Study**
Clear example	Group increases the birth rate in their model because population growth is lower than intended target.	Group decides to add a reproduce behavior because a species died out.
Border example	Group adds "get older" block because they are using age as a constraint, but the block's influence on model behavior is not evaluated or discussed.	Group decides to add a graph block to their model after noticing instructions on the worksheet about how to add a graph block.
Non-example	Group adds a "wander" block without explicit motivation, movement and spatial constraints are not a component of the model.	Group adds a "move" behavior without describing or discussing why they are adding the block.

We identified students' model construction approach as "breaking down" when they started with a complex or not well understood model, and removed or simplified elements of the model in an effort to isolate and understand the model's inner workings, or identify the most important relationships of interest for solving the curricular task. In general, when participants are "breaking down", they do not have a clear expectation for how their changes will influence the model, only that it is likely to change it in some way. Within each specific curricular focus, examples of "breaking down" are:

Table A3. Examples, borderline cases, and non-examples of "breaking down".

	Population Dynamics Study	**Natural Selection Study**
Clear example	Group notices that their model includes two possible causes for an unexpectedly high death rate. They remove one possible cause to focus on adjusting the second appropriately.	Group increases the initial number of a species in the model to see whether this will help the population survive.
Border example	Group lowers the age in an age constraint to "0", rendering the constraint inconsequential, without explicit justification.	Group changes the chance of reproduction for individuals in a species without explicitly describing why they were trying that out.
Non-example	Group removes blocks from a prior model in order to begin a new task, but keeps 1 or 2 blocks for the new task.	Group removes a block by mistake.

Appendix B.

Here we explore in further detail the notion of a *Curricular Example Space* using the Population Dynamics Library as an example. We illustrate how an intended collection of particular population dynamic patterns drove our selection of blocks in the design of the original library. Also, we illustrate how different modifications to the library or to the grain size or specificity of particular blocks could extend or limit that example space.

Target Dynamics. In designing the Population Dynamics Library, our original intention was to allow learners to explore interdependent relationships between probabilistic birth, death, spatial and time-dependent constraints, and competition in ways that connect to mathematical representations of population dynamics. Therefore, our curricular example space can be best described as a collection of mathematical models of population dynamics, along with the different combinations of agent behaviors that would generate each. For example, exponential growth can be generated with only reproduction. However, it can also be generated with a combination of reproduction, death, and age constraints on reproduction – as long as the probability to reproduce for eligible agents is larger than the probability for all agents to die, divided by the proportion of agents available to reproduce (to ensure that deaths among the entire population are still exceeded by births among a restricted set). Our set of intended mathematical models included basic models such as exponential growth and decay, common models such as logistic growth and stability, and simple models of competition.

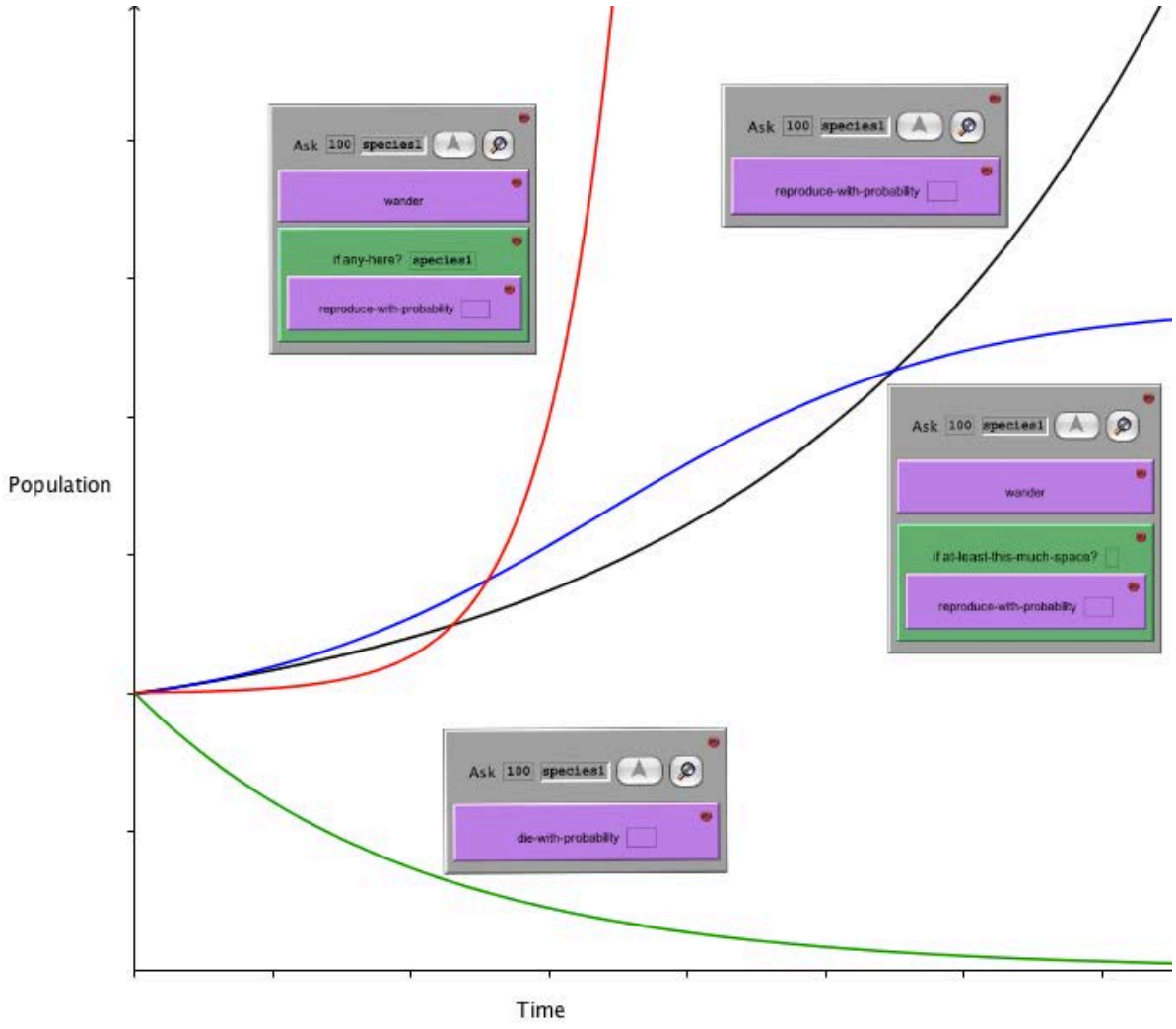

Figure B1. Simplest combinations of behavior blocks to create exponential growth (black), exponential decay (green), lagged exponential growth (red), and logistic-like growth (blue).

Component Elements. To determine what elements were needed to reconstruct this curricular example space in a way that was accessible to learners, we identified the smallest unit of behavior that could be directly linked to a given mathematical effect. For example, birth [reproduce-with-probability] and death [die-with-probability] are the smallest units of behavior that *increase* or *decrease* the population. Age [if-older-than?] and [if-between-ages?] and space constraints can *limit* or population growth or decay by restricting them to particular subpopulations or global conditions, or *supplement* growth or decay by offering further increasing or decreasing population under particular circumstances. Age and space

constraints are different from one another, however, in that space constraints produce an immediate effect while age constraints introduce time delays (for example, if the distribution of ages in a population is such that increases in new births does not effect increases in eligible agents within certain ages until later in the model's execution). At this level, only one block is needed to construct a simple model, and each block that is added or removed is likely to directly connect to a notable change in the dynamics of population growth over time.

Adapting the Example Space through Modification of Primitives. Since we designed each block in the Population Dynamics Library to correspond to some substantive change in quantitative dynamics, it is clear how removing certain blocks could further restrict the available set of models students had available to explore. For example, without spatial constraints, users would be unable to construct any models that exhibit logistic-like patterns of growth. Limiting or removing parameters on the available blocks would allow learners (like those in the second Population Dynamics study) to focus on how particular factors influence the *shape* of population dynamics patterns: that is, what factors lead to exponential growth or decay, logistic-like growth, stability or other patterns, and how those factors interact. Adding specificity to blocks, such as allowing users to control the speed at which agents [wander], would conversely highlight agent level and inter-agent dynamics that eventually lead to overall patterns. Adding blocks, as illustrated in Figure 6, expand the solution space to allow learners to explore patterns that occur within and across populations, such as infection spread or predator-prey interactions, rather than maintaining focus on those that manifest at the single population level.