"That's What Science Is, All This Data:" Coding Data Visualizations in Middle School Science Classrooms

Ari Krakowski
akrakowski@berkeley.edu
Lawrence Hall of Science,
University of California, Berkeley
Berkeley, California, USA

Eric Greenwald
eric.greenwald@berkeley.edu
Lawrence Hall of Science,
University of California, Berkeley
Berkeley, California, USA

Natalie Roman
nroman@berkeley.edu
Lawrence Hall of Science,
University of California, Berkeley
Berkeley, California, USA

ABSTRACT

In this experience report, we describe the Investigating Air Quality curriculum unit that integrates computational data practices with science learning in middle school science classrooms. The unit is part of the Coding Science Internship instructional model, designed to broaden access to computer science (CS) learning through scalable integration in core science courses, and through confronting barriers to equitable participation in STEM. In this report, we describe the core features of the unit and share preliminary findings and insights from student experiences in 13 science classrooms. We discuss affordances and challenges for student learning of computational data practices in formal science classrooms, and conclude with emerging recommendations for instructional designers.

CCS CONCEPTS

• **Social and professional topics** → **Computing education**; *Computational thinking*; *K-12 education*; *Model curricula*; • **Applied computing** → **Interactive learning environments**; • **Human-centered computing** → *Visualization systems and tools.*

KEYWORDS

data science education, middle school, computational thinking

ACM Reference Format:
Ari Krakowski, Eric Greenwald, and Natalie Roman. 2022. "That's What Science Is, All This Data:" Coding Data Visualizations in Middle School Science Classrooms. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2022), March 3–5, 2022, Providence, RI, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3478431.3499387

1 INTRODUCTION

Computation has become critical to an ever-broadening list of disciplines, particularly within STEM fields [6, 15, 27]. However, its growing role risks further deepening persistent inequities in STEM pathway participation[13, 43]. While there is growing recognition of its value, CS education continues to have a limited role in K-12 classrooms [20, 56]. In response, there is a push for transdisciplinary approaches that integrate CS concepts and practices in required subjects like science [4, 18, 19, 39]. This strategy serves not only to broaden access, it also affords authentic integration of CS with science [26, 33, 50]. Activating a greater diversity of youth toward engagement with data practices is particularly urgent to achieve more equitable participation in an increasingly data-driven world, and to position youth as agentic and critically conscious data practitioners equipped with data literacies that encourage interrogation of data production and usage [24, 53].

Approaches aimed at integrating CS in core science courses are well-served when they align with long-standing science and engineering practices (e.g., scientific modeling and data analysis) [44, 45, 50]. Data analysis is a core epistemic practice emphasized in both science and CS standards [11, 12, 48] and thereby represents a promising avenue for CS integration in science classrooms. Computationally rich data practices [50] are increasingly important for science learners: as new technologies both produce and become more capable of processing vast quantities of data, students must learn to comprehend, manipulate, and design computational tools to support such use [15, 36, 50]. Engaging students in computational data practices can help them understand how datasets are generated, and how to structure and analyze datasets to reveal patterns and identify relationships among variables [14, 34, 35, 55].

2 PROJECT BACKGROUND

2.1 Theoretical foundations

This paper focuses on student experiences with an instructional unit that integrates computational data practices into core middle school science classrooms, designed for students and teachers who have little or no prior programming experience. The instructional model is designed to immerse students (Grades 6-8) in a discourse-rich simulated internship (e.g., [46]) that mirrors the collaborative and computational work of practicing scientists. This model aims to offer a more inclusive representation of CS work [9, 51, 57]; and position coding in service of addressing real-world problems to counter negative perceptions of CS endeavors as limited in value [1, 9, 21, 37]. Through the simulated internship format, our pedagogical framework grounds Mitch Resnick's *coding to learn* approach to CS education [41] in situative learning theories and, in particular, the construct of legitimate peripheral participation (LPP, [32]). Coding to learn, in which learning is contextually embedded in authentic tasks, is well-aligned with situative theories, which posit that knowledge is constructed through activity and in relation to others—thus, learning is "situated" in the activity, context, and community in which it occurs [16]. The simulated internship model is therefore designed to encourage student progression along a trajectory from peripheral to more central participation in the practice of computational data processing.

SIGCSE 2022, March 3–5, 2022, Providence, RI, USA.

ACM ISBN 978-1-4503-9070-5/22/03.
https://doi.org/10.1145/3478431.3499387

2.2 Interdisciplinary Partnerships

The Investigating Air Quality unit is one of two 10-lesson units developed through the Coding Science Internships project. This project leverages a long-standing partnership between the Lawrence Hall of Science and Amplify Education, which has produced Amplify Science, a comprehensive K-8 science program currently in use across the US. The two units are designed to ultimately be incorporated into the Amplify Science curriculum, positioning them for broad uptake in science classrooms. In addition, the curriculum design team worked closely with Jacob Duke, a district lead in a western US state with extensive experience implementing CS learning experiences, who provided a practitioner lens during development and supported participant recruitment.

2.3 Instructional context and study sample

Drawing on the principles of design-based research [3, 10], the unit was iteratively piloted for two years with over 175 students in 7 class sections in two western states. We then revised the unit for broad implementation and conducted research trials with 13 middle school science classrooms in two states during the 2020-21 school year. Critically, the unit was designed before the COVID-19 pandemic and was thus created for in-person instruction. Given the pivot to remote learning in participating districts, we adapted the instructional materials and professional learning resources to enable remote implementation. We also provided ongoing support (through email, text) and weekly "office hours" for teachers during the research trials to respond to emerging questions or any technical issues that arose during research trial implementation.

Analysis of data about student learning gains (reported in Section 4.1), reflects the sample of n=478 students who participated in the revised (2020-21) version of the unit and who completed both pre- and post- versions of the *Assessment of Computational Thinking* (for background on items, see [47]; for technical report on evidence for measurement validity, see [54]). Insights into student experiences (Sections 4.2-4.4) are drawn from observations during the piloting phase and from semi-structured teacher interviews conducted with each of the 13 teachers shortly after they implemented the revised unit.

3 CURRICULUM DESCRIPTION

3.1 Overview and Instructional Goals

In the Investigating Air Quality unit, students inhabit the role of data science interns to work with a large US air quality dataset, and use a custom visual programming environment (VPE, Data Studio) to query, filter, explore, and create visualizations of the data that communicate information about air quality. Students work together to create and critique their code by testing code against expected outcomes. The national scope of the dataset enables place-based customization of instruction as students investigate air quality in their own community and compare it to other regions. Throughout the unit, students deepen their understanding of the science concepts related to air quality as well as basic coding concepts needed to create data visualizations from large datasets. Students also gain experience in data analysis and interpretation as they scrutinize the visualizations they create to make recommendations for improving air quality in specific locations. In order to promote the integration of CS into science, we developed a build of learning goals (Figure 1) that frames learning within the practice of contemporary computational science. Learning goals were developed through analysis and integrative interpretation of CS and science standards [11, 48], drawing on insights from literature about engaging students in computational data practices [14, 34, 35, 55]. The learning goals were iteratively refined during unit development in collaboration with leads and educators in partner districts. Additionally, students practice collaboration skills throughout the unit as they engage in pair programming to complete coding tasks, analyze and evaluate different coded solutions, and critique and improve their code.

1. **Scientists generate and organize data in ways that allow them to use computers to analyze and interpret the data.**
   - Scientists need to understand the real-world phenomena being investigated to determine which data should be collected, as well as when, where, and how data should be collected.
   - Scientists generate large amounts of data about phenomena by measuring things in different places over time.
   - Scientists organize their data into datasets using the variables they care about, which allows the data to be analyzed and interpreted.
2. **Scientists can use coding to analyze large datasets to reveal patterns and relationships between variables. They can reuse and/or slightly modify code to create and critique different visualizations to answer different questions or answer questions in different ways.**
3. **Scientists often need to employ additional coding moves in order to reveal patterns. For example:**
   - To specify which data to include in analysis and visualization
   - To organize data
   - To perform a mathematical operation

Figure 1: Build of learning goals in the Investigating Air Quality Coding Science Internship curriculum unit

3.2 Core curricular features

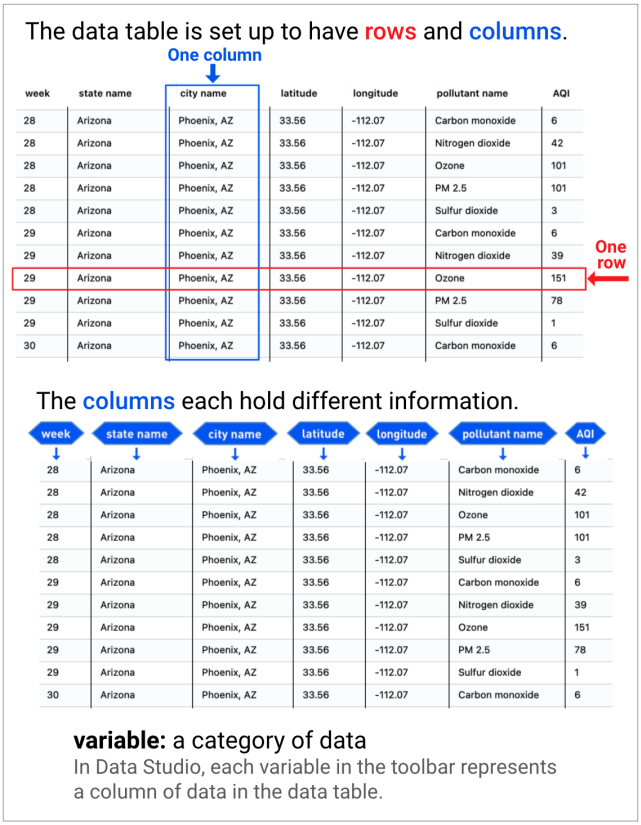
3.2.1 Real-world problems to motivate learning. The curriculum builds on our previous and ongoing work [17, 31], in which CS and science concepts are situated within a problem context that connects those concepts to compelling, real-world problems. Situating learning in a problem context enables students to move beyond receptive knowledge 'for school' (I know something important in science class) to productive knowledge (I can explain something of significance in the broader world). This, in turn, fosters a more expansive and inclusive epistemic and discursive repertoire that can promote broader, more equitable participation in STEM [2, 5, 7, 8]. The dataset is derived from one year of air quality index (AQI) monitoring data from the US Environmental Protection Agency (EPA). The massive size of the air quality dataset authentically motivates the need for computational data practices to enable data analysis and interpretation. Students analyze the AQI dataset and code data visualizations using the Data Studio VPE: they identify locations with air quality concerns, determine which pollutant(s) are problematic, and evaluate whether the problem is occasional or perennial. Students read about actions that places around the world

have taken to reduce levels of specific air pollutants, and then make recommendations for how to address air quality concerns in the US locations identified through their computational data analysis.

3.2.2 "Thinking in rows" to build understanding of dataset structure. To support student understanding of how data may be structured to enable computational analysis, we developed instructional strategies that invite students to "think in rows." While data tables are familiar representations in middle school science classrooms, their structure is typically "wide" (fewer rows and many columns) rather than "tall" (many rows and fewer columns), because the role of the data table is usually to communicate or organize information from smaller datasets, rather than to support computational analysis or visualization of large datasets. Since a "tall" dataset structure is more amenable (e.g. [22, 42, 52]) to "data moves" [14] for analyzing, visualizing, and interpreting data, we structured the AQI dataset in this way, such that each row represents a single AQI case, or observation for a given week, specific location, and pollutant; and each column represents a data variable (e.g., AQI value, week number, location city name). Figure 2 shows screenshots of instructional resources used in conjunction with whole-class and pair discussions to help students "think in rows," to understand: (1) how the dataset is structured into rows (cases) and columns (variables); and (2) how the variable code blocks (the blue hexagons) are derived from data table columns and instruct the computer to conduct specific data moves that permit analysis and visualization. This deeper understanding of dataset structure is intended to support students in visualizing what control structures in their code are instructing the computer to do. For example, students build understanding of how Boolean operators in code can instruct a computer to filter the dataset through use of three code blocks in the VPE: *INCLUDE DATA IF* (a natural language analogue of the traditional filter command), *AND*, and *OR*. Students' conceptual understanding of Boolean logic builds off their work with "thinking in rows:" as they make sense of the structure of the air quality dataset, they begin to understand and develop facility with Boolean operators that enable them to specify which rows to include in order to code a data visualization that answers a given question.

When a student is figuring out how to code a graph that will show data for a specific city and a specific pollutant, they can visualize the rows of the data table and use the natural language in the Boolean operator blocks for support. In this way, students can imagine the computer "grabbing" only rows that include both the city name and the pollutant name they are interested in. Using the language of the block, a student might reason *I want to tell the computer to include data if the row contains [city where student lives] AND PM 2.5.* "Thinking in rows" also helps students to synthesize the variables in the dataset into cases, making what is very abstract to students more concrete, connected, and understandable. In one activity, students talk with a partner and describe a single row in a sentence, including each variable in their description. For example, a student would describe the row in Figure 2 as *in week 29, in Phoenix, Arizona* (where the latitude is 33.56 and the longitude is -112.07) *the AQI for ozone was 151.*

3.2.3 Multimodal and "unplugged" engagement to support conceptual depth. To support conceptual understanding of foundational data practices, the curriculum included multimodal [25] activities,



week	state name	city name	latitude	longitude	pollutant name	AQI
28	Arizona	Phoenix, AZ	33.56	-112.07	Carbon monoxide	6
28	Arizona	Phoenix, AZ	33.56	-112.07	Nitrogen dioxide	42
28	Arizona	Phoenix, AZ	33.56	-112.07	Ozone	101
28	Arizona	Phoenix, AZ	33.56	-112.07	PM 2.5	101
28	Arizona	Phoenix, AZ	33.56	-112.07	Sulfur dioxide	3
29	Arizona	Phoenix, AZ	33.56	-112.07	Carbon monoxide	6
29	Arizona	Phoenix, AZ	33.56	-112.07	Nitrogen dioxide	39
29	Arizona	Phoenix, AZ	33.56	-112.07	Ozone	151
29	Arizona	Phoenix, AZ	33.56	-112.07	PM 2.5	78
29	Arizona	Phoenix, AZ	33.56	-112.07	Sulfur dioxide	1
30	Arizona	Phoenix, AZ	33.56	-112.07	Carbon monoxide	6

Figure 2: Example instructional resources to help students "think in rows."

including kinesthetic, unplugged activities [23], open-ended coding tasks, comparative worked examples, and discourse routines that support sensemaking and debugging. Student engagement with this multimodal interactive pedagogical approach is described below to illustrate students' conceptual engagement with Boolean logic.

Kinesthetic activity. Leveraging students' familiarity with playing cards, this activity creates a class playing card "dataset," where each student contributes information about three card variables: suit, color, and number. Students begin by moving to the outer edges of the classroom with their playing card and the accompanying "row" of playing card data that specifies the playing card attributes for each variable. The instructional resources include large whiteboard code blocks that employ Boolean operators to instruct students to move to the center of the classroom if their data is "true" for the given code sequence (e.g., *INCLUDE DATA IF* suit = hearts *AND* number <5). The class enacts multiple rounds of code sequences, including several sequences designed to address confusion we observed in pilots about interpreting *AND* and *OR*. For example, students discover that *AND* returns fewer results compared with *OR* by enacting scenarios where the results yield highly pronounced differences (e.g., *INCLUDE DATA IF* color = red *AND* color = black results in no students in the center of the room, whereas swapping *AND* for *OR* results in all students in the center of the room).

Coding tasks. Students apply their understanding of Boolean logic from the playing card activity to code a series of data visualizations that answer questions about air quality for specific locations and/or pollutants. Student work in Data Studio is supported by the Critiquing Code discourse routine and tasks that highlight commonly observed learner misconceptions with Boolean logic (see 3.2.4). A VPE-embedded visual "block glossary" also provides real-time support for understanding how Boolean operators interact with the data table and offers an example of a resultant data visualization from their use in code (see glossary entry for the *AND* block in Figure 3).

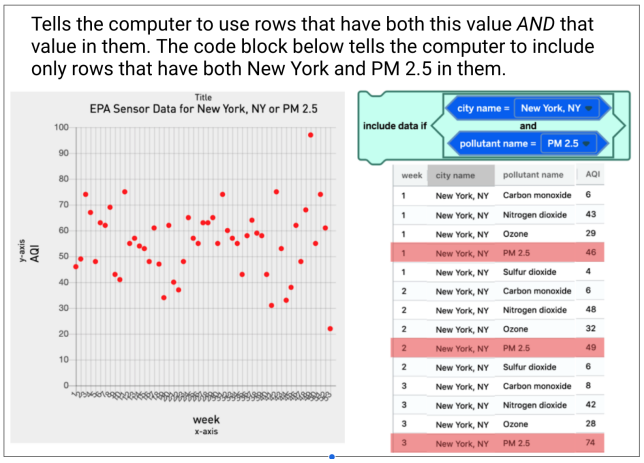


Figure 3: Visual block glossary example for the *AND* block

3.2.4 "Critiquing Code" to scaffold debugging practices and promote sensemaking. Resonating with insights from the field highlighting the importance of scaffolding debugging processes [28], pilot testing pointed to the importance of: (1) providing students with opportunities to critically examine code; and (2) explicitly supporting students' metacognitive capacity to interrogate the observed outcomes of algorithms against intended outcomes. Critiquing code supports students in the important practice of testing and refining computational artifacts [11] to debug code that doesn't work as expected, and to revise or adapt algorithms in response to shifting applications (e.g., to code a data visualization that answers a different question).

Instructional materials supported student engagement with critiquing code in two ways. First, a Critiquing Code discourse routine (Figure 4, top) accompanies pair programming activities and whole-class sensemaking discussions to provide explicit, structured support for evaluating coded solutions for data analysis in terms of both the code and the science goals of the analysis. The goal of the Critiquing Code routine is to help students see a computational artifact, like a graph produced in Data Studio, as executed code, and be able to ascribe particular features of the graph as resulting from specific elements in a given code sequence. Second, students interact with a series of Critiquing Code tasks in Data Studio (Figure 4, bottom) in which they run two preconstructed algorithms that address the same data analysis question, then use the discourse routine to compare the resulting graphs and discuss their affordances or limitations for effectively addressing the question. The Critiquing Code tasks were designed to target CS and science concepts identified in pilot testing as challenging for students, warranting additional sensemaking through focused worked examples that can support novice learners [49].

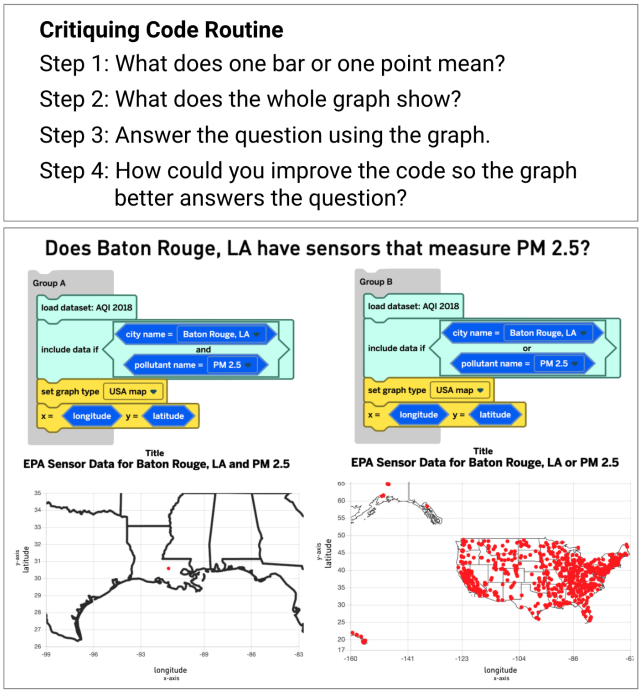


Figure 4: The Critiquing Code discourse routine (top) and an example Critiquing Code task in Data Studio (bottom)

4 INSIGHTS FROM STUDENT EXPERIENCES

4.1 Evidence of learning gains

Participating students demonstrated significant learning gains on an external measure of computational thinking [54] administered prior to and immediately after instruction. A paired samples t-test of performance on the CT measure for n=478 students (see Section 2.3 for further description) revealed that the mean score for students who participated in the 10-lesson/8-hour unit increased by .401*** on the 8pt measure of CT (effect size=0.214). While this is a small effect, we note the brief duration of the intervention and that it was observed in the midst of a pandemic and the unexpected pivot to remote implementation it required. We also examined science learning through a 2-item assessment task administered with the pre/post surveys. On this internally-developed assessment of the science content, we saw significant learning gains, with the mean score for participating students increasing by 0.199 on the 2pt scale (effect size=0.27). While further study is needed to evaluate validity evidence for the science content measure, these initial findings suggest that the unit is contributing to both CT and science learning. In the following sections, we present insights

from our mixed methods study (see Section 2.3) to shed light on the nature of these observed learning gains, and to explore ways in which instructional elements may be contributing to content learning and dispositional outcomes.

4.2 Conceptual engagement with Boolean logic

Given their critical role in computational data processing, learning experiences invested considerable time in helping students develop a conceptual understanding of Boolean operators and how to apply them to query and filter massive data sets. Given the need to pivot to remote learning with the emergence of the COVID-19 pandemic just prior to research trials, the instructional activities supporting student understanding of Boolean logic were significantly modified: the whole-class kinesthetic playing card activity was instead enacted through a video conferencing context that significantly curtailed student engagement with the Critiquing Code discourse routine and collaborative pair programming. Data from pilots and research trial implementation suggest that Boolean logic was unfamiliar and conceptually challenging for students, but was ultimately an area where students experienced growth in understanding and ability to make sense of code. Nearly all teachers identified the playing card activity as particularly valuable for building student dexterity using Boolean operators in algorithms. Teachers, like the one quoted below, often framed this progress as enabling students to think from the perspective of a computer: *"I feel like it gave them the vantage point of following the instructions, with them being sort of the machine [to] understand what was going on, on the other side of the screen… like inside the program. Once they got to sort of act as the computer... then they were able to understand. Right after [the playing card activity], there was way more success with it."*

Another theme we observed related to student learning of Boolean logic was that the instructional unit was seen by many teachers as promoting "productive struggle," where students persisted through challenge and felt proud of their work. As one teacher reported, *"That felt like a good struggle, where a lot of the students were working through it, but also arriving at answers that they understood."* Similarly, several teachers reported that students persisted through challenging content and leveraged instructional resources to advance their understanding, *"It felt like this was a time when students were really making connections to the critiquing code [activities], but also struggling with the concept of 'AND' and 'OR' ...That was a time when I felt like everyone was with me, but also struggling to get it."* As a component of productive struggle, teachers also identified the unplugged playing card activity with Boolean operators as promoting conceptual depth rather than simply procedural fluency. For example, one teacher reported that *"The playing card activity, it was nice to put everything else aside and be like 'Let's just talk about 'AND' and 'OR,' that concept, and let's play. It felt really accessible, and it felt like I was really teaching a new thing — to everyone — the kids who were already kind of like 'I know how to code already,' and the kids who were, you know, brand new … It's confusing, and it's tricky, and we worked on it together."*

As suggested in the quote above, we also saw some evidence that it was this attention to conceptual depth over procedural fluency that helped level the playing field in classrooms of students with widely divergent prior experiences with programming. As one teacher noted, *"Students who were completely new to coding made huge strides in understanding how a computer interprets code, as evidenced by their final explanations of their own work. Many students were a little nervous going into the unit but became more confident over time and began to think in a logical, stepwise manner."* Analyses to better understand the contribution of particular instructional features to these and related outcomes are ongoing.

4.3 Locally Relevant Data Science Context

Air quality was the driving phenomenon for the unit under investigation. Given that the students participating in the study were residents of states where forest fires are a perennial problem, air quality was a highly relevant real-world problem that students had recent firsthand experience with. Teacher interviews and student work samples suggest that this deepened student experiences with the material and helped to motivate learning. Teachers regularly reported that students brought in their own lived experiences to classroom activities, connected their learning about the problem with actions they could take to address it, and took what they learned back into their lives and community. For example, when asked what about the instructional model was most important for their students' learning, sentiments similar to those of the following teacher were common: *"I think that air quality is very immediate for especially kids in [this western US state] ... who have had school closed because [of] the wildfire smoke, maybe even had a family member that was evacuated, or they were evacuated at some point... I think that also what's nice about it is that there is actually a learning edge there also, because none of the kids could have labeled [smoke pollution] as PM 2.5... So it's like this thing where it feels important [and] it feels like they've had a real experience."* While this indicates the value of local relevance for our sample, a limitation is that we have not studied the unit in regions that are less directly affected by air quality concerns. We also see considerable opportunity to better support students in interrogating the dataset to examine how socioeconomic class and systemic racism contribute to varying air quality across locations.

4.4 Understanding the *Story of a Datum*

The instructional materials included custom media and visual resources that introduced students to the creation and processing of "big data," and specifically to the *story of a datum*: how air quality data are produced through measurement of different pollutants by sensors positioned at specific locations, recorded in tabular data structures, and ultimately represented in data visualizations. Interviews with participating educators, as well as researchers' observations during classroom pilots echo findings [29, 30] that engagement with data analysis in middle school science classrooms is typically limited to small datasets, as opposed to large datasets characteristic of modern science, such as the AQI dataset used in this unit. In interviews, nearly all teachers noted that engagement with large, externally-sourced datasets was a novel experience for their students. They also stressed the value of such engagement as central to the practice of contemporary science and authentically motivating the need for computational data practices. As one teacher reported, *"Seeing the long list of numbers [in the data tables] was really impressive for them, and for me ... just like oh, it keeps scrolling and it*

keeps growing! But we can organize it in a way ... this is what coders do, they take all these crazy numbers and they try to structure them. Being able to see that connection is really important, because that's what science is, all this data… and you have to present it to other people and make it make sense." Another teacher expressed that, *"the vastness of data was a really cool aspect of this, being overwhelmed by that was cool, and connecting that to the real world and professionals using real data. And ... if it's a dataset that is too huge, looking at the whole thing is pretty useless until you process it."*

Insights from teachers implementing the instructional unit suggest that conceptual scaffolding to connect the *story of a datum* and *thinking in rows* is critical to support student reasoning about data moves that enable analysis of large datasets, and the relationship of those moves to data visualization. Despite their lived experience with air pollution health concerns, several teachers commented that students were unfamiliar with how air pollution was measured or transformed into AQI, and noted the value of engaging their students in discussions about how the data were produced — from source to sensor to data table — to support student sensemaking and reasoning with data. In response to survey questions about their students' learning, a significant proportion of teachers identified understanding and interpreting graphs as important learning achievements: *"the idea of a graph telling a whole story and a point telling part of a story, how to build code to create a graph, and how graphs help us make sense of a large dataset."*

A few teachers also noted that some students experienced challenges in strategically planning an algorithm to produce a desired data visualization, and interpreting the resultant graphical representation to evaluate how well it answered a specific question. These interpretive challenges were more pronounced when students employed more complex data moves. In particular, the *group by* block in the Data Studio VPE can be used to create groups of data for a given variable (e.g., pollutant name, city name). One teacher commented that grouping data seemed "more abstract" for students, and during pilots, researchers observed that algorithms employing the *group by* block were more challenging for students to critique and debug. The conceptual challenges of grouping data intersected with students' greater familiarity with graph types (e.g., bar, line) requiring data for a given variable to be grouped to avoid returning errors (e.g., that multiple y values exist for each x value). In other words, the visualizations students were most comfortable with were the most dependent on correctly implementing the *group by* functionality. Through tinkering and experimentation, students discovered that grouping data would address errors (or, in the case of scatter plots, clarify which points were associated with specific variable attributes), but their use of the *group by* block was more trial-and-error than intentional or strategic. Relatedly, several teachers shared that grouping data presented conceptual challenges for their students: *"[for] students who struggle with graphing, it was hard to conceptualize the group by, and think like well, 'What is it making bars of, and is that answering your question?' So that [block] was the one I saw not always used appropriately by students."* As a result of the COVID-19 pandemic, the core pedagogical supports for more sophisticated data moves like grouping data — the Critiquing Code discourse routine; structured collaboration and reflection during coding; kinesthetic and unplugged activities, and modeling with whiteboard code blocks — were significantly impacted by the pivot to remote instruction. Thus, further research is needed to examine the potential impact of these pedagogical supports.

5 CONCLUSIONS

In synthesizing across themes described in Section 4, we offer the following recommendations for instructional designers seeking to integrate computational data practices in science classrooms.

Ground learning in real-world, locally-relevant contexts. Engaging students in exploring and analyzing large datasets to make sense of locally-relevant, real-world phenomena affords opportunities to draw on their lived experiences and funds of knowledge [38] to motivate learning and support civic participation and critical discourse [40]. Further, such approaches can position CS and data science in service of addressing meaningful problems that matter beyond the walls of the classroom.

Support understanding of dataset structure. Large datasets must be structured in specific ways to enable computational analysis [52]. Yet, these dataset structures are uncommon in middle school science classrooms, and students need explicit support to attend to how data is organized to engage in data practices that can reveal relationships and patterns in the data [29, 30].

Scaffold conceptual engagement with 'data moves.' Our experiences point to the value of learning experiences that move beyond line-by-line procedural instruction toward deeper understanding of why particular procedures are instrumental for particular analyses or questions. We've found that this deeper, more conceptual understanding can be facilitated and made more accessible through iterative experiences with multiple learning modalities.

Build facility with reasoning about data. Insights from implementation highlight the importance of providing structured sensemaking opportunities across the *story of a datum* — a conceptual throughline from a datum collected in the real world to a row in a data table to a specific component of a data visualization resulting from data processing. This is particularly salient for student engagement with data practices that enable them to ask questions about large datasets, and communicate their understanding of data representations that answer those questions.

6 ACKNOWLEDGMENTS

This work was funded by the National Science Foundation under grant 1657002. We are grateful to the teachers and students who participated in the study.

REFERENCES

[1] Neil Anderson, Colin Lankshear, Carolyn Timms, and Lyn Courtney. 2008. 'Because it's boring, irrelevant and I don't like computers': Why high school girls avoid professionally-oriented ICT subjects. *Computers & Education* 50, 4 (2008), 1304–1318.

[2] Megan Bang, Bryan Brown, Angela Calabrese Barton, Ann S Rosebery, and Beth Warren. 2017. Toward more equitable learning in science. In *Helping students make sense of the world using next generation science and engineering practices*. NSTA Press, 33–58.

[3] Sasha Barab and Kurt Squire. 2004. Design-based research: Putting a stake in the ground. *The Journal of the Learning Sciences* 13, 1 (2004), 1–14.

[4] Valerie Barr and Chris Stephenson. 2011. Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads* 2, 1 (2011), 48–54.

[5] Angela Calabrese Barton and Kimberley Yang. 2000. The culture of power and science education: Learning from Miguel. *Journal of Research in Science Teaching* 37, 8 (2000), 871–889.

[6] Chaitan Baru. 2016. Harnessing the data revolution: A perspective from the National Science Foundation. In *National Data Integrity Conference-2016*. Colorado State University. Libraries.

[7] Jennie S Brotman and Felicia M Moore. 2008. Girls and science: A review of four themes in the science education literature. *Journal of Research in Science Teaching* 45, 9 (2008), 971–1002.

[8] Bryan A Brown. 2006. "It isn't no slang that can be said about this stuff": Language, identity, and appropriating science discourse. *Journal of Research in Science Teaching* 43, 1 (2006), 96–126.

[9] Lori Carter. 2006. Why students with an apparent aptitude for computer science don't choose to major in computer science. *ACM SIGCSE Bulletin* 38, 1 (2006), 27–31.

[10] Paul Cobb, Jere Confrey, Andrea DiSessa, Richard Lehrer, and Leona Schauble. 2003. Design experiments in educational research. *Educational Researcher* 32, 1 (2003), 9–13.

[11] K-12 Computer Science Framework Steering Committee et al. 2016. *K-12 Computer Science Framework*. ACM.

[12] National Research Council. 2012. *A Framework for K-12 Science Education: Practices, crosscutting concepts, and core ideas*. National Academies Press.

[13] Wendy DuBow and AS Pruitt. 2018. NCWIT scorecard: The status of women in technology. *NCWIT, Boulder* (2018).

[14] Tim Erickson, Michelle Wilkerson, William Finzer, and Frieda Reichsman. 2019. Data moves. *Technology Innovations in Statistics Education* 12, 1 (2019).

[15] Ian Foster. 2006. A two-way street to science's future. *Nature* 440, 7083 (2006), 419.

[16] James G Greeno. 1998. The situativity of knowing, learning, and research. *American Psychologist* 53, 1 (1998), 5.

[17] Eric Greenwald and Ari Krakowski. 2021. Integrating Computer Science in Science Classrooms: Learning Computational Thinking and Expanding Perceptions of Computer Science. In *Proceedings of the NARST 2021 Annual International Conference*.

[18] Shuchi Grover, Gautam Biswas, Amanda Dickes, Amy Farris, Pratim Sengupta, Beth Covitt, Kristin Gunckel, Alan Berkowitz, John Moore, Golnaz Arastoopour Irgens, et al. 2020. Integrating STEM and computing in PK-12: Operationalizing computational thinking for STEM learning and assessment. In *Proceedings of the 14th International Conference of the Learning Sciences*. 1479–1486.

[19] Shuchi Grover, Kathryn Fisler, Irene Lee, and Aman Yadav. 2020. Integrating Computing and Computational Thinking into K-12 STEM Learning. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 481–482.

[20] Shuchi Grover and Roy Pea. 2013. Computational thinking in K–12: A review of the state of the field. *Educational researcher* 42, 1 (2013), 38–43.

[21] Shuchi Grover, Roy Pea, and Stephen Cooper. 2014. Remedying misperceptions of computer science among middle school students. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*. 343–348.

[22] Toby Dylan Hocking. 2021. Wide-to-tall Data Reshaping Using Regular Expressions and the nc Package. *The R Journal* (2021).

[23] Wendy Huang and Chee-Kit Looi. 2021. A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Computer Science Education* 31, 1 (2021), 83–111.

[24] Golnaz Arastoopour Irgens, Knight Simon, Alyssa Wise, Thomas Philip, Maria C Olivares, Sarah Van Wart, Sepehr Vakil, Jessica Marshall, Tapan S Parikh, M Lisette Lopez, et al. 2020. Data literacies and social justice: Exploring critical data literacies through sociocultural perspectives. In *Proceedings of the 14th International Conference of the Learning Sciences*. 406–413.

[25] Carey Jewitt, Gunther Kress, Jon Ogborn, and Charalampos Tsatsarelis. 2001. Exploring learning through visual, actional and linguistic communication: The multimodal environment of a science classroom. *Educational Review* 53, 1 (2001), 5–18.

[26] Kemi Jona, Uri Wilensky, Laura Trouille, MS Horn, Kai Orton, David Weintrop, and Elham Beheshti. 2014. Embedding computational thinking in science, technology, engineering, and math (CT-STEM). In *Future Directions in Computer Science Education Summit Meeting, Orlando, FL*.

[27] Lisa Kaczmarczyk and Renee Dopplick. 2014. *Rebooting the pathway to success: Preparing students for computing workforce needs in the United States*. ACM Education Policy Committee.

[28] Yasmin Kafai, Gautam Biswas, Nicole Hutchins, Caitlin Snyder, Karen Brennan, Paulina Haduong, Kayla DesPortes, Morgan Fong, Virginia J Flood, Oia Walker-van Aalst, et al. 2020. Turning bugs into learning opportunities: understanding debugging processes, perspectives, and pedagogies. In *Proceedings of the 14th International Conference of the Learning Sciences*. 374–381.

[29] Kim Kastens, Ruth Krumhansl, and Irene Baker. 2015. Thinking big. *The Science Teacher* 82, 5 (2015), 25–31.

[30] Melissa K Kjelvik and Elizabeth H Schultheis. 2019. Getting messy with authentic data: Exploring the potential of using data from scientific research to support student data literacy. *CBE—Life Sciences Education* 18, 2 (2019), 1–8.

[31] Ari Krakowski, Eric Greenwald, Jake Duke, Meghan Comstock, and Natalie Roman. 2020. Integrating Computer Science in Science: Considerations for Scale. In *Proceedings of the 14th International Conference of the Learning Sciences*.

[32] Jean Lave and Etienne Wenger. 1991. *Situated learning: Legitimate peripheral participation*. Cambridge University Press.

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

[34] Irene Lee, Fred Martin, and Katie Apone. 2014. Integrating computational thinking across the K–8 curriculum. *ACM Inroads* 5, 4 (2014), 64–71.

[35] Victor R Lee and Michelle H Wilkerson. 2018. Data use by middle and secondary students in the digital age: A status report and future prospects. (2018).

[36] Joyce Malyn-Smith, Irene A Lee, Fred Martin, Shuchi Grover, Michael A Evans, and Sarita Pillai. 2018. Developing a framework for computational thinking from a disciplinary perspective. In *Proceedings of the International Conference on Computational Thinking Education*. 182–186.

[37] Jane Margolis, Jean J Ryoo, Cueponcaxochitl DM Sandoval, Clifford Lee, Joanna Goode, and Gail Chapman. 2012. Beyond access: Broadening participation in high school computer science. *ACM Inroads* 3, 4 (2012), 72–78.

[38] Luis C Moll, Cathy Amanti, Deborah Neff, and Norma Gonzalez. 1992. Funds of knowledge for teaching: Using a qualitative approach to connect homes and classrooms. *Theory into Practice* 31, 2 (1992), 132–141.

[39] Don Passey. 2017. Computer science (CS) in the compulsory education curriculum: Implications for future research. *Education and Information Technologies* 22, 2 (2017), 421–443.

[40] Thomas M Philip, Sarah Schuler-Brown, and Winmar Way. 2013. A framework for learning about big data with mobile technologies for democratic participation: Possibilities, limitations, and unanticipated obstacles. *Technology, Knowledge and Learning* 18, 3 (2013), 103–120.

[41] Mitchel Resnick. 2013. Learn to code, code to learn. *EdSurge, May* 54 (2013).

[42] Lindy Ryan. 2018. *Visual Data Storytelling with Tableau: Story Points, Telling Compelling Data Narratives*. Addison-Wesley Professional.

[43] Allison Scott, F Kapor Klein, Frieda McAlear, Alexis Martin, and Sonia Koshy. 2018. *The Leaky Tech Pipeline: A Comprehensive Framework for Understanding and Addressing the Lack of Diversity across the Technology Ecosystem*. Technical Report. Technical Report. https://mk0kaporcenter5ld71a. kinstacdn. com/wp-content ….

[44] Pratim Sengupta, Amanda Dickes, and Amy Farris. 2018. Toward a phenomenology of computational thinking in STEM education. *Computational thinking in the STEM disciplines* (2018), 49–72.

[45] Pratim Sengupta, John S Kinnebrew, Satabdi Basu, Gautam Biswas, and Douglas Clark. 2013. Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies* 18, 2 (2013), 351–380.

[46] David W Shaffer. 2006. Epistemic frames for epistemic games. *Computers & Education* 46, 3 (2006), 223–234.

[47] Eric Snow, Daisy Rutstein, Marie Bienkowski, and Yuning Xu. 2017. Principled assessment of student learning in high school computer science. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*. 209–216.

[48] NGSS Lead States. 2013. Next Generation Science Standards: For states, by states.

[49] Tamara Van Gog, Liesbeth Kester, and Fred Paas. 2011. Effects of worked examples, example-problem, and problem-example pairs on novices' learning. *Contemporary Educational Psychology* 36, 3 (2011), 212–218.

[50] David Weintrop, Elham Beheshti, Michael Horn, Kai Orton, Kemi Jona, Laura Trouille, and Uri Wilensky. 2016. Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology* 25, 1 (2016), 127–147.

[51] Linda L Werner, Brian Hanks, and Charlie McDowell. 2004. Pair-programming helps female computer science students. *Journal on Educational Resources in Computing (JERIC)* 4, 1 (2004), 4–es.

[52] Hadley Wickham. 2014. Tidy data. *Journal of Statistical Software* 59, 1 (2014), 1–23.

[53] Michelle Hoda Wilkerson and Joseph L Polman. 2020. Situating data science: Exploring how relationships to data shape learning. *Journal of the Learning Sciences* 29, 1 (2020), 1–10.

[54] Eben B Witherspoon, Ross M Higashi, Christian D Schunn, Emily C Baehr, and Robin Shoop. 2017. Developing computational thinking through a virtual robotics programming curriculum. *ACM Transactions on Computing Education (TOCE)* 18, 1 (2017), 1–20.

[55] Hsin-Kai Wu and Joseph S Krajcik. 2006. Inscriptional practices in two inquiry-based classrooms: A case study of seventh graders' use of data tables and graphs. *Journal of Research in Science Teaching* 43, 1 (2006), 63–95.

[56] Aman Yadav, Hai Hong, and Chris Stephenson. 2016. Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends* 60, 6 (2016), 565–568.

[57] Kimberly Michelle Ying, Lydia G Pezzullo, Mohona Ahmed, Kassandra Crompton, Jeremiah Blanchard, and Kristy Elizabeth Boyer. 2019. In their own words: Gender differences in student perceptions of pair programming. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. 1053–1059.