A PLACE FOR A DATA SCIENCE INTRODUCTION IN SCHOOL: BETWEEN STATISTICS AND PROGRAMMING

SUSANNE PODWORNY
Paderborn University, Germany
podworny@math.upb.de

SVEN HÜSING
Paderborn University, Germany
sven.huesing@upb.de

CARSTEN SCHULTE
Paderborn University, Germany
carsten.schulte@upb.de

ABSTRACT

Data science surrounds us in contexts as diverse as climate change, air pollution, route-finding, genomics, market manipulation, and movie recommendations. To open the "data-science-black-box" for lower secondary school students, we developed a data science teaching unit focusing on the analysis of environmental data, which we embedded in a ninth-grade computer science class. In this unit, students experience a new insight-driven programming approach, using Jupyter Notebook and the programming language Python for their data analysis. In this paper, we evaluate the second cycle of this project, report how the students coped with the Jupyter Notebooks for doing statistical investigations and describe the insights they gained.

Keywords: *Statistics Education Research; Data Science Education; Reproducible Data Analysis; Epistemic Programming; Jupyter Notebooks; Python*

1. INTRODUCTION

Data science is becoming increasingly important in a world awash with data and affects the daily life of almost everyone (Ridgway et al., 2018). The topic has received increased attention in German schools in recent years. There is a consensus not to introduce data science only from a simplistic and isolated point of view, but rather to show its complexity when working with authentic data. A project-based learning approach (Garfield & Ben-Zvi, 2008) is a suitable starting point in school that allows for authentic and rich data. However, the didactic requirements of project-based learning are made even more complex in the context of data science because this is essentially an interdisciplinary field, incorporating statistics, computer science, and the related domains (Biehler & Schulte, 2018; Ben-Zvi, 2018).

Computer-supported data analysis is an important feature of data science education. Since there is only limited space for a long, complex project in the mathematics class, one option is to place such a project in computer science. Additionally, programming is a powerful tool for data analysis (Biehler, 1997), so students can build on and enhance their programming skills by using programming instead of pre-made tools when they analyze data.

How does programming change how we learn about data? What new insights do we get through programming that we do not get any other way? We are interested in the deep connection between the programming process itself and the insights that this process reveals. We have attempted to design our teaching unit to take advantage of that connection, and call this approach *didactical epistemic programming* (Hüsing, 2021) in accordance with Hershkowitz et al. (2001) and Bereiter (1980).

Regarding data analysis projects, epistemic programming represents a departure from the typical computer science curriculum, as data analysis projects are different from the product- or algorithm-oriented tasks typical of computer science classes. This change of emphasis accomplishes two things: first, it introduces students to the somewhat different, but equally important, world of computation with

data; and second, it helps students apply computational results to the real world, also emphasizing the power of programming. This second point is crucial. Data makes computing directly relevant to the real-world lives of students, and thereby generates interest. This point builds on decades of work and insights from the statistics education community (e.g., Biehler et al., 2015, Wild & Pfannkuch, 1999; Chance et al., 2007).

In the data science project presented in this paper, students use (epistemic) programing to develop inquiry-based reasoning skills in a data investigation, and actively gain knowledge about their surroundings by analyzing authentic data they collect from the local environment (e.g., temperature). Students analyze their data by adapting and executing Python-Code within *Jupyter Notebooks*.

The theoretical framework of epistemic programming aligns with the purpose of the study and can be a beneficial framework for data science education in general. This approach assumes that new knowledge is created from using programming tools and unites three essential elements of data science: programming/coding, (statistical) interpretation, and an authentic context. It ties in with the notion of epistemic writing from mathematics (Strohmaier et al., 2018) and development in writing (Bereiter, 1980) and combines adequately using digital tools with understanding the non-digital world (Schulte, 2013).

Two major conjectures driving the current study are that (1) students are motivated to work with data that are personally relevant to them (Garfield & Ben-Zvi, 2008; Lesser, 2007) and (2) students with nearly no pre-knowledge in programming can use a prepared professional tool to gain insights from data. The goal of the current exploratory study is to investigate these conjectures by answering the research question: How do students utilize *Jupyter Notebooks* in an introductory data science project using the new epistemic programming approach for doing statistical investigations?

Regarding such investigations, Chance and colleagues (2007) emphasize the importance of interpreting statistical results in context. We know that it is a challenge for students in statistics classes to interpret data in context and more precisely, to relate statistical findings to the context (e.g., Biehler, 1997; Biehler et al., 2015; Pfannkuch, 2007). A data science project in a computer science class may face these difficulties as well. But, as the focus of the project is not on statistics, but on using programming to examine the real-world situation using a data science lens.

2. THEORETICAL BACKGROUND

This teaching unit, based on the didactical epistemic programming approach, focuses on giving lower secondary school students their first experiences in data science by implementing a data science project in a computer science class in which students use a professional programming language. In this section, we outline the approaches that we used to design the elements of the unit in a ninth-grade computer science class from statistics and computer science perspectives.

2.1. FRAMEWORK FOR A PROJECT-BASED DATA SCIENCE TEACHING UNIT

Project-based learning centers around authentic, real-world activities. Projects are widely seen as effective vehicles for teaching and learning statistics (Garfield & Ben-Zvi, 2008), learning data science, and for providing learning experiences in statistical investigations (Makar & Fielding-Wells, 2011). Projects move statistics out of the classroom, contextualize its application, and demonstrate the usefulness of statistics in everyday life (Verhoeven, 2013). This active learning attracts students' attention and raises their motivation (Fincher & Petre, 1998; Krajcik & Blumenfeld, 2006; Bilgin et al., 2015). Projects also give students a context for exploring and experiencing fundamental statistical ideas such as data representations and variability (Burrill & Biehler, 2011).

Five features are central to a project-based learning environment. Students typically (1) start with an authentic problem or question, (2) explore the problem or question by participating in an authentic inquiry adapted from expert performance, (3) use collaborative activities, (4) use technology in parts of the inquiry process to extend their abilities, and (5) create tangible products (Krajcik & Blumenfeld, 2006, p. 318). From a statistics perspective, in projects, students "have the primary responsibility of formulating the data collection plan, actively collecting the data, analyzing the data, and then interpreting the data to a general audience" (Chance, 2002, p. 6), while using their previously gained

knowledge and skills (Fincher & Petre, 1998). In our computer-science context, the additional knowledge and skills are in the domain of programming.

In this regard, our approach also fits the GAISE guidelines for K–12 education (Bargagliotti et al., 2020). Concretely, we developed a framework combining approaches from statistics like the PPDAC Cycle (Wild & Pfannkuch, 1999), and computer science such as the CRISP-DM model (Shearer, 2000). The PPDAC Cycle contains the phases of stating a problem, planning a data collection and analysis, collecting data, analyzing data, and drawing conclusions. The CRISP-DM is a process model used by data mining experts and contains the phases of business understanding, data understanding, data preparation, modeling, evaluation, and deployment. The respective new insight-driven framework (Figure 1) focuses on newly acquired insights as a nexus connecting four "areas of action." Within the framework, there is an interrelation between each of the four areas of action (pose questions and develop ideas; collect and prepare data; explore, analyze, and visualize; and interpret and communicate) and the area of new insights. It is possible to use the insights gained from one action in one area to initiate another action in another or the same area.

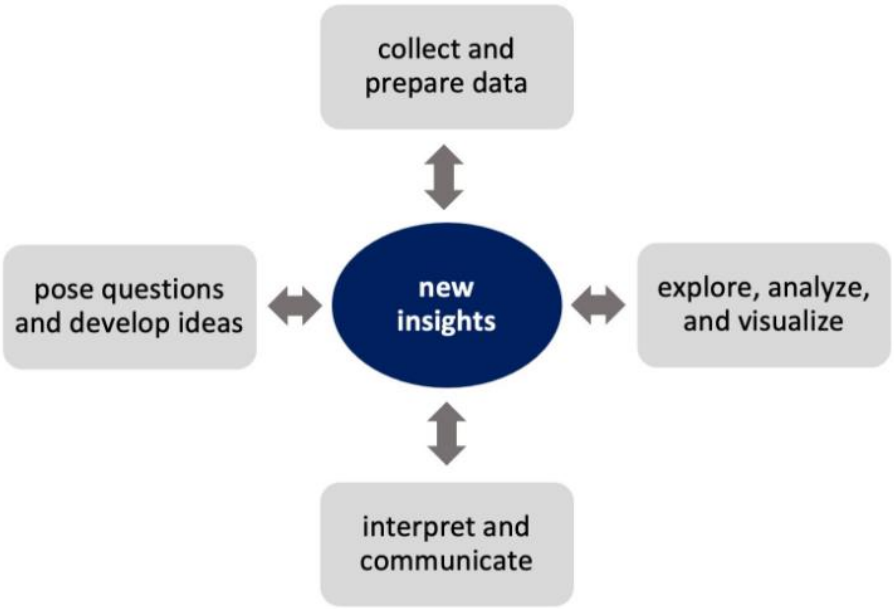

Figure 1. Framework showing how new insights connect areas of action in a data project (based on Höper et al., 2021)

2.2. REPRODUCIBLE DATA ANALYSIS

According to McNamara (2015), one of the main properties of a statistical tool is "reproducibility". This means that the process of data analysis can be rerun several times with the same or another version of the data to get comparable results (McNamara, 2019, Kandel et al, 2011; Sandve et al., 2013). Reproducibility also means that the process of the data analysis can be saved for later use, which is not possible in interactive tools that do not track the single data-analysis steps (e.g., Excel, CODAP, or similar tools). By using such interactive tools, the results of the data analysis are immediately visible, but the process of getting them is not (easily) repeatable by somebody else (McNamara, 2019; Biehler, 1997). In contrast, if one uses programming for conducting data analyses, the program code is already part of the documentation needed to understand and reproduce this process. Written program code is traceable, and lets programmers develop parts of the code through multi-step cycles of designing, writing, and testing (Caspersen, 2007; Pattis, 1990).

In summary, the goal of a reproducible data analysis approach is to gain knowledge in a traceable process and to present the new insights to make this cognition process accessible to others. Thus, we conclude that a tool that allows doing reproducible data analysis fosters the application of an epistemic approach to a data science project.

2.3. EPISTEMIC PROGRAMMING

The newly developed didactical approach of epistemic programming (Hüsing, 2021) is the source for "new insights" in this project. Epistemic programming has the purpose of acquiring new knowledge that goes beyond the activity of programming itself and cannot be obtained without the programming process, or only with great difficulty. The approach relates to the dimension of "thinking" or "coping with affordances", which Schulte (2013) mentions as essential to the educational role of programming. On the one hand, students learn to handle the affordances of the digital tools used (internal knowledge). On the other hand, they get new insights about the non-digital world (external knowledge).

Put another way, epistemic programming balances and integrates two worlds. In the "internal" world, students learn about the "architecture" of the digital artifacts being programmed. In the "external" world, students explicitly focus on the exploration of the world, interpreting results of their data science project in the relevant societal context (Kroes & Meijers, 2006; Schulte & Budde, 2018). Here, the epistemic programming approach extends the scientific tradition of programming (Tedre & Apiola, 2013) and allows for cross-curricular and interdisciplinary teaching in school.

Science education faces a similar issue. Students typically acquire scientific insights and results by reading and hearing about scientific topics. However, according to Kay (2007), students should also get the chance to extract knowledge through activities that relate to the corresponding scientific field, for example, through experimentation. Here, we make a parallel argument: Data analysis is about the extraction of knowledge from data (Kandel et al., 2011; Guo & Seltzer, 2012). Therefore, data science and data analysis are perfect for this didactical programming approach. For us, the student's goal is to acquire new knowledge from data, using programming as a tool (Winkelnkemper, 2018), and not to create a product or a system as is often the case in classes, following the engineering tradition of programming (Tedre & Apiola, 2013). When programming epistemically, students always have to reflect their (interim) results in the domain context before revising their program code in order to achieve the desired results. Thus, epistemic programming alternates between code-writing and reflecting on the real-world-context. This creates an interdisciplinary endeavor, combining real-world contexts with programming as a way to gain new insights and knowledge in the scientific fields that are relevant to the respective context (Chance, 2002).

2.4. THE (DIDACTICAL) USE OF JUPYTER NOTEBOOKS

Considering the interdisciplinary character of epistemic programming as well as the benefit of reproducible data analysis, there is a need for a programming environment that contains insights about the real-world context, the programmed code, and its explanation in one place (Knuth, 1984). In this regard, Rule et al. (2018) suggest using *Jupyter Notebooks* to carry out data science projects and investigations.

A Jupyter Notebook is essentially an interactive document, which can be accessed via an internet browser. A notebook appears as a sequence of cells of different types. In this setting, we are mostly concerned with *markdown* cells (Figure 2), which contain instructions or explanations; and *code* cells (Figure 3), which contain *Python* code that one can execute in place.

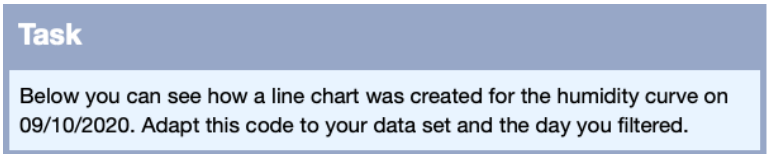

Figure 2. Example for a task, written in a markdown cell

Figure 3. Example for a code cell with pre-written code and its execution. The students had to change the data set (df100920) here as well as the date in the visualization. The number [6] in square brackets indicates the execution order: This cell was the sixth one the student executed during the session.

According to Perez and Granger (2015), the results of a data analysis and the data itself have to be combined with narrative text to make the code and the results accessible and understandable, thus *reproducible*, for the human reader. Working in *Jupyter Notebooks* therefore helps students implement this reproducible research approach (Rule et al., 2018).

We decided to use the programming language *Python* as it enables students to carry out data analyses using existing powerful libraries for data analysis as well as for exploring and creating AI systems in future projects, which we want to build on in the ProDaBi-project (see Fleischer et al., this issue). In addition, *Python* is very readable compared with other scripting languages through its simple structure and clear syntax (Nagpal & Gabrani, 2019).

From a didactical perspective, *Jupyter Notebooks* allow a teacher to guide and support students' work by creating "prepared" notebooks. A prepared notebook may consist of markdown cells with tasks (see example in Figure 2), prepared and adaptable code cells (like in Fig. 3), and explanations. It should be noted that the less students are used to a programming language, the more preparation a teacher may need to do.

3. METHODOLOGY

The first cycle of the data science project analyzing environmental data was developed by the ProDaBi project group (www.prodabi.de) for use in a non-compulsory "project course" in Grade 12 (aged 17-19 years). It was further developed for use in lower secondary schools (Grade 9, aged 14-16) using the design-based research paradigm (Cobb et al., 2003). This study reports on students' work from that second cycle.

3.1. OUR APPROACH TO A DATA SCIENCE PROJECT IN A COMPUTER SCIENCE CLASS: THE TEACHING UNIT

The approach to the data science project described in this article differs from regular teaching units in statistics classes. While statistics is mainly a topic for mathematics classes in German schools, our approach addresses this topic in a computer science teaching unit. In our ideal, students carry out a project completely on their own. They are free to explore the data and can pursue individual questions with their own data, as Tissenbaum et al. (2019) suggest in their computational action approach. This teaching unit is a step towards that ideal. The students use professional tools in the form of prepared *Jupyter Notebooks* and the *Python* programming language to perform the data analysis. By using such

prepared tools, the students get familiar with methods and processes used by "real" data scientists (Kay, 2007; Odden & Malthe-Sorenson, 2021). Concretely, gaining insights into (1) the environmental data context, (2) programming for data analyses, and (3) doing statistical investigations are the goals of the teaching unit. In order to develop materials for our data science project, we used the four criteria for projects described by Rubin and Mokros (2018), which were designed to give lower secondary-school students a "taste" of data science. These criteria are "topics, datasets, tools, and activities" (p. 1).

The teaching unit (the data science project) consists of 12 lessons of 45 minutes each. Students get to know basic methods of data analysis and learn to use *Python* within *Jupyter Notebooks* (tool criterion). Using this knowledge, the students conduct their data analysis with environmental data (datasets) and document their programming process, findings, and interpretations in a Jupyter Notebook (activity criterion), to gain insights into their environment (topic criterion).

Specifically:

- At the beginning of the unit, sensor boxes (www.sensebox.de/en) are programmed to collect local environmental data for several days (tool and dataset criteria).
- The students formulate several questions to be answered through data analysis (activity criterion) based on the data they expect.
- While the sensor boxes collect data (dataset criterion), a six-hour online course introduces *Python* and *Jupyter Notebooks* (activity criterion).
- Using the didactical approach of epistemic programming, the students are then asked to analyze the environmental data, visualize it, and formulate results using prepared worked examples (Atkinson et al., 2000) to answer their initial questions (activity criterion) in the following four lessons.
- At the end of the teaching unit, the students summarize their findings regarding the insights about their local surrounding in a presentation by interpreting their visualizations from their *Jupyter Notebooks* (topic and activity criteria).

3.2. IMPLEMENTING THE TEACHING UNIT

The teaching unit was started as planned in September 2020 but was interrupted by a two-week quarantine due to COVID-19 and an additional two weeks of autumn holidays. With the COVID-situation, students were restricted in their use of the computers; only one student was allowed to use the computer while the other in a pair had to keep a distance of 1.5 meters. Exchange between pairs was limited because they were not allowed to walk around. The hoped-for positive effects of group work during the programming phase and data analysis were thus unfortunately reduced. At the end of the teaching unit, a second quarantine situation was imposed, using distance learning, so the teacher decided to ask for summaries in Word documents and abandoned classroom presentations because the *Jupyter Notebook* environment was not accessible from students' homes.

3.3. PARTICIPANTS

The data science project took place in a German secondary school in a ninth-grade computer science class with 23 students aged 13–15 years. The students had some prior experiences with programming in *Scratch* but no experience with a text-based programming language. *Python* was completely new to them. The students also had little prior experience in statistics; they knew about absolute and relative frequencies, bar charts, boxplots, and measures like mean, median, minimum, and maximum, but were not accustomed to interpreting statistical diagrams. Their socio-economic background was mainly middle class with good technical facilities.

3.4. THE DATA SETS USED IN THE PROJECT

In August and September 2020, two sensor boxes (Figure 4) were programmed in class with the help of the teacher, and then set up to collect environmental data.

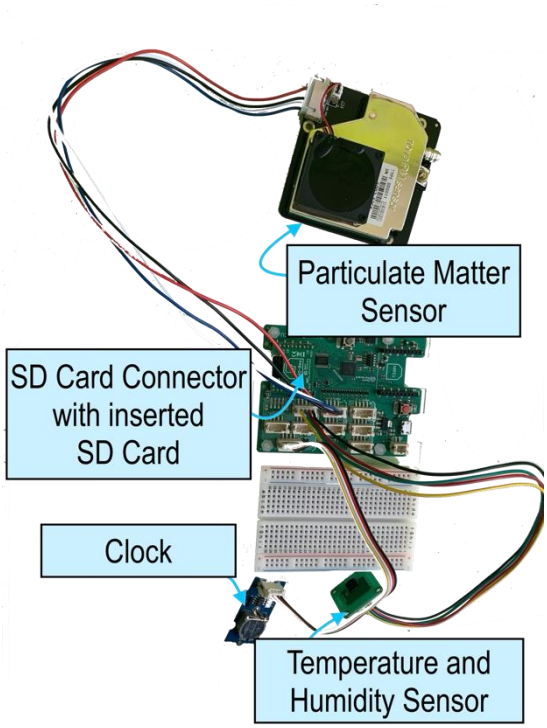

Figure 4. The sensor box used in the teaching unit

The first sensor box was installed near a busy road for three days, while the second one was attached to a window outside of the school. The school was located in a park and at quite a distance to the nearest street. Each data set contained the variables "timestamp" and "value", where "value" records the output of a sensor. We used three sensors: temperature, humidity and particulate matter, all of which collected data at five-minute-intervals. This resulted in three data sets each with about 900 cases for the sensor box on the street, which collected data from Thursday to Sunday, September 10–13, 2020. Due to a problem with the power supply, the box at the school collected only temperature and humidity data.

3.5. DATA COLLECTION

All final *Jupyter Notebook* files (n = 12) and reports (as Word documents) (n = 13) of the students were collected to allow an analysis of students' use of the *Jupyter Notebook* environment and the interpretation of results. Additionally, after the unit was done, the students completed an online survey to assess their programming and statistics competence and reporting on their overall attitudes towards the teaching unit. Nineteen students (10 female, 8 male, 1 non-binary) participated in this survey, which consisted of many different statements to be rated on a four-point Likert scale and two open-ended questions. For the analysis, we aggregated the two items "agree" and "tend to agree" as well as "tend to disagree" and "disagree" to get an overview of how the project was received by students.

3.6. ANALYSIS METHOD

The analysis happened in two steps. First, we analyzed and interpreted the students' work in the *Jupyter Notebooks* and reports submitted. Here, a systematic approach was taken to describe and quantify phenomena with qualitative methods. Second, we evaluated the quantitative and qualitative results from the survey.

In the first step, we categorized how the students used the notebooks (which variables they used and combined, which visualizations they created, which (prepared) code they used/adapted, and in what order, etc.) using the method of qualitative content analysis (Mayring, 2015) with regard to the research question.

Recall that a number in brackets at the beginning of a *Jupyter Notebook* cell (e.g., [6]) indicates the execution order of that cell. We used these numbers to infer the students' use of the *Jupyter Notebook*. We categorized the adapted or newly added code using a deductive category system that was elaborated by the authors in advance. Additionally, we summarized and categorized students' reports accordingly (Which visualizations? How many visualizations? Which data did they use? How did they describe and interpret the visualizations?).

The survey was evaluated using frequencies and the categorization of the open-ended qualitative answers (e.g., What are your top 3 insights and experiences from the project?).

4. RESULTS AND DISCUSSION

In this section, we present results from the analysis of the student work in the *Jupyter Notebooks* and the Word document reports and discuss the students' self-assessment from the survey.

4.1. STUDENTS' USE OF *JUPYTER NOTEBOOKS*

The first remarkable result concerning the notebooks is that all students used programming cells but no markdown cells. This means that all explanations and interpretations happened orally during the data analysis and were written down only in the Word documents. This was probably due to the distance-learning situation at the end of the teaching unit. Students could not access their *Jupyter Notebooks* from home, so the teacher advised them to copy their diagrams to Word documents and interpret them there.

Concerning *Jupyter Notebooks*, the number of programming cells and their execution number was an astonishingly good source for the analysis. An insight arising from the study of the execution numbers is that students used the adapted or self-written code to facilitate reproducibility: Since there were different sets of data that the sensor boxes collected (temperature, humidity, and particulate matter data), the analysis process in the corresponding notebook had to be done several times. The analysis of the students' notebooks showed that at least four student groups ran their whole *Jupyter Notebook* multiple times but each time with a different data set. This means that they only had to change the read-in data set at the beginning of the notebook to get comparable visualizations and evaluations for the different variables like temperature, humidity, or particulate matter.

As an example of this reproducibility approach, consider the notebook that had a top cell starting with the execution number 27 and consecutive execution numbers in all following code cells. From this, we inferred that the students executed all code cells twice (2 x 12 cells and some cells maybe more often in the first run) and the execution numbers displayed (27 to 38) showed a third run of the whole notebook. Apparently, the three runs were of one dataset each (temperature, humidity, particulate matter). A look at the corresponding reports supported this interpretation because the *Jupyter Notebook* showed only data on particulate matter but the report contained visualizations for three data sets. This supports the assumption that the students worked in a reproducible research manner and thus shows a huge potential of epistemic programming in *Jupyter Notebooks*, as comparable results for multiple data sets are easy to produce (McNamara, 2019, Kandel et al., 2011, Sandve et al., 2013). In particular, this differentiates the use of *Jupyter Notebooks* from interactive tools with direct feedback (such as Excel or Fathom), since the entire data analysis process would have had to be repeated again for different data sets when using such tools.

Additionally, the numbering showed that some students did not work linearly (from top to bottom) within their *Jupyter Notebook* but changed direction and jumped to different places. Two notebooks started with "high" numbers at the beginning followed by smaller numbers. In one *Jupyter Notebook,* for example, the first nine code cells had execution numbers 43–51, while the following code cells had numbers 35–37. Here we infer that the code cells were executed several times from top to bottom (we are not able to interpret the sequence of the first executions from the final notebooks) and the first nine cells were executed once again in the end. Gaps in the numbering suggest that a code cell was executed multiple times at this point. This may be a hint that the students needed several attempts to make the code work or that they wanted to adapt the output further, which fits the idea of a multi-step cycle of testing, designing, and writing the program code, as described by Caspersen (2007) and Pattis (1990), and represented in the framework in Figure 1.

All student groups managed to create meaningful visualizations from the prepared code with the *iplot*-command by filtering a single day from one dataset. Nine groups used the temperature data, and three groups used the particulate matter data (at least in the final execution run). All student groups managed to customize the labels of the time series visualizations by changing the corresponding *Python* code.

One instruction in the prepared code aimed at visualizing the data from multiple days in one graph. Figure 5 shows a graph of how it could have been executed with the prepared code. No group used the prepared code in that way, but most groups adapted the code to display the data for two or more days, one after the other, like in Figure 6.

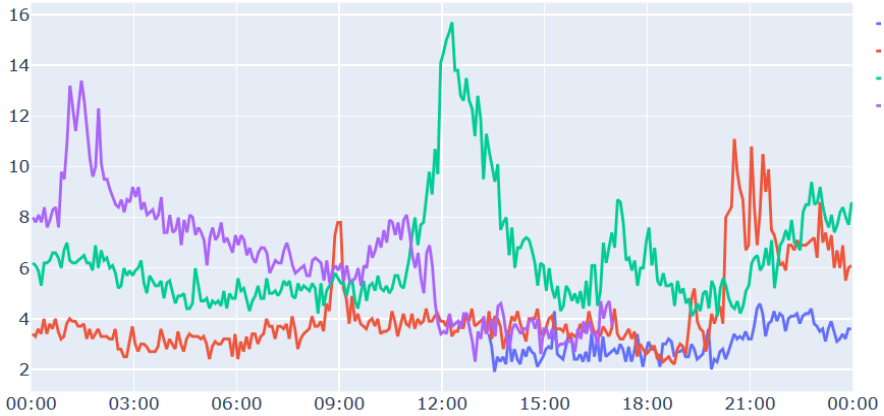

Figure 5. Expected display for particulate matter data for Thursday (blue), Friday (red), Saturday (green) and Sunday (purple)

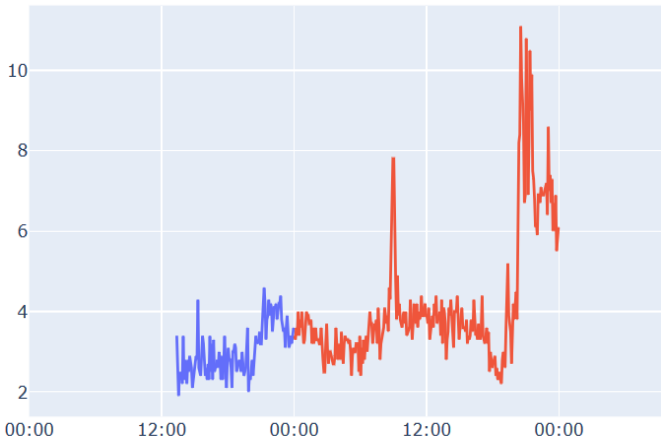

Figure 6. Most common type of display in students' notebooks for particulate matter data for Thursday (blue) and Friday (red).

Nine student groups used a consecutive visualization for one of the three data sets like in Figure 6. They selected one data set (mostly temperature or particulate matter) and selected two days for the visualization. Three groups did not manage to produce any consecutive visualization of data from several days. One group displayed all three data sets in one visualization for three days as shown in Figure 7. In that notebook, there was a gap of 33 executions between this and the next code cell which is an indication that the code had been tinkered with several times. From the gaps in the execution numbering in the corresponding cell for at least ten notebooks, we interpret that nearly all groups faltered at this cell and maybe tried different variants (Caspersen, 2007; Pattis, 1990).

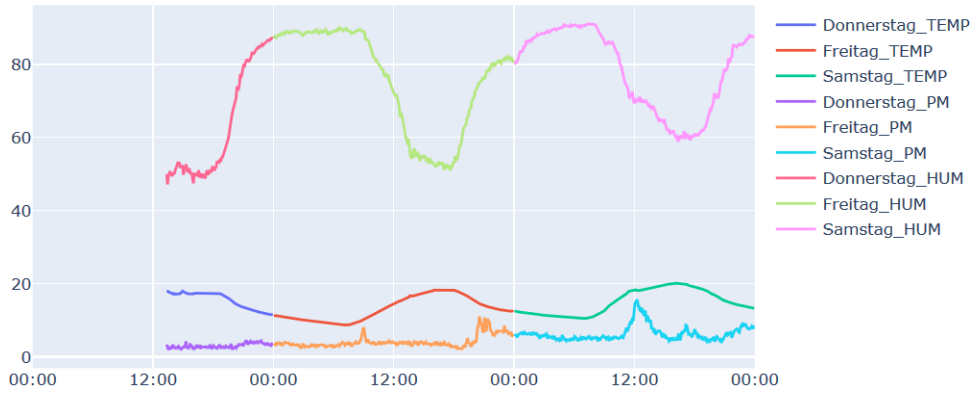

Figure 7. Students' visualization for all three datasets for three days in one display (Donnerstag = Thursday, Freitag = Friday, Samstag = Saturday, TEMP = temperature, PM=particulate matter, HUM = humidity)

4.2. STUDENTS' INTERPRETATIONS OF THEIR VISUALIZATIONS

The student description and interpretation of the data occurred using the Word reports. All groups included visualizations there. However, the visualizations could not always be found in the respective *Jupyter Notebook* files. Those who included the time-series display in Figure 6 had only this visualization in their report, all others included at least three visualizations, one for each data set. Four groups used single visualizations for each day, and one group visualized all days for each data set in a total of nine visualizations.

The reports varied considerably in length. The number of words varied between 86 and 266 (including title, footnotes, etc.). Four student groups wrote or cited short definitions (one or two sentences) for temperature, five did this for humidity, and ten for particulate matter. Three additionally wrote the standard values for the measures, for example, "particulate matter should not exceed 20 µg/m³". This shows that most groups found it necessary to explain the scientific terms and measures used. Background information on the data collected was included by six student groups for location, and by seven for the data-collection-period. So, about half of the groups reported on what might be called "meta-data", which was both contextual and scientific. Due to the fact that almost no specifications were given in advance for the reports by the teacher, we interpret this as a positive outcome.

All visualizations were taken from the notebooks without a heading, but nine groups named the variables shown in their visualizations by using a short title such as "Humidity at school" or in a short description, such as "The temperature rises from the midday hours until late afternoon and then decreases." Unfortunately, the analysis in many reports was limited to such descriptions. Some students presented a few more details. Seven groups briefly mentioned statistical variation. For example,

> *The graph is very unsteady and quite irregular. At 9 a.m. and at about 8 p.m. there are very high peaks of fine particulate pollution.*

Other students' comments were related to parameters such as maximum and minimum. For example,

> *The graph reaches its highest value at 3 p.m. with approx. 35 degrees.*

An interpretation of why this highest value might occur at that time or a connection to the writer's real life was not discussed in any report. Some students, however, tried to find relationships between variables. For example, six groups suspected correlation between temperature and humidity. A group commented,

> *There is certainly a correlation between temperature and humidity, because whenever the temperature rises, the humidity decreases, and when the temperature decreases, the humidity rises.*

Another student wrote,

> *Whenever the humidity has gone down, the temperature has gone up. They are not inversely proportional, because the values always vary a little, but a regularity is recognizable.*

4.3. RESULTS REGARDING THE RESEARCH QUESTION

With regard to the research question "How do students utilize *Jupyter Notebooks* in an introductory data science project using the new epistemic programming approach for doing statistical investigations?" we can state that the preparation of worked examples (Atkinson et al., 2000) enabled all but one student group to use the notebooks for the data science project. In eleven of the twelve notebooks analyzed, several meaningful visualizations were created and partially used for later interpretation. This showed that the students gained some programming knowledge in the areas of action from the framework displayed in Figure 1. Furthermore, this implies that the instructions contained in the prepared notebooks, together with the *Python* online course, enabled the students to develop a certain code sense, which made it possible for them to adapt the prepared code. Notably, they were able to independently exercise the concept of reproducible research (McNamara, 2015; 2019).

Additional results can be drawn from the qualitative answers in the survey to the open question "What are your top 3 insights and experiences during the project?". Eleven of the 19 students who participated in the survey gave statements we interpret to mean that they gained confidence in their ability to program or came to view programming as useful. A typical statement was, "Python is suitable for visualization of data."

Regarding the epistemic programming approach, the students also reflected that the programming activities in this project led to new insights about their own environment. While 74% of the students stated that they gained new insights that they could not have gotten without programming, 53% even indicated that programming was a necessary tool to gain new insights about the environment. One of the students who agreed with both aspects also mentioned that one of the "top 3 insights" was that it became clear that the influence and uses of computer science were applicable and helpful in many fields. An additional insight of the student concerned the relationships among temperature, humidity, and particulate matter and how these can help develop problem solutions. Another student noted that it was motivating to collect and evaluate data to recognize relationships between particulate matter and humidity in order to develop solutions for reducing particulate matter pollution in the future. Here, both students saw programming as a tool to gain knowledge (Winkelnkemper, 2018) that could be helpful for future research and problem-solving (Wing, 2011).

Concerning statistics, we can say that the students did use statistical skills like visualizing data or interpreting data in graphs. However, the majority (68%) of the students would have preferred to receive more information and further statistical knowledge in order to perform the data analysis. About 32% of the students named statistical topics (especially data-evaluation aspects) such as "evaluating environmental data" or "working directly with the data" as particularly motivating. Only one student expressed that using statistical actions (more precisely actions concerning data visualization) was disruptive during the project. Thus, it can be said that although the students were missing some statistical knowledge for the data analysis, some students still recognized the statistical aspects of this project and found them interesting and motivating. Consequently, in future implementations of this project, a greater focus should be placed on the statistical foundations of data analysis and interpreting diagrams by providing students with more information, for example, in the form of prepared worked examples (Atkinson et al., 2000).

With regard to the students' perception of the series of lessons, the local and/or the environmental context was particularly motivating. Students referred to gaining new insights about the environment in general as well as specific environmental variables like particulate matter in their own surroundings. An example for such an insight is,

> *I learned about environmental data in the school's surroundings and I [saw] the need to develop new solutions for the particulate matter pollution in the world.*

This showed clearly that the data analysis led to an awareness for the local environmental situation. Furthermore, it indicated that the environmental and concrete local references in this project are big

motivating factors as predicted by Garfield and Ben-Zvi (2008). Beyond that, they represent a field that the students can discover by doing the data analysis through epistemic programming.

5. SUMMARY AND CONCLUSION

There is international consensus regarding the importance of data and statistics education (Ben-Zvi, Makar, & Garfield, 2018). We succeeded in bringing a detailed, real data science project into the regular lessons of a ninth-grade class. To do this, we placed it in computer science education, facilitating the use of the *Python* programming language. Our approach to a data science project is based on ideas from Krajcik and Blumenfeld (2006) and Rubin and Mokros (2018). Additionally, the didactical approach of epistemic programming (Hüsing, 2021) provided an appropriate framework for active participation of the students in the field of data science (Kay, 2007).

The prepared worked examples (Atkinson et al., 2000) represented an appropriate pedagogical tool to introduce the reproducible research approach (McNamara, 2019), combined with the idea of epistemic programming in a lower secondary school class. It was possible to go beyond methods normally used in grade 9 (Gómez-Blancarte & Ortega, 2018). As the students created their own data analyses through programming, they reported gaining insights about aspects of programming itself like learning the programming language *Python* or reading in real data. They also reported learning about the domain knowledge of their own environment, for example regarding the connection between various measures. In this way, the students learned an adequate use of *Python* and *Jupyter Notebooks* to gain insights about the real world (Schulte, 2013). It turned out that the students coped well with the project overall, despite little prior knowledge as they managed to create meaningful visualizations of the local environmental data. Especially when examining relations among certain variables, the students took advantage of the reproducibility approach to get comparable data visualizations.

Generally, without this active way of working in the context of the domain, it would have been difficult for the students to get and internalize the findings from their data analysis, as Kay (2007) described, and several students confirmed. Concretely, the students used programming as a tool to gain new insights (Hüsing, 2021; Winkelnkemper, 2018), which can serve as a basis for new solutions for—in this case environmental—problems or questions.

However, since the idea of the project was to carry out data explorations from a computer science perspective, the "tools" for statistical interpretation of the results were a little lacking in the end as only a few students made detailed interpretations of their visualizations. To enhance the statistical knowledge that the students were missing, more direct integration of activities about visualizing and interpreting could be helpful, so that they could comment on the programming results in a more intuitive way. However, the incomplete aspects of the student work could also be partially due to the pandemic-related shortening of the unit and the distance-learning situation.

To enable a more direct integration of programming, visualizing and interpreting, we want to investigate the use of computational essays (diSessa, 2000; Wolfram, 2017) in future implementations. Computational essays include prose text but also "live code, […] mathematics, and pictures or diagrams in order to make an argument, explain an idea, or tell a story" (Odden & Malthe-Sorenson, 2021, p. 3). They would provide an opportunity to gradually gain knowledge through programming and interpreting results in the context of the data, while simultaneously recording the programming/knowledge process. Our vision is to let students create computational essays (diSessa, 2000; Odden & Malthe-Sorenson, 2021; Wolfram, 2017) in the form of *Jupyter Notebooks*, in which they explain their programming code and their general approach, as well as interpret and report on the results (Knuth, 1984; Lopez et al., 2020). This method would reinforce reproducibility and epistemic programming. We will evaluate the use of such computational essays by students in future cycles of this project.

Overall, this study can be seen as a "proof-of-concept" that students approximately 14 years-of-age can work with data using a professional-level programming tool, provided their environment is set up appropriately. Based on our findings, we conclude that computer science classes can be a place for introducing data science, incorporating elements of statistics and programming. For the next cycle of the project, as well as for a recommendation for other educators, using a professional tool like *Python* combined with writing computational essays by adapting and enhancing prepared *Jupyter Notebooks* might enable ninth-grade students to get in touch with real data science projects.

ACKNOWLEDGEMENTS

The lesson series was developed and discussed by the entire ProDaBi team. We thank all those involved including the teachers who taught and evaluated the project with us.

We thank the anonymous reviewers for their helpful comments and feedback on earlier versions of this article. To Tim Erickson, thank you very much for numerous discussions on concepts and for smoothing the language!

REFERENCES

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from examples: Instructional principles from the worked examples research. *Review of Educational Research, 70*(2), 181–214. https://doi.org/10/csm67w

Bargagliotti, A., Franklin, C., Arnold, P., Gould, R., Johnson, S., Perez, L., & Spangler, D. A. (2020). *Pre-K–12 guidelines for assessment and instruction in statistics education II* (GAISE II). American Statistical Association. https://www.amstat.org/asa/files/pdfs/GAISE/GAISEIIPreK-12_Full.pdf

Ben-Zvi, D. (2018). Three paradigms to develop students' statistical reasoning. In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics* (ICOTS10), Kyoto, Japan, July 8–13. International Statistical Institute. https://iase-web.org/icots/10/proceedings/pdfs/ICOTS10_2E1.pdf?1531364242

Ben-Zvi, D., Makar, K., & Garfield, J. (Eds.), (2018). *International handbook of research in statistics educatio*n. Springer.

Bereiter, C. (1980). Development in writing. In L. W. Gregg & E. R. Steinberg (Eds), *Cognitive processes in writing* (pp. 73–93). Erlbaum.

Biehler, R. (1997). Students' difficulties in practicing computer-supported data analysis: Some hypothetical generalizations from results of two exploratory studies. In J. Garfield & G. Burrill (Eds), *Role of technology in teaching and learning statistics* (pp. 169–190). International Statistical Institute.

Biehler, R., Frischemeier, D., & Podworny, S. (2015). Preservice teachers' reasoning about uncertainty in the context of randomization tests. In A. Zieffler & E. Fry (Eds), *Reasoning about uncertainty: Learning and teaching informal inferential reasoning* (pp. 129–162). Catalyst Press.

Biehler, R., & Schulte, C. (2018). Perspectives for an interdisciplinary data science curriculum at German secondary schools. In R. Biehler, L. Budde, D. Frischemeier, S. Podworny, C. Schulte & T. Wassong (Eds), *Paderborn symposium on data science education at school level 2017: The Collected extended abstracts* (pp. 2–14). Universitätsbibliothek Paderborn.

Bilgin, A. A., Newbery, G., & Petcoz, P. (2015). Engaging and motivating students with authentic statistical projects in a capstone unit. In M. A. Sorto (Ed.), *Advances in statistics education: Developments, experiences, and assessments. Proceedings of the Satellite Conference of the International Association for Statistical Education* (IASE), Rio de Janeiro, July 22–24. https://iase-web.org/documents/papers/sat2015/IASE2015%20Satellite%2028_BILGIN.pdf?1438922661

Burrill, G., & Biehler, R. (2011). Fundamental statistical ideas in the school curriculum and in training teachers. In C. Batanero, G. Burrill, & C. Reading (Eds), Teac*hing statistics in school mathematics: Challenges for teaching and teacher education. A joint ICMI/IASE study* (pp. 57–69). Springer Science+Business Media. https://doi.org/10.1007/978-94-007-1131-0_10

Caspersen, M. E. (2007). *Educating novices in the skills of programming.* [Doctoral dissertation, University of Aarhus].

Chance, B. (2002). Components of statistical thinking and implications for instruction and assessment. *Journal of Statistical Thinking 10*(3), 1–14.

Chance, B., Ben-Zvi, D., Garfield, J., & Medina, E. (2007). The role of technology in improving student learning of statistics. *Technology Innovations in Statistics Education 1*(1), 1–6. https://doi.org/10.5070/T511000026

Cobb, P., Confrey, J., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational Researcher, 32*(1), 9–13.

DiSessa, A. A. (2000). *Changing minds: Computers, learning, and literacy*. MIT Press.

Fincher, S., & Petre, M. (1998). Project-based learning practices in computer science education. In *FIE '98. Proceedings of the 28th Annual Frontiers in Education Conference: Moving from "teacher-centered" to "learner-centered" education*, Tempe, Arizona (pp. 1185–1191). https://doi.org/10.1109/FIE.1998.738607

Garfield, J., & Ben-Zvi, D. (2008). *Developing students' statistical reasoning: Connecting research and teaching practice*. Springer.

Gómez-Blancarte, A., & Ortega, A. S. (2018). Research on statistical projects: Looking for the development of statistical literacy, reasoning and thinking. In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics* (ICOTS10), Kyoto, Japan, July 8–14. International Statistical Institute. https://iase-web.org/icots/10/proceedings/pdfs/ICOTS10_1E3.pdf

Guo, P. J., & Seltzer, M. (2012). BURRITO: Wrapping your lab notebook in computational infrastructure. In *Proceedings of TaPP 12, 4th USENIX workshop on the theory and practice of provenance*. https://www.usenix.org/conference/tapp12/workshop-program/presentation/Guo

Hershkowitz, R., Schwarz, B. B., & Dreyfus, T. (2001). Abstraction in context: Epistemic actions. *Journal for Research in Mathematics Education*, *32*(2), 195–222.

Höper, L., Hüsing, S., Malatyali, H., Schulte, C., & Budde, L. (2021). Methodik für Datenprojekte im Informatikunterricht. [Methods for data projects in computer science classes] *LOG IN*, *195/196*, 37–44.

Hüsing, S. (2021). Epistemic programming: An insight-driven programming concept for data science. *21st Koli Calling: International Conference on Computing Education Research*, Article 42. https://doi.org/10.1145/3488042.3490510

Kandel, S., Heer, J., Plaisant, C., Kennedy, J., van Ham, F., Riche, N. H., Weaver, C., Lee, B., Brodbeck, D., & Buono, P. (2011). Research directions in data wrangling: Visualizations and transformations for usable and credible data. *Information Visualization*, *10*(4), 271–288.

Kay, A. (2007). Thoughts about teaching science and mathematics to young children. *Viewpoints Research Institute memo. Viewpoints Research Institute.*

Knuth, D. E. (1984). Literate programming. *The Computer Journal*, *27*(2), 97–111. https://doi.org/10.1093/comjnl/27.2.97

Krajcik, J. S., & Blumenfeld, P. C. (2006). Project-based learning. In R. K. Sawyer (Ed), *The Cambridge handbook of the learning sciences* (pp. 317–333). Cambridge University Press.

Kroes, P. & Meijers, A. (2006). The dual nature of technical artefacts. *Studies in History and Philosophy of Science, 37*, 1–4.

Lesser, L. M. (2007). Critical values and transforming data: Teaching statistics with social justice. *Journal of Statistics Education*, *15*(1), 2–22.

Lopez, M. Lisette, Wilkerson, Michelle H, & Gutiérrez, K. (2020). Contextualizing, historicizing, and re-authoring data-as-text in the middle school science classroom. *The Interdisciplinarity of the Learning Sciences, 14th International Conference of the Learning Sciences* (ICLS), Nashville, Tennessee. https://par.nsf.gov/biblio/10209452

Makar, K., & Fielding-Wells, F. (2011). Teaching teachers to teach statistical investigations. In C. Batanero, G. Burrill, & C. Reading (Eds.). *Teaching statistics in school mathematics: Challenges for teaching and teacher education. A Joint ICMI/IASE Study, the 18th ICMI Study* (pp. 347–358). Springer.

McNamara, A. (2015). *Bridging the gap between tools for learning and for doing statistics.* [Doctoral dissertation, University of California].

McNamara, A. (2019). Key attributes of a modern statistical computing tool. *The American Statistician*, *73*(4), 375–384.

Mayring, P. (2015). Qualitative content analysis: Theoretical background and procedures. In A. Bikner-Ahsbahs, C. Knipping, & N. Presmeg (Eds.), *Approaches to qualitative research in mathematics education* (pp. 365–380). Springer.

Nagpal, A., & Gabrani, G. (2019). Python for data analytics, scientific and technical applications. *In Proceedings of the 2019 Amity International Conference on Artificial Intelligence* (AICAI) (pp. 140–145). Dubai, United Arab Emirates. https://ieeexplore.ieee.org/document/8701341/

Odden, T. O., & Malthe-Sørenssen, A. (2021). Using computational essays to scaffold professional physics practice. *European Journal of Physics*, *42*(1), 1–22.

Pattis, R. E. (1990). A philosophy and example of CS-1 programming projects. *ACM SIGCSE Bulletin*, *22*(1), 34–39.

Perez, F., & Granger, B. E. (2015). *Project Jupyter: Computational narratives as the engine of collaborative data science*. https://documents.pub/reader/full/project-jupyter-computational-narratives-as-the-engine-of-

Pfannkuch, M. (2007). Year 11 Students' informal inferential reasoning: A case study about the interpretation of box plots. *International Electronic Journal of Mathematics Education*, *2*(3), 149–167.

Ridgway, J., Ridgway, R., & Nicholson, J. (2018). Data science for all: A stroll in the foothills. In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics* (ICOTS10), Kyoto, Japan, July 8–13. ISI. https://iase-web.org/icots/10/proceedings/pdfs/ICOTS10_3A1.pdf?1531364253

Rubin, A., & Mokros, J. (2018). Data clubs for middle school youth: Engaging young people in data science. In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics* (ICOTS10), Kyoto, Japan, July 8–13. ISI. https://iase-web.org/icots/10/proceedings/pdfs/ICOTS10_9B2.pdf?1531364299

Rule, A., Tabard, A., & Hollan, J. D. (2018). Exploration and explanation in computational notebooks. *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 1–12.

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig, E. (2013). Ten simple rules for reproducible computational research. *PLoS Computational Biology*, *9*(10), 1–4.

Schulte, C. (2013). Reflections on the Role of Programming in Primary and Secondary Computing Education. *Proceedings of the 8th Workshop in Primary and Secondary Computing Education*, 17–24. https://doi.org/10.1145/2532748.2532754

Schulte, C., & Budde, L. (2018). A framework for computing education: Hybrid interaction system: The need for a bigger picture in computing education. *18th Koli Calling: International Conference on Computing Education Research*, Article 12. https://doi.org/10.1145/3279720.3279733

Shearer, C. (2000). The CRISP-DM model: The new blueprint for data mining. *Journal of Data Warehousing*, *5*(4), 13–22.

Strohmaier, A., Vogel, F., & Reiss, K. M. (2018). Collaborative epistemic writing and writing-to-learn in mathematics: Can it foster mathematical argumentation competence? *RISTAL*: *Research in Subject-matter Teaching and Learning*, *1*(1), 135–149. https://doi.org/10.23770/rt1817

Tedre, M., & Apiola, M. (2013). Three computing traditions in school computing education. In D. M. Kadijevich, C. Angeli, & C. Schulte (Eds.), *Improving computer science education* (pp. 100–116). Routledge.

Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From computational thinking to computational action. *Communications of the ACM*, *62*(3), 34–36.

Verhoeven, P. S. (2013). Engaging students in statistics education: Situated learning in statistics projects. *Proceedings of the 59th ISI World Statistics Congress*, Hong Kong.

Wild, C. J., & Pfannkuch, M. (1999). Statistical thinking in empirical enquiry. *International Statistical Review*, *67*(3), 223–248.

Wing, J. (2011). Research notebook: Computational thinking: What and why. *The link magazine*. http://people.cs.vt.edu/~kafura/CS6604/Papers/CT-What-And-Why.pdf

Winkelnkemper, F. (2018). *Responsive positioning: A user interface technique based on structured space* [Doctoral dissertation, Paderborn University].

Wolfram, S. (2017). What is a computational essay? *Stephan Wolfram Writings*. https://writings.stephenwolfram.com/2017/11/what-is-a-computational-essay/

SUSANNE PODWORNY
Warburger Str. 100,
33098 Paderborn, Germany