COMPUTATIONAL ESSAYS AS AN APPROACH FOR REPRODUCIBLE DATA ANALYSIS IN LOWER SECONDARY SCHOOL

Sven Hüsing and Susanne Podworny
Paderborn University
sven.huesing@uni-paderborn.de

Data Science has become an increasingly important aspect of our everyday lives as we gain many different insights from data analyses, for example in the context of environmental issues. To make the process of data analyses comprehensible for lower secondary school students, we developed a data analysis project for computer science classes, focusing on gaining insights from environmental data by using the didactical concept of epistemic programming. In this article, we report on the second implementation of this project, which was conducted in a ninth-grade computer science class. Concretely, we examine, how far the students were able to create computational essays to conduct reproducible data analyses on their own. In this regard, the computational essays created with the help of the professional tool Jupyter Notebooks were examined regarding aspects of reproducibility.

INTRODUCTION

Nowadays, data science is encountered by us every day and should therefore be a part of any education (Ridgway, 2016). For example, many people track their jogging activities to gain new insights into their running performance. Also, people evaluate environmental data like temperature to reason about global warming, consider variables to analyze natural disasters or measure the CO_2-content to decide when to open a window in the classroom. Data like this is always collected, visualized, and analyzed with a specific interest in gaining insights about a certain topic. When doing data analysis and visualization using tools like Excel or CODAP, there are certain features available for performing the data analysis. For example, one can compute specific statistics, create diagrams, or compute evaluation parameters and apply predefined filters on the data. Another option is to do data analysis by programming. With programming, there is more freedom, as specific aspects or routines that are not available in spreadsheet tools like Excel or CODAP can be implemented. Programming allows great flexibility with a huge number of libraries available and the option to revise each part of the data analysis in hindsight through the loop-like interleaving of exploration and reflection of the programming results and performing the changes to the program code that become evident through this. Additionally, pre-written code for a concrete data analysis can be re-used to perform the same evaluations for other data.

For conducting a data science project in school, we created a teaching unit in which students can carry out their own data analysis as flexibly as possible through programming. In particular, programming allows a high reproducibility of the results and at the same time allows documenting the process of data analyses as implied by e.g. diSessa (2000) and Wolfram (2017). As a programming environment, we used Jupyter Notebooks (Perez & Granger, 2015), which are interactive documents with different kinds of cells. The cells can contain live code, the respective output of a code-cell, or additional text. A Jupyter Notebook can be prepared and distributed via a server so that it can be used from any (mobile) device. Prepared Jupyter Notebooks with prewritten code-cells are especially suitable for use in schools as they provide a programming setting even for inexperienced programmers.

In this article, we examine, in how far students can develop computational essays by adapting prepared Jupyter Notebooks in order to create a reproducible data analysis.

THEORETICAL BACKGROUND

For our newly developed approach, we build on the concept of reproducibility in order to describe a new way of gaining insights in the context of data analysis. We use a didactical programming approach focusing on gaining new insights and on the persistent programming process as a “product”. We call this didactical approach *epistemic programming*.

Epistemic Programming

The term "epistemic programming" describes a programming approach that focuses on acquiring new knowledge within epistemic actions. These are mental actions with the goal of using or constructing knowledge (Hershkowitz et al., 2001). Regarding epistemic programming, these actions include, for example, reading in data or creating a graph that represents that data. Within the didactical concept of epistemic programming, students gain insights through their programming process by reflecting the programming results and adapting the program based on this. The students go through a kind of "tinkering process" in which they gradually expand or improve their program or their data analysis regarding the aspired insights. Hence, instead of focusing on the creation of a program running correctly as in an 'engineering' programming approach (Tedre & Apiola, 2013), the focus within epistemic programming is on gaining new insights within the programming process and from the reflection of the results. Programming here represents a tool, that enables learners to gain insights on their own, instead of adopting findings by reading or listening, when there is no direct relation to the knowledge-creation-process (Kay, 2007).

In the context of epistemic programming, data analysis in school offers the opportunity for students to ask their own questions and answer them by collecting and evaluating data in the respective domain like proposed in the PPDAC cycle (Wild & Pfannkuch, 1999). Additionally, publishing results in an appropriate way is important (McNamara, 2019). This relates to the concept of reproducibility.

Reproducible data analysis

There are several suggestions to make data analyses reproducible in order to give other people the possibility of rerunning it with the same or similar data, to get comparable results (Kandel et al., 2011; McNamara, 2019). According to McNamara (2019) and Biehler (1997), interactive data analysis tools like Excel prevent reproducibility, as the individual steps of the data analysis are not saved. Sandve et al. (2013) suggest connecting textual statements directly to the results. For reproducible data analyses, this means connecting the process and the results with the respective documentation.

Carver et al. (2016) point out, that reproducible research has not only gotten important in a scientific context but can also be brought to school. The associated idea is to let students report on the results of their data analyses and to explain the program code from which the output was built in order to make it understandable. In this sense, programming with Python represents a suitable tool for reproducible data analysis since it shows high readability in contrast to many scripting languages by containing fewer keywords, a simple structure, and a clear syntax (Nagpal & Gabrani, 2019).

Computational Essays as a tool to carry out reproducible data analysis

Considering the interdisciplinary character of epistemic programming as well as the benefit of reproducible data analyses there is a need for a programming environment that unites insights in the real-world context with the code in one place.

One approach for such a programming environment is the idea of computational essays (Odden & Malthe-Sørenssen, 2021; diSessa, 2000; Wolfram, 2017). Odden & Malthe-Sørenssen (2021) state that 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". Viewers are thus given the opportunity to explore the programming results, its interpretations, and the process itself through interacting with the computational essay.

From this perspective, the process of writing computational essays corresponds to the idea of reproducible research as well as epistemic programming with regard to acquiring new insights through programming.

In this paper, we evaluate how lower secondary school students can use computational essays as a method for conducting their own data analysis in the sense of reproducibility and epistemic programming. In order to answer this research question, we developed a data analysis project for lower secondary school and conducted it in a Design-Based-Research approach (Cobb et al., 2003).

THE TEACHING UNIT

We identified computer science classes as an appropriate place for data science projects as they provide time for larger projects. Additionally, students have already basic programming knowledge. A benefit of using Jupyter Notebooks from a didactical point of view is that it is possible to prepare them by pre-writing parts of code, tasks, or explanations that enable students to carry out their data analysis like in worked examples (Atkinson et al., 2000).

The teaching unit consists of twelve lessons of 45 minutes each. Students independently conduct projects in groups of 2-3 by developing a research question in terms of environmental variables such as particulate matter, humidity, or temperature, followed by collecting and analyzing local data in order to answer their own research question. In doing so, students get in touch with methods and processes from the 'real science' (Kay, 2007). By conducting the data science project, the students gain insights regarding their environment, expand their programming skills in a real data-driven project and learn to document the process and the results in a computational essay.

The data collection is carried out with several sensor boxes (https://sensebox.de/en/) - Arduinos with different measurement sensors. While the sensor boxes collect the data for a longer period, the students complete an online Python programming course in order to be able to execute the data analysis later on. After the data collection is completed, the students start with the data analysis by reading in the data into the prepared Jupyter Notebooks. These prepared Jupyter Notebooks are developed as worked examples (Atkinson et al., 2000) for a data analysis on temperature data. The students then have to adapt the pre-written program code and expand it regarding further visualizations or statistical values. Finally, they present their findings in a presentation of the computational essays created in the Jupyter Notebooks.

Conducting the teaching unit

This paper reports on the second cycle of the teaching unit that was conducted in August/September 2020 in a 9th-grade computer science class with 23 students, aged 13-15. The students had little prior knowledge in programming with Scratch (a visual, block-based programming language) and no prior knowledge in text-based programming like Python. Regarding statistical knowledge, the students knew about absolute and relative frequencies, bar charts, boxplots, and measures like mean, median, min, and max but had little more experience in the statistical area. The students collected six data sets from August and September 2020 - two data sets each of temperature, humidity, and particulate matter data, with one data set recorded on a busy street and another data set recorded in a park. Because of a problem with the respective power supply, the second box only collected temperature and humidity data.

Due to the COVID-19-situation, the project had to be shortened regarding the presentations of the students because of a distance-learning situation during the teaching unit. Therefore, the students conducted the data analysis within the Jupyter Notebooks but did the interpretation separately in Word-documents since they were used to this kind of interpretation in contrast to the interpretation process within Jupyter Notebooks. Hence, the computational essays were distributed among the Jupyter Notebooks and the Word-documents.

METHODOLOGY

We analyzed all of the student-groups' Jupyter Notebooks (n=12) and Word-documents (n=12) regarding the way the students created computational essays by adapting the Jupyter Notebooks and interpreting their results in the Word-documents. We used the method of qualitative content analysis (Mayring, 2015) to categorize the adapted code within the Jupyter Notebooks and for the written reports in the Word-documents by a deductive category system, developed in advance. Here, we considered aspects like 'variables used', 'visualizations created' or 'cells adapted/used' concerning the Jupyter Notebooks and 'visualizations included', 'number of visualizations used' or 'connection between visualizations and interpretations' regarding the Word-documents.

Please note, that distributing the data analysis between Jupyter Notebooks and Word-documents does not correspond to the actual idea of computational essays, since the goal is to merge the programming process and the interpretations and explanations as closely as possible in one document (diSessa, 2000; Wolfram, 2017). However, conducting the data analysis in the Jupyter Notebooks and explaining/interpreting it in Word-documents still gives opportunities regarding computational essays by connecting the code, results, and interpretations because the viewer can read the interpretations within the Word-document before interacting with the visualizations and the tools in the Jupyter Notebook like in "complete" computational essays (Odden & Malthe-Sørenssen, 2021; diSessa, 2000).

RESULTS AND DISCUSSION

Analysis of the Jupyter Notebooks

The analysis of the Jupyter Notebooks shows that the students explored at least one data set by creating visualizations in their Jupyter Notebooks. While there were different levels of complexity regarding different types and adaptions of visualizations, all student groups created a time series representation of at least one variable after filtering a single day from the data set and customized the respective graphics by giving individual x-labels and y-labels as well as a title (see Figure 1 as an example). This kind of visualization enabled the students to identify first rough findings regarding patterns of the respective variable. However, only one group created a time series visualization in which multiple data sets were visualized (like in Figure 2). Obviously, this visualization was shared with other groups, because the corresponding code was found only in one Jupyter Notebook while the visualization was found in eight Word-documents.

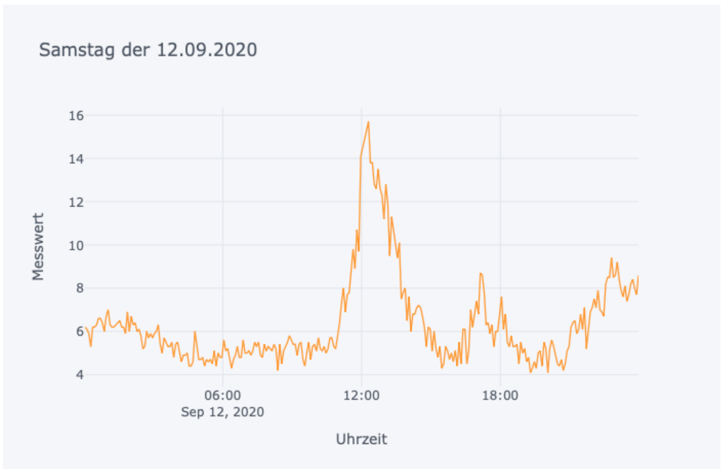

Figure 1. Students' visualization of the particulate matter level for September 12, 2020, created within a Jupyter Notebook

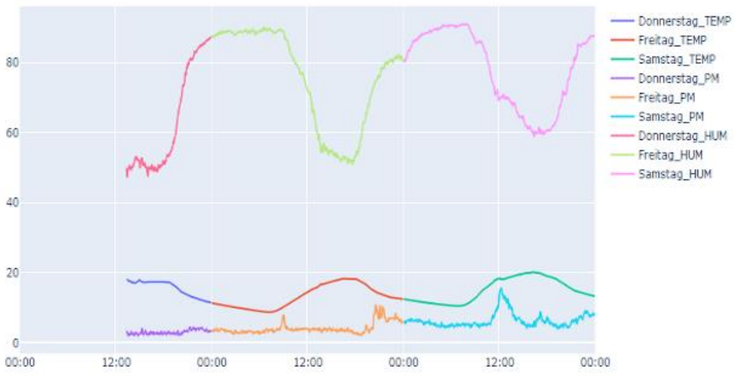

Figure 2. Simultaneous visualization of the particulate matter, humidity and temperature data for September 10, 2020 until September 12, 2020, created within a Jupyter Notebook

Concluding, all student groups were able to develop suitable visualizations by adapting the prepared Jupyter Notebooks. Differences in number of analyzed data sets, periods, x- and y-axis- and title labeling, and number of visualizations were found. Together with the code cells that are prior to the visualizations, these Jupyter Notebooks fit to an aspect of computational essays - bringing together code and the respective results (Odden & Malthe-Sørenssen, 2021; diSessa, 2000), in order to enable the viewer to understand the process of creating these visualizations.

Another remarkable aspect regarding reproducibility is that four groups used the potential of the Jupyter Notebooks regarding the repeated execution with similar data sets as Kandel et al. (2011) and McNamara (2019) mention: At least four of the student groups re-ran parts of the Jupyter Notebooks with different data sets to compare the same variable but consider the different locations of the data or to analyze relations between the different variables. This could be interpreted from analyzing the execution numbers of the single Jupyter Notebook code cells. In these four Jupyter Notebooks, some code cells producing visualizations had higher execution numbers than the cells beneath. This shows that previous cells were executed again after the following code cells, so we interpret that the students repeatedly executed them with different data sets.

Analysis of the reports

All but one student group used correct time-series visualizations in their Word-documents and commented on the respective variables either on a describing level or by also interpreting the results. Eight groups created comprehensible interpretations for the visualizations. For example, they described findings in terms of comparing values in their city with average values across the country. Additionally, many (n=10) student groups reported on the relation between two or more variables: Eight student groups discussed the relation of time and particulate matter. Among many other relations, five student groups discussed the connection between humidity and particulate matter. One student group apparently also used external resources to find an explanation for the relation: “When the air humidity increases, larger ‘particle balls’ are formed from the particles. Thus, there are more particulate matter particles in the same place and the particulate matter levels increase. Since the particles are very small and very well distributed outside, it takes some time for the accumulation of many particles to form, which is why sometimes the values rise with a slight offset. But in general, it is easy to see that the values behave analogously.” The students reported and discussed relations that were always accompanied by the corresponding visualizations. This shows that the Word-documents were useful to connect the programming results with the respective interpretations in the sense of reproducibility, as described by Carver et al. (2016). Thus, the creation of the Word-documents supported the students in reporting on their findings, gained from the programming results. Because of the time pressure due to another imminent homeschooling situation no deeper analyses were guided in class. Nevertheless, the students recorded the first findings gained in the context of an epistemic programming process and reproducible data analysis.

CONCLUSION AND OUTLOOK

In this paper, we reported on a data science project for lower secondary school. The results show that the students coped well with the project and were able to gain insights in form of certain environmental variables like temperature, humidity, and particulate matter as well as their relations, based on the didactical approach of epistemic programming. The computational essays were distributed among the Jupyter Notebooks and the Word-documents. The Jupyter Notebooks supported the students in the sense of reproducibility as the programming results like data visualizations were connected to the respective Python code (diSessa, 2000). The combination of Jupyter Notebooks and Word-documents made the data analysis more understandable and comprehensible for the reader, regarding the results and the process itself. In this respect, the computational essays implemented here represent an approach for reproducible data analysis in the sense of Carver et al. (2016) and McNamara (2019) and enable students to perform their own data analyses in prepared Jupyter Notebooks (Atkinson et al., 2000). In addition, some student groups used the potential of Jupyter Notebooks in terms of creating the same visualizations and evaluations for different data sets by re-running the program code with different data sets (McNamara, 2019).

In general, the use of Jupyter Notebooks enabled the recording of the single data analysis steps and they therefore represented a tool for reproducible data analysis. However, due to the separation of the program code and the interpretations, the students could not bring together the interpretations, the results, and these data analysis steps in one (digital) document.

In future iterations of the teaching unit, we want to further elaborate, to what extent computational essays implemented exclusively through Jupyter Notebooks can contribute to a more reproducible (McNamara, 2019) and comprehensible data analysis. By bringing together the process of evaluating data and interpreting the results, a stronger focus on analytical skills will be possible, as the students statistically interpret the results obtained in the Jupyter Notebook and then - if necessary - directly make changes to the code in a kind of "tinkering" and epistemic programming process.

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.

Biehler, R. (1997). Software for Learning and for Doing Statistics. *International Statistical Review, 65*(2), 167–189.

Carver, R., Everson, M., Gabrosek, J., Horton, N., Lock, R., Mocko, M., Rossman, A., et al. (2016). Guidelines for Assessment and Instruction in Statistics Education (GAISE) *College Report* 2016, 143.

Cobb, P., Confrey, J., diSessa, A., 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*. Cambridge, Mass: MIT Press.

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

Kandel, S., Heer, J., Plaisant, C., Kennedy, J., van Ham, F., Riche, N. H., Weaver, C., et al. (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. Glendale, CA.

Mayring, P. (2015). *Qualitative Inhaltsanalyse: Grundlagen und Techniken* (12., überarbeitete Auflage.). Weinheim Basel: Beltz Verlag.

McNamara, A. (2019). Key Attributes of a Modern Statistical Computing Tool. *The American Statistician, 73*(4), 375–384.

Nagpal, A., & Gabrani, G. (2019). *Python for Data Analytics, Scientific and Technical Applications*. 2019 Amity International Conference on Artificial Intelligence (AICAI) (pp. 140–145). Presented at the 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates: IEEE. Retrieved May 29, 2021, from 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).

Perez, F., & Granger, B. E. (2015). *Project Jupyter: Computational narratives as the engine of collaborative data science*. Retrieved September, 11(207), 108.

Ridgway, J. (2016). Implications of the Data Revolution for Statistics Education. *International Statistical Review, 84*(3), 528–549.

Sandve GK, Nekrutenko A, Taylor J, Hovig, E. (2013). Ten Simple Rules for Reproducible Computational Research. *PLoS Comput Biol 9*(10): e1003285.

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.* New York, NY and London: Routledge.

Wild, C. J., & Pfannkuch, M. (1999). Statistical Thinking in Empirical Enquiry. *International Statistical Review, 67*(3), 223–248.

Wolfram, S. (2017, November 14). *What Is a Computational Essay?* [Blog Post]. Retrieved May 31, 2021 from https://writings.stephenwolfram.com/2017/11/what-is-a-computational-essay/