Preparing K-12 Students to Meet their Data: Analyzing the Tools and Environments used in Introductory Data Science Contexts

Rotem Israel-Fishelson
University of Maryland
rotemisf@umd.edu

Peter F. Moon
University of Maryland
pmoon@umd.edu

Rachel Tabak
University of Maryland
retabak@umd.edu

David Weintrop
University of Maryland
weintrop@umd.edu

ABSTRACT

Data science education has gained momentum in recent years. Along with the development of curricula to teach data science, the number and diversity of tools for introducing data science to learners are also multiplying. The tools used to teach data science play a central role in shaping the learning experience. Therefore, it is important to carefully choose which tools to use to introduce learners to data science. This article presents a systematic review of 25 data science tools that are, or can be, used in introductory data science education for K-12 students. The identified tools list includes spreadsheets, visual analysis tools, and scripting environments. For each tool, we examine facets of its capabilities, interactions, educational support, and accessibility. This paper advances our understanding of the current state of introductory data science environments and highlights opportunities for creating new tools to better prepare learners to navigate the data-rich world surrounding them.

CCS CONCEPTS

• **Human-centered computing** → Interaction design; • **Applied computing** → Education; Interactive learning environments.

KEYWORDS

Data science education, Data analysis tools, Introductory data science

ACM Reference Format:
Rotem Israel-Fishelson, Peter F. Moon, Rachel Tabak, and David Weintrop. 2023. Preparing K-12 Students to Meet their Data: Analyzing the Tools and Environments used in Introductory Data Science Contexts. In *Learning, Design and Technology (LDT '23), June 23, 2023, Evanston, IL, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3594781.3594796

1 INTRODUCTION

The field of data science has grown immensely over the past decade, following the exponential growth of data, the improvement of computational capabilities, and the increasing demand for data-driven decision-making. The proliferation and development of data science tools have played a significant role in this growth, enabling data scientists to analyze and interpret large, complex datasets more effectively and efficiently than ever before [32]. The abilities and potential of data science tools are vast, ranging from simple data cleaning and wrangling tasks to complex machine learning algorithms that can predict user behavior and identify patterns and anomalies in large datasets [12].

In recent years, ideas and skills from data science have begun permeating the educational sector by those seeking to equip students with the skills and knowledge they need to excel in the field and better understand the world around them. To that end, various data analysis tools have been developed to support students in learning about and engaging with the data [13]. These tools have been incorporated into new curricula designed to introduce students to data science and teach them foundational concepts and computational practices for collecting, sorting, extracting, and analyzing data from different sources [11]. The various analysis and visualization tools allow students to investigate phenomena and deal with questions emerging from diverse datasets [25]. Furthermore, such tools enable students to identify patterns and trends, extract insights from data, and make informed decisions [23]. To maximize the potential of data science tools in K-12 education, it is crucial for educators to carefully select the right tools for their students, as the tools themselves play a critical role in shaping learners' experiences [21].

There are various aspects to consider when choosing the right tool to introduce data science to students. Multiple factors may be considered, including the tool's functionality, the user interface, the support materials available to teachers and students, and the learning curve. The proliferation of different tools for data analysis and the emergence of simple user interfaces for the automation of complex tasks invites students to engage in complex data science practices [8]. However, no consensus exists on the ideal tool for introducing data science in K-12 education [20]. A recent study of 330 teachers from across the US found that the most common tool is spreadsheets, while the R language was the least common. Among the barriers to using the tools were: the necessary resources (the cost of the tool and time to develop dedicated lessons) and the required learning curve [24].

This paper builds on previous research characterizing different attributes of data science tools [20, 32] but focuses specifically on tools used in K-12 educational contexts. Pimentel et al. [21] examined several spreadsheets, visual tools, and scripting tools used in K-12 data analysis across nine features, including accessibility, flexible plot creation, and extensibility. Our analysis draws on this

LDT '23, June 23, 2023, Evanston, IL, USA

ACM ISBN 979-8-4007-0736-0/23/06.
https://doi.org/10.1145/3594781.3594796

and the preceding framework [20] while adding additional analytic dimensions, including statistical and graphing capabilities. It also considers the most recent set of data science tools that coincide with the emergence of K-12 data science curricula [17]. After compiling a set of educational data science tools, we examined each tool to answer the following research questions:

- *RQ1: What capabilities does the tool possess in terms of data science practices, data visualization, statistical calculations, and extensibility?*
- *RQ2: How do users interact with the tool, and how might the interaction support learning?*
- *RQ3: What accessibility features does the tool provide?*

2 PRIOR RESEARCH

The tools offered for K-12 data science education include diverse capabilities for collecting, analyzing, and visualizing data inside and outside the school walls [8]. They range from simple spreadsheets to complex scripting languages like Python and R. These tools differ in their ease of use, learning curve, user interface, and functionality. Pimentel et al. [21] suggested dividing the tools into four broad categories: spreadsheets; visual interfaces; scripting languages; and other tools.

Spreadsheet tools, such as Microsoft Excel and Google Sheets, are commonly used in data science education because they are ubiquitous, easy to use, and freely available for educational purposes. Spreadsheets enable the collection of data from various sources. They can be used to clean, organize, filter, manipulate, and prepare data for analysis. Moreover, spreadsheets support data exploration and analysis. They can be used to perform both basic and advanced statistical analyses. Furthermore, spreadsheets offer visualization capabilities by creating charts, graphs, and other visualizations. These capabilities can help students understand statistical concepts, identify trends and patterns, and interpret the data [1, 16]. Nevertheless, spreadsheets are limited in their ability to perform sophisticated analysis, and thus, it is often required to load them into other data science tools [10].

Visualization tools provide graphical user interfaces for building visual representations of data. Their interfaces often include menus or drag-and-drop features to ease the analysis and understanding of complex data. In addition, these tools provide functionality for arranging quantitative and qualitative data in tables, graphs, and other graphic depictions so that users can discover trends in a dataset without knowing how to program [8]. The interactive, friendly user interface also supports transforming data representations and performing exploratory analyses from various perspectives [21]. Under this category are dedicated tools developed for educational purposes, such as CODAP [6], Tuva [9], and DataClassroom [7]. These tools constitute a scaffold for learning data science practices, and their learning curve is small. However, they have limited analytical capabilities and the ability to manipulate raw data. There are also professional visualization tools, such as Tableau [27] and Power BI [22] which offer advanced capabilities and enable the creation of dashboards and interactive reports [2].

Scripting languages include R, Python, and Pyret, and they are used to perform data science activities and manipulations, statistical analysis, and machine learning. These languages offer the widest functionality but often have a steep learning curve [12]. These languages are used in educational settings despite being more complicated than other analytic tools because they enable the automation of advanced functions on large datasets [14, 21]. Python is a widely used general-purpose programming language that has a large community of users and various libraries and modules that make working with data easy. Similarly, R is another popular data science scripting language designed for statistical analysis and data visualization. Pyret is a functional-programming language with Python-like syntax designed for educational purposes, featuring color-coded error messages written in student-friendly language.

Other tools include environments that support scripting along with scaffolded workbooks. These environments offer built-in interactive exercises for performing step-by-step operations of changing existing code or writing code incrementally [21]. Tools in this category include One such tool is Jupyter Notebook, an open-source web application that allows users to run chunks of Python code and immediately display output and text notes in the same document [5]. Google Colab [3] is another tool that falls under this category.

3 METHODS

To answer the stated research question, we conducted a systematic review to identify the current state of the tools and environment that are or can be used in introductory K-12 data science education. Our first step in assembling the list of tools was to review tools presented and discussed in the academic literature. The analysis began by searching the keywords "Data Science Education Tools" in the Association for Computing Machinery (ACM) and Institute of Electrical and Electronics Engineers (IEEE) digital libraries. Next, we examined the articles' title, abstract, and methodology to identify whether a data science tool was included as part of the research. This strategy helped us formulate an initial list of tools. After that, we added to the list additional tools recommended by colleagues not identified in the systematic search. After compiling a list of potential data science education tools, we create inclusion and exclusion criteria for selecting the relevant tools for analysis. The inclusion criteria were: (1) must be a tool for learning / focused on education; (2) must be focused on data science concepts or practices; (3) must be available for analysis (i.e., can be run and used). Exclusion criteria were: (1) must not be focused on AI or machine learning; (2) must not be a library (e.g., pandas) or general-purpose programming language (e.g., R, Python). A spreadsheet was used to record and organize the results of the analysis.

We identified five distinct analytic categories for each of our three research questions to evaluate each tool and answer our stated questions. These categories are informed by prior work in this space [20, 21]. Table 1 outlines the 15 analytic categories organized by guiding research questions:

4 FINDINGS

Our analysis yielded 25 tools that can or are being used in K-12 data science education (Table 2).

Table 1: Distribution of the Tools by the Creation Method of the Plots

RQ	Criteria	Description
RQ1 Capabilities	Data Manipulation	The processes for organizing and extracting data, and how they are carried out.
	Statistical Capabilities	Can the tools be used to conduct correlation analysis and linear regression?
	Data Visualization	Support for tabular and graphic displays of the data, the type of graph creations supported by the tool, and their creation method.
	Dataset Availability	The existence of built-in datasets and capabilities to import data (as a file or via API calls from external services) and export the data.
	Extensibility	Open-source usage and flexibility to build extensions.
RQ2 Interactions & Education	Runtime Environment	The environment in which the tool is executed, (web browser, or operating system)
	Programming Modality	The representative infrastructure used and the variety of interactions that the interface enables.
	Scripting Language	The programming language is supported by the tool.
	Instructional Materials	The tool includes resources aimed to assist educators and students in their teaching and learning processes, including written guides, tutorials, lesson plans, etc.
	Activities/ Assignments	The ability for instructors to create and/or assign activities for their students within the tool.
RQ3 Accessibility	Screen Reader	Is the environment compatible with screen reader software?
	Color Palette	Does the tool consider color blindness or offer accessible color palettes?
	Simplified Representations	Does the tool provide the ability to simplify the provided graphical data representations?
	Customized Interactions	Can the user customize interactive features to support learners with limited mobility or fine motor skills?
	Font size Modification	Can the user customize the font size to support those with limited sight?

4.1 RQ1: K-12 Data Science Education Tool Capabilities

To answer our first research question, which focused on the capabilities of the tools, we identified five distinct analytic categories. A complete listing of each tool and associated capabilities can be found in Appendix1, Table A.1.

4.1.1 Data Manipulation. To begin our analysis of K-12 data science education tool capabilities, we investigated the data manipulation capabilities built into each tool. This is important as the learner's ability to transform the data is central to the meaningful interpretation of that data. Twenty-one of the 25 tools we reviewed featured some sort of data manipulation capability. Our analysis of the tools revealed that the data manipulation capabilities featured most prominently were filtering, deleting, sorting, and aggregating. Our analysis highlights the prevalence of each of these, with consideration of how data manipulation occurred. Only one tool (of the 21 that included data manipulation capabilities) did not offer to filter the data. Sorting (available in 14 tools), deleting (available in 15 tools), and aggregating (available in 12 tools) were less prevalent. Ten tools of the 25 tools we reviewed included all four data manipulation capabilities. Of these ten tools, four of these tools utilized a graphical user interface (GUI) for data manipulation (CODAP, DataClassroom, DbSnap, MakeCode, and EduBlocks), while five utilized code-based data manipulation. One of these ten tools (Kaggle) used both GUI and code for data manipulation.

4.1.2 Statistical Capabilities. We also were interested in whether each tool offered statistical capabilities. We evaluated if the varying tools could conduct correlation analysis and linear regression, given that these are the two most commonly used techniques for investigating quantitative relationships. Fourteen of the tools we reviewed offered both correlation and regression analysis. Each of the ten tools that had the full range of data manipulation capabilities also supported correlation and regression analysis. There were four tools that had limited data manipulation capabilities, yet still offered correlation and regression analysis: Pyret, iNZight, TinkerPlots, and Tuva.

4.1.3 Data Visualization. As part of our analysis, we examined which data visualizations the tools support. Specifically, we focused on tabular displays, the types of graphs supported, and how users create them. Most of the tools (16 out of 25) allow the presentation of the data in tabular format. All but one (BlocklySQL) allow visualizing the data in a graph. We found a uniform distribution of the tools according to their creation method. Seven tools enable the creation of graphs by writing a specific code, and seven by using their inherent blocks (Figure 1, left). We also found three tools, namely BlockPy, Edublocks, and mBlock, which support both methods and allow users to seamlessly alternate between the two creation methods. Additionally, we found that seven tools enable the creation of graphs using their graphical user interface (GUI). In these tools, the user can create a graph by dragging and dropping attributes into the axes of a predefined visualization (Figure 1, right), often in the form of a scatterplot, as found in 22 tools. Other types of graphs that were supported by the tools are line graphs (in 16 tools), bar charts (12), pie charts (11), and histograms (9). Also, a few tools allowed the creation of a radar graph, violin, heatmap, and box plot.

Table 2: List of the 25 tools that were identified and analyzed.

Tools	Brief Description
Blockly	Blockly is an open-source block-based programming library.
BlocklySQL	BlocklySQL is a block-based editor for SQL.
BlockPy	BlockPy is a web-based, block-based Python environment designed for data science.
Bridges CS	Bridges CS is an easy-to-use interface with a set of classes (C++ and Java are supported) that serve as building blocks to the common data structures (lists, tree structures, graphs) used in computer science.
CODAP	CODAP is a free educational software for data analysis.
DataClassroom	DataClassroom is a web-based data analysis and visualization application.
Datacommons	Datacommons provides a visualization tool to explore aggregated data from different sources.
DBSnap	DBSnap is a web-based application to build database queries with block-based tools.
EduBlocks	EduBlocks is a visual block-based programming tool that helps teachers to introduce text-based programming languages, like Python & HTML, via a drag-and-drop programming experience.
GapMinder	Gapminder identifies systematic misconceptions about important global trends and proportions and uses reliable data to develop easy-to-understand teaching materials to rid people of their misconceptions.
Google Colab	Google Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis, and education.
GP	GP is a general-purpose block-based language that enables generating high-quality graphics, manipulating images and sounds, and analyzing text files or CSV datasets.
iNZight	iNZight is a free, easy to use software for statistical data analysis.
iSnap	iSnap is a data-driven, intelligent tutoring system for block-based programming, offering support such as hints, feedback and curated examples.
Jupyter Notebook	Jupyter Notebook is a "code notebook" that allows users to run chunks of Python code, along with displayed output and text notes all in the same document.
Kaggle	Kaggle is an online community platform for data scientists and machine learning. It allows users to collaborate with other users, find and publish datasets, and use GPU integrated notebooks.
MakeCode Data Science Editor	An experimental block-based editor to teach data science to middle school students.
mBlock	mBlock 5 is a graphical programming platform tailored to coding education. It allows users to switch between block-based interface and programming interfaced based on Python.
NetsBlox	NetsBlox is a cloud-based block-based tool that enables the creation of networked programs.
Pyret	Pyret is a programming language and environment designed for teaching computer science to beginners.
Quorum	An evidence-oriented programming language tool, that support blind or visually impaired students.
Scratch Data Blocks	Scratch Data Blocks is a Scratch-based toolkit that allows Scratch users to program with public data from the Scratch online community.
Snap!	Snap! is a block-based programming language and platform for creating programs.
TinkerPlots	TinkerPlots is a data visualization and modeling tool.
Tuva	Tuva is a platform that enables students to easily explore, manipulate, and analyze data.

Figure 1: Creation of graph using NetsBlox's inherent blocks (https://netsblox.org/) (left) and creation of graph using Tuva's graphical user interface (https://tuvalabs.com/) (right)

4.1.4 Data Availability. Since data science is fundamentally about data, it was important to consider how the data is accessed in the various tools. Our analysis shows that over half of the tools (14 out of 25) include built-in datasets which enable students to explore the provided data while learning and practicing data science techniques. Some tools offer a limited number of built-in datasets, while others include a wide variety. BlockPy, for example, provides 60 built-in datasets which cover topics related to entertainment (i.e., data about Broadway shows and video games), environment (i.e., wind turbines, earthquakes, global emissions), health (i.e., COVID-19, hospitals), education (i.e., school scores), and politics (i.e., state demographics, elections). Of the reviewed tools, 22 allow users to import datasets, and 22 tools enable exporting all or part of the datasets. Another practice for gathering data is application programming interface (API) calls. API calls enable data retrieval from web services or external databases. Our analysis revealed that six tools support API calls to access data from external services. NetBlox, for example, allows users to query and retrieve data from different services on topics including maps, weather, earthquakes, movies, games, museums, social media, and more. The data from these services are accessed using an API call integrated into the user interface in the form of dedicated blocks with dropdown lists of the services and arguments [4].

4.1.5 Extensibility. Extensibility is an essential aspect of data science tools because it allows the flexibility to adapt to evolving computational methods and analysis requirements of different users. Moreover, the ability to build extensions based on existing modules prevents them from becoming obsolete [20]. Therefore, we examined whether the tools we analyzed support the ability to develop extensions and whether they have an open-source policy that allows customization of the tool. We found ten tools that are open-source and allow the creation of tool extensions. Jupyter Notebook, for example, has several extensions that allow users to customize their notebooks and add additional functionality, such as table of contents and variable inspector. Blockly enables users to enhance its functionality by creating custom blocks using the Blockly extension builder without writing code or developing new blocks using JavaScript. A notable advantage of the open-source policy is that the changes to the code become public domain at no cost [32]. The developed extensions will sometimes be published or even embedded as an integral part of the tool. In CODAP, for example, a Plugins section was added to present various extensions developed by users. One such extension is called "transformers," which enable transforming datasets to produce new, distinct output or values instead of modifying the original input dataset. The Transformers plugin allows users to compare variables, perform advanced filtering and aggregation, and run statistical analyses [28].

4.2 RQ2: K-12 Data Science Tool Supported Interactions and Educational Features

To answer our second research question, we first examined how learners engage with the tools and which educational features and supports they provide. A complete listing of each tool's interactions and educational features can be found in Appendix2, Table A.2.

4.2.1 Runtime Environment. Our analysis revealed that most tools (23 out of 25) are operated through the browser. Of these, four tools, Quorum, GapMinder, GP, and Jupyter Notebook, also enable local installation of their environment using the Windows or Mac operating systems. Two additional tools, Bridges CS and TinkerPlots, can only be accessed by installing them on Windows or Mac. mBlock provides the broadest access among the tools as it can be run on a browser, Windows, Mac, Linux, or Android operating systems. The runtime environment is an essential parameter because there is no uniformity among the operating systems students use. Additionally, selecting a tool that runs on a browser requires the device to be connected to the internet. In contrast, local installation requires technological literacy from the student and appropriate support from the administrative shell.

4.2.2 Programming Modality. The data science tools studied here differ in their interface design. Seven tools offer a user interface that allows manipulating the data and its visualization by selecting from menus or drag-and-drop components. The rest of the tools offer block-based, text-based, or hybrid interfaces, which we refer to as programming modality. Programming modality encompasses the representational infrastructure used and the range of interactions facilitated by the interface [31]. Block-based interfaces involve dragging and dropping graphical blocks and binding them together to form a script. Such modality may limit the ability to write complex programs, but, on the other hand, it is visually intuitive and helps prevent syntax errors. We found six tools that offer a block-based interface. An example of such an interface can be found in Snap!, where users can interact with the program handling the data in a block-based form (Figure 2).

Text-based interfaces involve writing code using a text editor. This modality provides greater flexibility, more concise expressiveness, and often more fine-grained control over the program's behavior. However, it can be difficult for novices to learn due to the syntax and structure required to write the code. We found seven tools that fall under this category. In Google Colab and Kaggle, for example, users can assemble their code in either Python or R languages, while in Bridge CS, they can choose between C++, PHP, and Python (Figure 3).

Hybrid interfaces combine elements of both block-based and text-based interfaces. This modality can provide a smoother learning curve for novices and help bridge the gap between visual and traditional text-based programming [30]. Like Lin and Weintrop [18], we further break these tools into sub-categories based on the interaction and presentation of the modalities. Read-only one-way transition, like Blockly, DBSnap, and mBlock, enables viewing the text-based version of the block-based program. Editable one-way transition tools, like EduBlcoks, allows viewing the blocks' textual version and even editing it but do not support text-based edits being converted back to blocks. Dual-modality tools, such as BlockPy and BlockSQL, enable switching between the block and the text-based representations where a change in one modality affects the other.

4.2.3 Scripting Language. We examined which scripting/programming languages are used by the tools included in this study. Our analysis shows that the most common scripting language is Python, used by Blockly, BlockPy, Bridges CS, EduBlocks, Google Colab, Jupyter Notebook, Kaggle, and mBlock.

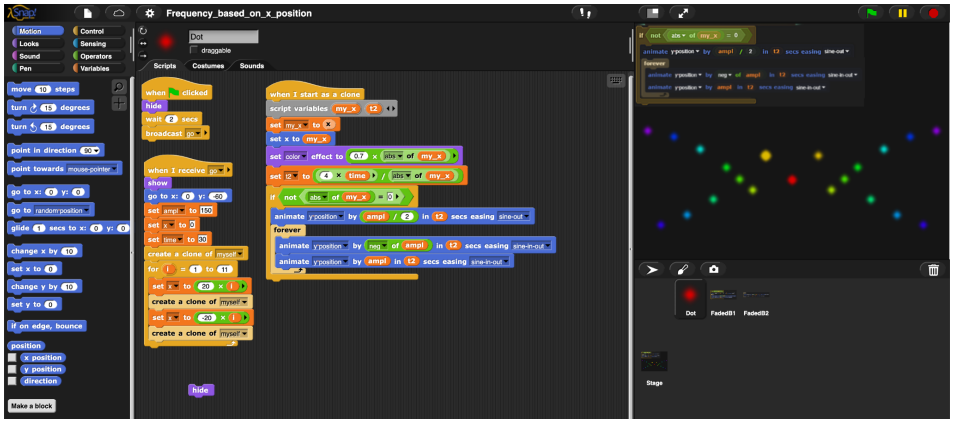

Figure 2: Snap! (https://snap.berkeley.edu/)

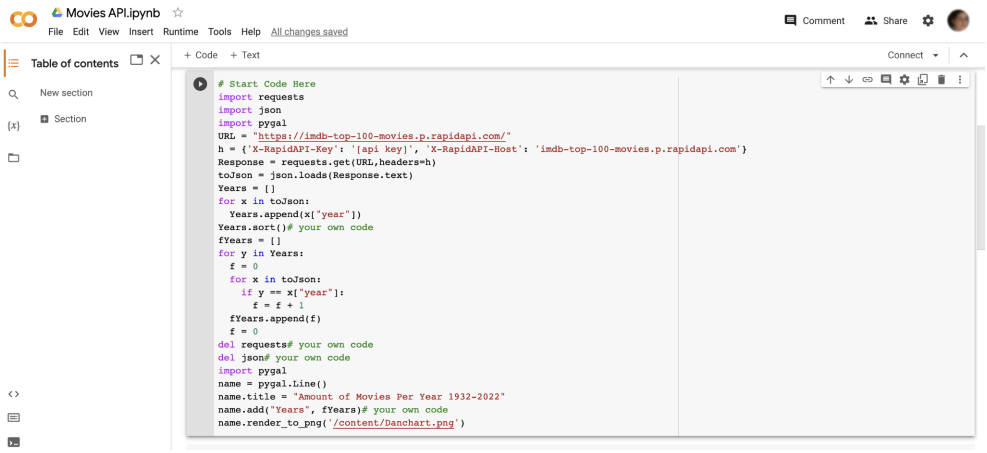

Figure 3: Google Colab (https://colab.research.google.com/)

The next most common languages are R (DataClassroom, Google Colab, Kaggle) and SQL (BlockSQL, DBSnap). Blockly is the tool that supports the widest variety of scripting languages. A drop-down menu within Blockly allows quick switching between code views in multiple languages.

4.2.4 Instructional Materials. Most of the tools (17 out of 25) provide instructional materials. These materials include written guides, video tutorials and demos, presentations, lesson plans, and sample projects. Some tools even have a section or repository dedicated to instructional materials. In Dataclassroom, for example, there is a user guide detailing how to work with data, conduct statistical analysis, and create visualizations. Additionally, under their resource library, they have ready-to-teach datasets accompanied by complete lesson plans. Teachers and students can use such materials to perfect their skills while working with the tools. Indeed, in recent research with math, computer science, and data science instructors, the cruciality of such instructional resources was highlighted [26].

4.2.5 Classroom Assignments Creation. Along with instructional materials, we also examined whether the tools enable the creation of learning activities in an integral manner. Seven of the 25 studied tools allow teachers to create classroom assignments. For example, in EduBlocks, instructors can create an assignment that includes a name, description, and initial project created using the tool, and then can link it to a specific class and then grade it after submission. Other tools provide a rudimentary feature that can be used to write assignments for learners. BlockPy, for example, has a component for creating task instructions with a built-in text editor that allows adding links, images, and tables. Dataclassroom and CODAP enable embedding GoogleDoc or HTML files to create a guided assignment. mBlock enables the creation of assignments using the integration of Google Classroom.

4.3 RQ3: K-12 Data Science Education Tools Accessibility Features

We looked for five kinds of accessibility features in each tool, based on prior work on software accessibility [19]. A complete listing of each tool's accessibility features can be found in Appendix3, Table A.3.

4.3.1 Screen Reader Support. To evaluate screen reader capabilities, we started with a "tab test", which means we pressed the "tab" key repeatedly to see if relevant objects were selected. This is important as this is how a screen reader navigates a graphical user interface to the narrative of the various features and objects. Only two of the tools looked at in this study – Tuva and DataClassroom – passed this test. Part of the reason for the low success rate is due to the fact that many of the K-12 data science tools have coding or interactive graphing panes that are tabbed to as a single object, which the screen reader will often describe as a "Coding Window" or a similar term. While this tells the user where they are on the screen, it is only useful to a visually impaired learner.

4.3.2 Color Palette. To evaluate whether or not the environment included accessible color palettes for color-blind users, we started by creating a basic graph (ideally a pie chart or side-by-side bar chart) to compare the color palette to a list of problematic color combinations for people with common color vision deficiencies (e.g., green & red, green & brown, blue & purple, etc.). One tool was removed from this analysis because it did not have plotting capabilities; of the remaining 24, all 24 had default color combinations different from the problematic pairs listed above.

4.3.3 Simplified Representations. Our next criteria investigated if or how it was possible to simplify the graphical representations provided or generated by the tool. This is important as removing extraneous data, or visual elements (e.g., unneeded lines and redundant labels) can be beneficial for neurodiverse learners. Again, one tool was removed from the first analysis because it did not support plotting. Of the remaining 24, only two tools examined (GapMinder and BridgesCS) supported some options to simplify graphical representations, but the options were limited.

4.3.4 Customized Interactions. Our penultimate accessibility criteria investigated if the data science tool supported customized interaction patterns. Particularly, if there was an option to customize how the users created or interacted with data graphics. This is important as such flexibility can make tools accessible for learners with fine motor difficulty. Adjusting the sensitivity of inputs or the movements associated with zooming or panning can make tools more accessible. None of the tools provided an option to customize the standard features for interacting with a graph or draggable blocks: the closest was GapMinder, which allows you to add the ability to pan with the cursor and zoom with the mouse wheel but does not allow you to modify the movement necessary for these actions. This feature should be strongly considered for existing and future projects in data science education, particularly for drag-and-drop block-coding or data-visualization programs where a motor disability might significantly hinder a student's ability to use the tool.

4.3.5 Font Size Modification. The final accessibility criteria examined if tools allowed the learner to modify the font size. This can be an important accessibility feature for learners with limited vision or difficulty with reading. Seven of the studied tools supported this behavior (2 using some form of "zoom" button that magnified the block-coding pane). While this is somewhat mitigated by most web browsers allowing magnification of the current tab, effectively changing the font size of the window, we were surprised to see that most of the tools we surveyed did not include some ability to change font size within the tool.

5 DISCUSSION

This paper presents the results of a systematic analysis of data science tools that are or can be, used in introductory data science education. Our analysis identified 25 tools, which we then analyzed according to 15 metrics, organized around three areas to deepen our understanding of the field's current state. Such tools are essential enablers for data science education, and it is thus crucial to understand the affordances, drawbacks, and differences between available tools. In this section, we reflect on the results of our analysis and highlight encouraging trends, opportunities for future design work, and the implications of these findings for both designers and educators.

5.1 Trends in Data Science Tools for K-12 Learners

In looking across the 15 analytic dimensions employed in this work, we see some emerging trends and interesting tensions in the design of K-12 data science education. One encouraging pattern across the tools is the ease with which they allow exploring exciting and diverse datasets. This can be seen by the tools providing built-in datasets and supporting mechanisms to import new data sets. One particular approach that we think is promising is scaffolded support for querying external APIs. Providing low-threshold approaches to allow novices to explore and pull in data from publicly available datasets can broaden the questions that learners can pursue and increase the authenticity of educational data science activities as learners will be learning with real, live datasets. Coupling live data with interfaces that allow novices to explore trends and answer questions they are interested in can make for a compelling introductory data science learning experience and help K-12 students understand the power and utility of having foundational data science skills.

In examining the tools identified as being used in K-12 data science education, an interesting tension emerges in the relationship between data science and programming. While some tools make programming the central mechanism by which learners will engage in data science (e.g., Jupyter Notebook, iSnap, Quorum, Kaggle) other tools do not include any programming or programming-like interfaces for learners (e.g., CODAP, GapMinder, Tuva). Further, how coding is presented to learners (i.e., graphical block-based vs. conventional text-based programming) also highlights a difference in what is prioritized by the tool and what is being foregrounded. The decision of if and how programming is being included in the environment impacts what ages/grades the tool is appropriate for, the necessary prior knowledge needed by the educator, the level of scaffolds needed by the environments, and the ways the tool can and cannot connect with data science tools and practices beyond the specific learning environment. Given that this review was interested in data science tools that span K-12, it is not surprising to find a breadth of ways of engaging (or not engaging) with programming. Thinking through how these tools might fit together and what tools fit where across the K-12 spectrum is an important open question for further work.

5.2 Implications for Designers

The analysis presented in this work has several implications for designers and suggestions for future work. The first set of implications is in response to the third research question exploring various ways tools attend to accessibility. It is critical that the designers of K-12 data science tools consider accessibility as a foundational goal. This is a legal requirement (at least in the United States) and a central aspect of creating equitable tools. In this analysis, accessibility was operationalized across five dimensions, some of which should be relatively easy to address in the design or re-design of data science tools. Giving special consideration to the color palette used (or offered), allowing learners to modify the text size, and providing options for simplifying data visualizations seem like relatively straightforward features to implement/add and should be prioritized in the design/re-design of K-12 tools. Other accessibility features, such as supporting customized interactions and screen readers, may take more effort but are still important and worthy pursuits for designers of educational data science tools.

A second implication for designers is acknowledging the various practices and skills that constitute data science and thinking through if/how their tool supports the breadth of what it means to engage in data science. While data visualization and statistical analysis seem to be prioritized in the reviewed tools, thinking through how to support the identification of relevant datasets, how to clean and normalize them, and how to communicate insights are all critical parts of the data science process. Our analysis suggests there is an opportunity for designing new K-12 data science tools that emphasize these other aspects of data science. In particular, a tool that helps learners identify potentially useful datasets and determine if/how a given data might be useful for answering a question is emerging as an under-explored design space in K-12 data science education.

5.3 Implications for Educators

This work also has implications for educators and can help them decide what tool(s) to use in their classroom. Prior work has found that data science instructors favor tools that support diverse online materials, provide explanations and examples, and include classroom activities [26]. In including supplemental educational materials as part of our analysis of the tools, we hope to provide a fuller picture for educators of the tools available to them. Furthermore, this analysis can live alongside other efforts to support educators in making informed decisions about tools and curricula they choose for their classrooms, such as the TEC Rubric [29].

It is our aspiration that this analysis would also assist educators in determining when and where to include data science in the K-12 curriculum. Following the approach seen with K-8 computer science and computational thinking (e.g., [15]), educators are exploring ways to integrate data science across the curriculum rather than create a stand-alone course. In highlighting what tools support importing learner-created data sets or supporting pulling in live datasets that may be relevant to a given science or social studies course, this analysis may help inform when/how data science shows up across the K-12 curriculum.

5.4 Limitations

While this systematic review was performed to the best of our ability, it has limitations. The first one is related to locating and extracting the data. We may have missed tools not mentioned in the sources reviewed or published in venues beyond ACM and IEEE. To minimize this limitation, we also used the recommendations of content experts who helped expand the list of tools we explored. The second limitation relates to the difficulty of analyzing a rapidly changing landscape. This analysis only includes tools we could identify at a certain moment. Over time, more tools will be developed. Therefore, as soon as this analysis was complete, it was outdated. This is also the challenge designers and educators face when trying to implement updated and relevant technologies in their curricula.

6 CONCLUSIONS

In light of the growing importance of data in society, it is crucial that students understand the role of data in their lives, develop an understanding of what data science is, and experience data analysis practices. The existing data analysis and visualization tools are essential in introducing students to the field. Therefore, examining the various data science tools available is an important consideration for educators and instructional designers. This review advances our understanding and helps us identify gaps and opportunities for creating new and innovative tools.

ACKNOWLEDGMENTS

This research is supported by the National Science Foundation (Award #2141655).

REFERENCES

[1] Bargagliotti, A. *et al.* 2020. *Pre-K-12 guidelines for assessment and instruction in statistics education II (GAISE II).* American Statistical Association.
[2] Batt, S. *et al.* 2020. Learning Tableau: A data visualization tool. *The Journal of Economic Education.* 51, 3–4 (Sep. 2020), 317–328. DOI:https://doi.org/10.1080/00220485.2020.1804503.
[3] Bisong, E. 2019. Google Colaboratory. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners.* E. Bisong, ed. Apress. 59–64.
[4] Brady, C. *et al.* 2022. Block-based abstractions and expansive services to make advanced computing concepts accessible to novices. *Journal of Computer Languages.* 73, (Dec. 2022), 101156. DOI:https://doi.org/10.1016/j.cola.2022.101156.
[5] Chandel, S. *et al.* 2022. Training and evaluating a Jupyter notebook data science assistant. arXiv.
[6] CODAP - Common Online Data Analysis Platform: 2022. https://codap.concord.org/. Accessed: 2022-12-19.
[7] DataClassroom: 2022. https://about.dataclassroom.com. Accessed: 2023-02-21.
[8] Deahl, E. 2014. Better the data you know: Developing youth data literacy in schools and informal learning environments.
[9] Erickson, T. 2016. Practical public online data: Introducing Tuva and CODAP. *Promoting Understanding of Statistics about Society IASE Roundtable Conference* (Dec. 2016).
[10] Gavriilidis, H. *et al.* 2023. SheetReader: Efficient specialized spreadsheet parsing. *Information Systems.* 115, (May 2023), 102183. DOI:https://doi.org/10.1016/j.is.2023.102183.
[11] Gould, R. 2021. Toward data-scientific thinking. *Teaching Statistics.* 43, S1 (Jul. 2021). DOI:https://doi.org/10.1111/test.12267.
[12] Haq, H.B.U. *et al.* 2020. The popular tools of data sciences: Benefits, challenges and applications. (2020).
[13] Krishnamurthi, S. *et al.* 2020. Data science as a route to AI for middle- and high-school students. arXiv.
[14] LaMar, T. and Boaler, J. 2021. The importance and emergence of K-12 data science. *Phi Delta Kappan.* 103, 1 (Sep. 2021), 49–53. DOI:https://doi.org/10.1177/00317217211043627.
[15] Lee, I. *et al.* 2014. Integrating computational thinking across the K–8 curriculum. *ACM Inroads.* 5, 4 (Dec. 2014), 64–71. DOI:https://doi.org/10.1145/2684721.2684736.

[16] Lee, V.R. *et al.* 2021. A call for a humanistic stance toward K–12 data science education. *Educational Researcher.* 50, 9 (Dec. 2021), 664–672. DOI:https://doi.org/10.3102/0013189X211048810.

[17] Lee, V.R. and Delaney, V. 2021. Aesthetics of Authenticity for Teachers' Data Set Preferences. (2021), 8.

[18] Lin, Y. and Weintrop, D. 2021. The landscape of block-based programming: Characteristics of block-based environments and how they support the transition to text-based programming. *Journal of Computer Languages.* 67, (Dec. 2021), 101075. DOI:https://doi.org/10.1016/j.cola.2021.101075.

[19] Marriott, K. *et al.* 2021. Inclusive data visualization for people with disabilities: a call to action. *Interactions.* 28, 3 (May 2021), 47–51. DOI:https://doi.org/10.1145/3457875.

[20] McNamara, A. 2019. Key attributes of a modern statistical computing tool. *The American Statistician.* 73, 4 (Oct. 2019), 375–384. DOI:https://doi.org/10.1080/00031305.2018.1482784.

[21] Pimentel, D.R. 2022. Tools to support data analysis and data science in k-12 education. (2022), 22.

[22] Power BI: 2023. https://powerbi.microsoft.com/en-us/. Accessed: 2023-02-21.

[23] Ridgway, J. 2016. Implications of the data revolution for statistics education. *International Statistical Review.* 84, 3 (2016), 528–549. DOI:https://doi.org/10.1111/insr.12110.

[24] Rosenberg, J.M. *et al.* 2022. Big data, big changes? The technologies and sources of data used in science classrooms. *British Journal of Educational Technology.* 53, 5 (Sep. 2022), 1179–1201. DOI:https://doi.org/10.1111/bjet.13245.

[25] Schanzer, E. *et al.* 2022. Integrated data science for secondary schools: Design and assessment of a curriculum. *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education* (Providence RI USA, Feb. 2022), 22–28.

[26] Schwab-McCoy, A. *et al.* 2021. Data science in 2020: Computing, curricula, and challenges for the next 10 years. *Journal of Statistics and Data Science Education.* 29, sup1 (Jan. 2021), S40–S50. DOI:https://doi.org/10.1080/10691898.2020.1851159.

[27] Tableau: Business intelligence and analytics software: 2023. https://www.tableau.com/node/62770. Accessed: 2023-02-21.

[28] The Brown PLT Blog 2021. Adding function transformers to CODAP.

[29] Weintrop, D. *et al.* 2019. The teacher accessibility, equity, and content (TEC) rubric for evaluating computing curricula. *ACM Transactions on Computing Education.* 20, 1 (Dec. 2019), 5:1-5:30. DOI:https://doi.org/10.1145/3371155.

[30] Weintrop, D. and Holbert, N. 2017. From blocks to text and back: Programming patterns in a dual-modality environment. *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (New York, NY, USA, Mar. 2017), 633–638.

[31] Weintrop, D. and Wilensky, U. 2018. How block-based, text-based, and hybrid block/text modalities shape novice programming practices. *International Journal of Child-Computer Interaction.* 17, (Sep. 2018), 83–92. DOI:https://doi.org/10.1016/j.ijcci.2018.04.005.

[32] Wimmer, H. *et al.* 2016. A comparison of open source tools for data science. *Journal of Information Systems Applied Research.* 9, 2 (2016), 4–12.

A APPENDICES

A.1 Appendix1

Table A.1: Coding Scheme of Tools' Capabilities (RQ1)

Tools	Data Manipulations	Statistical Capabilities	Data Visualization			Data Availability				Extensibility
			Tabular Display	Type of Graph	Creation Method	Built-in Data	Import	API	Export	
Blockly	Aggregating	N/A	×	Line	Code	×	√	×	√	√
BlocklySQL	Filtering, Sorting, Aggregating	N/A	√	N/A	N/A	√	×	×	√	√
BlockPy	Filtering, Sorting	N/A	×	Scatterplot, Bar Chart, Line Chart, Box Plot, Histogram	Blocks / Code	√	√	×	√	×
Bridges CS	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	×	Line Chart	Code	√	√	×	√	×
CODAP	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Scatterplot, Bar Chart, Histogram, Box Plot, Map	GUI	√	√	√	√	√
DataClassroom	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Scatterplot, Bar Chart, Line Chart, Pie Chart, Histogram, Box Plot	GUI	√	√	×	√	×
Datacommons	N/A	N/A	√	Scatterplot, Map	GUI	√	×	√	√	√
DBSnap	Filtering, Deleting, Sorting, Aggregating	N/A	√	Scatterplot, Bar, Line Chart, Pie Chart, Area, Histogram, Bubble Chart	Blocks	√	√	×	×	×

EduBlocks	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	×	Scatterplot, Bar, Stack Chart, Bar Chart, Line Chart, Stack Graph, Pie Chart, Radar	Blocks / Code	×	√	×	×	√
GapMinder	N/A	N/A	×	Scatterplot	GUI	√	×	×	√	×
Google Colab	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Line Chart, Bar Chart, Histogram, Scatterplot, Stack Chart, Pie Chart, 3D Graphs	Code	×	√	√	√	×
GP	Filtering, Deleting, Sorting	N/A	×	Scatterplot	Blocks	×	√	×	√	×
iNZight	Filtering, Sorting	Correlation, Linear Regression	√	Bar, Dotplot, Box Plot, Grid Density, Hexbin Plot	GUI	√	√	√	√	×
iSnap	Filtering, Deleting	N/A	×	Scatterplot, Line Chart	Blocks	×	√	×	√	×
Jupyter Not ebook	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Line, Scatterplot, Bar, Histogram, Box Plot, Violin, Pie Chart, etc.	Code	×	√	√	√	√
kaggle	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Scatterplot, Bar, Line Chart, Histogram, Box Plot, Area Plot, Pie Chart	Code	√	√	×	√	×

Make Code Data Science Editor	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	√	Scatterplot, Blocks Bar, Line Chart, Histogram, Box Plot, Heatmap	√	√	×	√	√
mBlock	Filtering	N/A	√	Scatterplot, Blocks Bar, Line / Code Chart, Pie Chart	√	√	×	√	×
NetsBlox	N/A	N/A	×	Scatterplot, Blocks Line Chart, Histogram, Map, Bar Chart	×	√	√	×	×
Pyret	filter, sort	Correlation, Linear Regression	√	Scatterplot, Code Bar, Line Chart, Pie Chart, Histogram	×	√	×	√	×
Quorum	Filtering, Deleting, Sorting, Aggregating	Correlation, Linear Regression	×	Scatterplot, Code Bar, Pie Chart, Line Chart, Histogram, Box Plot, Violin Plots	×	√	×	√	✓
Scratch Data Blocks	N/A	N/A	√	Scatterplot, Blocks Line Chart	×	√	×	√	√
Snap!	Filtering, Deleting	N/A	√	Scatterplot, Blocks Line Chart	×	√	×	√	√
TinkerPlots	filter, sort, aggregate, delete	Correlation, Linear Regression	√	Scatterplot, GUI Pie Chart, Histogram	√	√	×	√	×
Tuva	Filtering, Deleting	Correlation, Linear Regression	√	Scatterplot, GUI Bar, Line Chart, Pie Chart, Histogram, Box Plot, Map	√	√	×	√	×

A.2 Appendix2

Table A.2: Coding Scheme of Supported Interactions and Educational Features (RQ2)

Tools	Runtime Environment	Programming Modality	Scripting Language	Instructional Materials	Classroom Assignments
Blockly	Browser	Read-Only One-Way Transition	Python, Javascript, PHP, Lua, DART	×	×
BlocklySQL	Browser	Dual Modality	SQL	×	×
BlockPy	Browser	Dual Modality	Python	√	√
Bridges CS	Win/Mac	Text-Based	C++, Java, Python	√	×
CODAP	Browser	GUI	N/A	√	√
DataClassroom	Browser	GUI / Text-Based	R	√	√
Datacommons	Browser	GUI	N/A	√	×
DBSnap	Browser	Read-Only One-Way Transition	SQL	×	×
Edu Blocks	Browser	Editable One-Way Transition	Python	√	√
GapMinder	Browser/Win/Mac	GUI	N/A	√	×
Google Colab	Browser	Text-Based	Python, R	×	×
GP	Browser/Win/Mac	Block-based	N/A	√	×
iNZight	Browser/Win/Mac/Linux	GUI	N/A	√	×
iSnap	Browser	Block-based	N/A	×	×
Jupyter Not ebook	Browser/Win/Mac/Linux	Text-Based	Python	×	×
kaggle	Browser	Text-Based	R, Python	√	√
MakeCode Data Science Editor	Browser	Block-based	N/A	×	×
mBlock	Browser/Win/Mac/iOS/Android	Read-Only One-Way Transition	Python	√	√
NetsBlox	Browser	Block-based	N/A	√	×
Pyret	Browser	Text-Based	Pyret	×	×
Quorum	Browser/Win/Mac	Text-Based	Quorum, MySQL	√	×
Scratch Data Blocks	Browser	Block-based	N/A	√	×
Snap!	Browser	Block-based	N/A	√	×
TinkerPlots	Win/Mac	GUI	N/A	√	×
Tuva	Browser	GUI	N/A	√	√

A.3 Appendix3

Table A.3: Coding Scheme of Accessibility Features (RQ3)

Tools	Screen Reader Support	Color Palette	Simplified Representations	Customized Interactions	Font Size Modification
Blockly	×	N/A	N/A	×	×
BlocklySQL	×	N/A	N/A	×	×
BlockPy	×	√	×	×	×
Bridges CS	×	√	√	×	×
CODAP	×	√	×	×	×
DataClassroom	√	√	×	×	√
Datacommons	×	√	×	×	×
DBSnap	×	N/A	N/A	×	×
Edu Blocks	×	√	×	×	√
GapMinder	×	√	√	×	√
Google Colab	×	√	×	×	√
GP	×	N/A	N/A	×	×
iNZight	×	√	×	×	√
iSnap	×	N/A	N/A	×	×
Jupyter Not ebook	×	√	×	×	√
kaggle	×	√	×	×	×
MakeCode Data Science Editor	×	√	×	×	×
mBlock	×	N/A	N/A	×	×
NetsBlox	×	√	×	×	×
Pyret	×	√	×	×	√
Quorum	×	√	×	×	×
Scratch Data Blocks	×	N/A	N/A	×	√
Snap!	×	√	×	×	×
TinkerPlots	×	√	×	×	×
Tuva	√	√	×	×	√