Computer Science Education

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/ncse20

A randomized controlled trial on the nomenclature of scientific computing

Timothy Kluthe, Hannah Stabler, Amelia McNamara & Andreas Stefik

To cite this article: Timothy Kluthe, Hannah Stabler, Amelia McNamara & Andreas Stefik (10 Nov 2024): A randomized controlled trial on the nomenclature of scientific computing, Computer Science Education, DOI: 10.1080/08993408.2024.2403971

To link to this article: https://doi.org/10.1080/08993408.2024.2403971

COMPUTER SCIENCE EDUCATION
https://doi.org/10.1080/08993408.2024.2403971

Routledge
Taylor & Francis Group

EMPIRICAL STUDIES

A randomized controlled trial on the nomenclature of scientific computing

Timothy Kluthe [a], Hannah Stabler [a], Amelia McNamara [b] and Andreas Stefik [a]

[a]Computer Science, University of Nevada, Las Vegas, Las Vegas, NV, USA; [b]Computer & Information Sciences, University of St. Thomas, St. Paul, MN, USA

ABSTRACT

Background and Context: Data science and statistics are used across a broad spectrum of professions, experience levels and programming languages. The popular scientific computing languages, such as Matlab, Python and R, were organized without using empirical methods to show evidence for or against their design choices, resulting in them feeling eclectic or esoteric in their design.

Objective: To meaningfully organize scientific computing based on evidence gathered through user feedback, build a statistical package based on the findings and provide a replication packet to run similar studies on people with different backgrounds.

Method: A randomized controlled trial using a weighted, ranked choice survey ($n = 118$) with between-subjects design having two independent variables: Language Group (Matlab, Python and R) and Method Name options. Our dependent variable was a normalized preference rating.

Findings: There was a very small interaction between Language Group and Method Name. Language Group did not have a statistically significant effect, but Method Name did ($F(4, 27037) = 2211.23$, $p < .001$)($\eta_p^2 = .247$). Finally, many names in Matlab, Python and R were ranked so poorly that they were not statistically significantly different from a random word in 63.0%, 62.2% and 30.4% of concepts respectively.

Implications: We found organized and structured names were ranked by a large margin, suggesting statistical programming today likely needs considerable improvement. Finally, we outline a statistical package built using these principles, provide comparison scripts and describe some of the challenges from going from simple surveys to in-practice libraries.

ARTICLE HISTORY
Received 23 August 2023
Accepted 10 September 2024

KEYWORDS
Programming language usability; scientific computing; data science; statistics

1. Introduction

1.1. Problem

The National Science Board's 2022 Science and Engineering Indicators report shows an estimate of United States gross research and development expenditure of $657.5 billion

CONTACT Timothy Kluthe kluthe@unlv.nevada.edu Computer Science, University of Nevada, Las Vegas, Las Vegas, USA

Supplemental data for this article can be accessed online at https://doi.org/10.1080/08993408.2024.2403971.

(Burke et al., 2022). The mathematical and statistical analysis involved in this research can be complicated and time-consuming, and it often requires the use of scientific computing languages.

As data has become easier to collect and cheaper to store (O'Connor, 2014), there has been increasing demand for data science skills and training programs. For example, since its creation in April 2014, the Johns Hopkins Data Science Specialization online program has seen 4 million enrollments (Kross et al., 2020). Even in high-school, movements like Data Science for Everyone are encouraging data science in a K-12 setting (The Rockefeller Foundation, 2023).

Despite the interest, several evidence-based studies have been conducted that show stark problems with current data science tools. While there are many issues, problems include poor naming conventions (Rafalski et al., 2019) and various problems writing and maintaining code when using online "Notebooks" (Chattopadhyay et al., 2020). Furthermore, statistical anxiety is both real and has a real-world impact on student grades and learning behaviors (Macher et al., 2012; Onwuegbuzie & Wilson, 2003), in addition to documented evidence that it impacts people of different genders in different ways (Rendulic & Terrell, 2012). Lessons and experience using statistical software, while improving statistical anxiety, did not eliminate it completely (Rode & Ringel, 2019a). Finally, data science has significant issues of accessibility to people with disabilities, including the generation of charts for the blind (Godfrey et al., 2018; Zong et al., 2022), but also in ensuring that our colleagues and students with learning disabilities are looking at reasonably organized and structured libraries and naming to assist. Naming, however, is not enough. Even a well named and structured library may be hard to use, as there are a plethora of decisions to get "right".

Thus, we have begun an investigation into "how" we might re-think statistics, starting from naming conventions and organization. Scientific computing uses many different statistical tests and data science operations, which are used in very specific situations based on the type of data you have, how you are trying to compare sets of data or how you want to transform the data. For example, if you are comparing the means of two groups you would need a "t-test", but if you are comparing more you might need an "ANOVA". Building on that, you may also need to run a "Shapiro-Wilk test" to check if your sample fits a normal distribution, because if it does not, then you would need to instead run a non-parametric test. In this case, you may need a "Mann-Whitney", "Wilcoxon Signed-Rank", "Kruskal-Wallis" or "Friedman" test. Our running hypothesis is that if we start by thinking about what these statistical tests and data science operations actually do and are used for, we might be able to make the nomenclature match more directly with how the library is organized and make it more obvious and meaningful to students. If we then build the library and provide a way to reproduce the work, we might be able to garner information on: 1) how the naming choices and organization of the library into groupings of related actions could improve the use of the library in practice and 2) how the naming choices impact many different samples of people through a community effort by performing a replication of this nomenclature study.

We make three contributions in this work. First, we created a survey where we tried to create meaningful names for a wide variety of statistical tests. For each of the tested statistical concepts, we included two of our own names derived from reading about the purpose and usage of the statistical tests, reading the equations, and then debating what

we thought would be sensible. We also included names from the languages Matlab, Python and R, the technical names observed in textbooks, and a word generated randomly as a control. We then gathered user feedback about name preferences based on a provided description of a statistical concept. Results show that participants ranked our potentially "meaningfully generated" words highly. A large proportion of words in the major data science languages did so poorly that their rankings often did not exceed words we generated randomly.

Second, in order to help us better understand whether our library might make statistics easier in practice, we undertook a dog-fooding activity, a process in which you use your own product as a way of testing it in practice. Notably, we built a new statistical library for all of the statistical tests based on the organizing principles we described. We report on some of the challenges we faced, which as the reader can probably imagine given the mathematics is intense, were numerous and complex. We make no claims that this library is good or bad, but we provide runnable scripts using it, along with R, as a way to "rethink" what such a system might be like in practice.

Finally, we have a primary focus on making data science easier by organizing it in a more obvious and meaningful way. Our sample consisted primarily of undergraduates with a Computer Science background. We recognize that this does not generalize to the entire population of potential data science users, but in an effort to "aim down" by making a scientific computing library that is easier for new users to interact with, we have used a sample of participants with very little statistical knowledge as a starting point. Furthermore, we hypothesize a much more diverse sample of people may be necessary to know how to organize statistics in a general purpose way. For this reason, we provide here a standardized way of replicating our study through a repository with a Docker container-based replica of our experimental website; which will spin up containers with the full web stack necessary for hosting without needing the domain knowledge of how to put those pieces together. This simplifies the replication of this study down to running a few scripts and gathering participants, and we call on other researchers to investigate this issue with their own sample of participants. If the broader community can easily replicate the study, perhaps we can obtain information quickly on different ages, experience levels, majors, professions, disabilities, others. This data can be fed back into the design of the library, providing an evidence-based and community driven approach to building the next generation of statistical languages.

The rest of this paper proceeds as follows: First, we discuss relevant work and then move to our study and its results. After discussing our study, we follow with a brief description of the challenges in building a statistical library in the manner described and then finally move to our method for replication and conclude.

1.2. Review of relevant scholarship

To our knowledge, there has been no evidence-based research done in the design of current scientific computing languages. A literature review by Kaijanaho investigated the historical use of empirical evidence of the human factors involved in programming language design (Kaijanaho, 2015), but did not report any findings on scientific computing. Studies have investigated the usability of programming languages in general, such as Stefik et al.'s (Stefik et al., 2011) work on the impact of design choices on accuracy rates,

have been used to make evidence-based choices in the design of a language (e.g. syntax and semantics), but are unrelated to data science. More recently, Becker et al. (Becker et al., 2019) provided a comprehensive review of the literature on compiler errors, runtime errors and warnings to provide a set of guidelines to help make those messages more meaningful. Stefik and Seibert highlighted the importance of conducting empirical studies on programming languages has on Computer Science education; programming language syntax presents a significant barrier to novice programmers and through a series of four empirical studies, they first identified preferred syntax choices through a set of surveys and then tested those choices against the status quo when being used in small programming tasks (Stefik & Siebert, 2013).

A similar line of evidence-based research could greatly benefit scientific computing languages as it will have an impact on a growing population of users. While it may seem like a niche segment of Computer Science education, data science coursework is increasingly integrated within the Computer Science curriculum. In addition to data science courses, some undergraduate and graduate Computer Science students interact with statistics as a section in some courses. As Computer Science education researchers, we teach statistics to our students using statistical programming languages, and this is similarly true for graduate students and faculty in other areas of Computer Science research. We do not do this because they are easy to use, but because they are necessary in order to use statistics in quantitative empirical research and to write research papers. Computer Science students are increasingly needing to interact with these types of languages, either at an introductory level in learning data science in general and at a higher level putting it into practice through research. As part of our experimental design, we have selected a wide range of statistical tests and data science operations; starting with what is available in the Advanced Placement Statistics coursework, an introductory college-level statistics course that is common throughout the United States, to try and cover anything that might be used as a base-level (e.g. "Skew", "T-Test", "Confidence Interval"), and expanding into some of the more complex concepts that would be more suitable for Computer Science graduate and faculty analyzing a broad variety of research (e.g. "Kruskal–Wallis Test", "Cohen's kappa", "Mauchly's Sphericity Test"). In this study, similar to the methodology used in Stefik and Seibert (Stefik & Siebert, 2013), we will investigate improving the nomenclature used in scientific computing by conducting a survey into the naming convention of data science related methods and operations. The tools and programming languages used in scientific computing may suffer from similar design problems found in general purpose programming languages and further investigation could provide similar benefits in lowering the barrier to entry.

There have been attempts to improve the development process, maintainability and reproducibility of scientific computing languages at a higher level, such as through processes like Agile, online resources like Jupyter Notebook, containers and cloud-based resources like Docker and Code Ocean, and in general, applying best practices from software development ("Easing the burden of code review, 2018; Noble & Lewitter, 2009; White et al., 2013; Wilson et al., 2014). While these technologies can help to improve the day-to-day use, they do not change the core. Notably, Chattopadhyay et al. interviewed professionals in data science and categorized the kinds of problems they face while using online "Notebooks" (Chattopadhyay et al., 2020). In particular, one participant in that study mentioned they had trouble

remembering the syntax, saying they had to "know all the function names and class names correctly and have another browser open to search for help and documentation" (Chattopadhyay et al., 2020). Names alone are not enough; for example, names like "Greenhouse – Geisser correction" or "t-test" hardly imply what the math actually does or the purpose.

Previous work comparing the syntax of variations of the R programming language and how it impacts novice programmers provides a foundation for this study, as it showed with evidence that there is a problem in the design of scientific computing languages (Rafalski et al., 2019). Several studies have investigated the readability of source code in relation to identifiers naming convention using full words and abbreviated words, and found that both provided more readability than single letter identifiers (Lawrie et al., 2006), but there was minimal difference between full words and their abbreviated variations when performing programming comprehension tasks (Lawrie et al., 2006; Scanniello & Risi, 2013). Binkley et al. ran a study that investigated the use of camel-case versus underscore delimiters in identifier names and found that, while professionals were not impacted by either, beginners benefited in both completion time and accuracy when writing code with camel-case styled identifiers (Binkley et al., 2013). Such differences appear small, but the point is they can matter.

Deissenboeck and Pizka recommend defining a naming dictionary and employing two rules: consistency and conciseness (Deissenboeck & Pizka, 2006). Consistency takes into consideration the semantics of the words as it is important that the name maps to the concept it represents. In other words, you avoid using the same word for different meanings (e.g. "file" to describe a filehandler or filename) and different words for the same meaning (e.g. "accountNumber" and "number" to describe an account number) (Deissenboeck & Pizka, 2006). For example, conciseness is making use of the linguistic mechanism of hypernymy; when choosing a descriptive option, you want to pick the most *precise* descriptive options (e.g. "animal", "dog", "poodle" may all be accurate descriptors for a pet, but "poodle" helps to remove some of the ambiguity) (Deissenboeck & Pizka, 2006). Previous work in psychology found that when faced with these types of linguistic interpretations while reading a sentence, the reader had to slow down to resolve the ambiguity by determining what the surrounding context was (Anderson, 1995).

The semantics of words also play a role in memorization and are a researched topic in neuroscience (Deese, 1959; Pardilla-Delgado & Payne, 2017; Roediger & McDermott, 1995). This is often termed the "congruence effect", and it has shown to benefit long-term memory and recall by improving the encoding phase of schema formation (Kapur et al., 1994). The ability to memorize new information can be influenced by how closely it relates to previously encoded schemas in semantic memory (Bartlett, 1932), and it has been widely accepted that retrieving this information from memory makes use of congruent schema structures (Bower, 1972; Tibon et al., 2014). Basically, by employing a rule of consistency to make sure there is a good semantic mapping between an identifier and the concept, there should be more success at the encoding stage of memorization and improved recall of name to concept pairings. For example, a t-test might be more memorable if it were called "CompareMeans" and "Kruskal-Wallis" might make more sense as "CompareRankedMeans". However, not all statistical

tests are easy to name and some take significant effort to even understand their purpose.

1.3. Hypothesis, aims, and objectives

The primary goal of the study part of this paper is to find a more direct mapping of the action an operation is meant to evoke to the name used. To that end, we investigated three key research questions:

- **RQ1**: do novice data science users have a preference in what method names are chosen for data science related operations?
- **RQ2**: does the option of existing nomenclature change the novice data science users' preference?
- **RQ3**: what are the best ranked method names for each concept based on novice data science users?

By collecting user feedback on preferred method names they would assign to the description of statistical tests and data science operations, we aim to gather some empirical evidence that sheds some light on the type of nomenclature a novice data science user may prefer and how this compares with some of the existing options in Matlab, Python and R.

Table 1. Participant demographics showing degree major, year in school and gender separated by assigned language group.

		Matlab (n)	Python (n)	R (n)
Degree Major	Computer Science	36	36	39
	Engineering	4	2	1
Year In School	Sophomore	1	1	1
	Junior	17	16	17
	Senior	21	21	21
	Graduate	1	0	1
Gender	Male	28	30	37
	Female	10	7	3
	Non-binary	1	1	0
	Other	1	0	0

Table 2. Additional participant demographics showing the mean and standard deviation of age and experience in years of programming, statistics, matlab, Python and R separated by assigned language group.

	Matlab $\bar{x}(s)$	Python $\bar{x}(s)$	R $\bar{x}(s)$
Age	24.28(8.22)	23.29(4.28)	22.25(2.75)
Experience in Programming (Years)	4.15(6.1)	3.29(1.8)	4.15(2.12)
Experience in Statistics (Years)	.93(1.16)	.68(.99)	.88(1.59)
Experience in Matlab (Years)	.3(.46)	.26(.5)	.38(.54)
Experience in Python (Years)	.75(1.13)	1.13(1.4)	1.0(1.48)
Experience in R (Years)	.38(.63)	.18(.39)	.28(.45)

2. Method

2.1. Participant characteristics

Demographic information about the participants in the study can be found in Tables 1 and 2, and provide some insights on the characteristics of the sample population. Table 1 includes the degree major, year in school and gender separated by assigned group. Note that the majority of participants are Computer Science majors and several are generically classified as Engineering. From the Computer Science courses that were recruited from, not every student was a Computer Science major, but each of these courses were 300 and 400-level coursework which is primarily Junior or Senior level courses. Although a few of the participants were self-reported as Engineering majors (specifically six in Computer Engineering and one in Mechanical Engineering), they all have passed similar Computer Science prerequisite coursework to be able to take these higher level Computer Science course. Additionally, Table 2 provides the mean and standard deviation for age (ranging from 18 to 60) and several self-reported years of experience in subjects that may be relevant to the study; programming, statistics, Matlab, Python and R. The inclusion criteria to participate in this study were that participants must be 18 years of age or older, and there was no other exclusion criteria.

2.2. Recruitment

The experiment was conducted through a web-based survey platform between October 2022 and May 2023. Participants were recruited from students in courses (e.g. CS 326: Programming Languages, CS 370: Operating Systems, CS 472: Software Product Design and Development) in the Computer Science department at University of Nevada, Las Vegas under IRB protocol UNLV-2022-77. Each participant was compensated with 2% extra credit in one of those courses.

2.3. Sample size, power, and precision

Ideally, a power analysis would provide some guidance on sample size, but we were unaware of any evidence-based research that has investigated questions similar that could directly compare, apples to apples, as a baseline. We had a sample size of 118 (124 before data exclusion).

2.4. Measures and covariates

The primary measure was the weighted, ranked choice survey. Due to some cases where there was no available method for one of the three existing languages (Matlab, Python or R), some of the questions only had 4 options and thus 20 points to allocate. To accommodate these differences in our analysis, the scores were normalized by the maximum available points.

2.5. Conditions and design

The experiment used a between-subjects design with 3 group assignments (Matlab, Python and R) and 46 tasks. Each task consisted of a description of a statistical concept with five name options for each. The provided name options for each task can be broken down into five categories:

- **Experimental**: An experimental method name that we created based on the use-case of the concept.
- **Experimental2**: A secondary experimental method name we created based on the use-case of the concept.
- **Existing**: The method name used to invoke the concept in one of the three existing programming languages.
- **Technical**: The technical name for the concept often observed in textbooks, but using the same formatting style as the Experimental options.
- **Random**: A randomly generated string of letters.

For the **Experimental** and **Experimental2** methods, we mapped these options semantically. Our thinking was that these statistical tests could potentially use names that describe what they do. The theory, although we use the term loosely, is one of pragmatism: *say what you do*. However, statistical analysis is complex and even deriving what a statistical function does is not always easy. As such, we followed a process. First, we researched the statistical concepts to better understand the purpose and approach of the tests (e.g. to name the Kruskal – Wallis test, you first need to learn what it does). We then considered recommendations from the literature in regard to previous work with naming conventions (Binkley et al., 2013; Deissenboeck & Pizka, 2006; Lawrie et al., 2006). Note that Deissenboeck and Pizka (Deissenboeck & Pizka, 2006) provide some guidelines for how to name identifiers, but they hardly apply to statistics directly and, so far as we can tell, do not represent some kind of more unified theoretical, let alone, predictive approach for naming conventions. For example, Lawrie et al. provide some evidence to suggest that single letters perform worse than full or abbreviated words in terms of readability (Lawrie et al., 2006), but this alone does not predict what naming convention humans would understand. When inspecting the name options in existing languages, we found that acronyms were not uncommon, which at least in practice does not follow Lawrie et al., nor does it say what you do (e.g. "dt", "lm", "df", "qf"). Finally, we ran all of our naming conventions by a Ph.D. statistician that checked our assumptions. Admittedly, our *say what you do* approach is pragmatic. There is, after all, no equation you can run as of the time of this writing where you can enter "Kruskal – Wallis" and get back a good naming convention along with corresponding, replicable, evidence of that fact with people at different ages, experiences, genders, disabilities or other properties. In real-world statistical tools, we suspect people have just guessed. We thus settled on creating two experimental choices in an effort to acknowledge the possibility that a plurality might be just as good (or might not).

For the **Existing** language, we used one exact method name that could be used to invoke that statistical test in one of the three languages (Matlab, Python and R). Note, there are potentially many valid method names from different libraries and packages that

will perform the same test (e.g. Pearson Correlation in Python could use "pearsonr" in scipy, "corrcoef" in numpy and "corr" in pandas). Rather than provide all possible method names for the existing name option or divide the participants into further group assignments, we elected to use just one valid, compilable method option per existing language that seemed reasonably common for that tool. Next, for the fourth option, as a middle ground between our two **Experimental** options and the **Existing** language option, we created the **Technical** option, which made use of the exact technical name for the concept that would be found in a textbook, but using the same naming convention style guide as the **Experimental** options (i.e. full words and camel-case). Finally, the **Random** option was the control which acts as a metaphorical placebo. These were generated using a random length string between 5 and 7 characters with alternating randomized consonants and vowels.

For each task, participants were presented with a description and one name choice for each of the five categories. Participants were separated into three groups, where their group assignment determined which existing language would be used for their **Existing** option. Thus, if a participant was assigned to the R group, for each task, one of their five available choices would be the method name used in R. Figure 1 shows the randomization of group assignments between the three existing languages (Matlab, Python and R) and what words would be shown to them as options. In this case, for the task "Pearson Correlation", each participant is presented with the same description: "Linear relationship between two variables". Additionally, they are each provided a set of five name options. You will notice that "Correlate", "CorrelateLinearly", "PearsonCorrelation" and "copov" appear in all three group's name options and relate to the categories **Experimental**, **Experimental2**, **Technical** and **Random** respectively. The last option listed in each set, "corr", "pearsonr" and "cor" are categorized as **Existing**. Note that, while this diagram

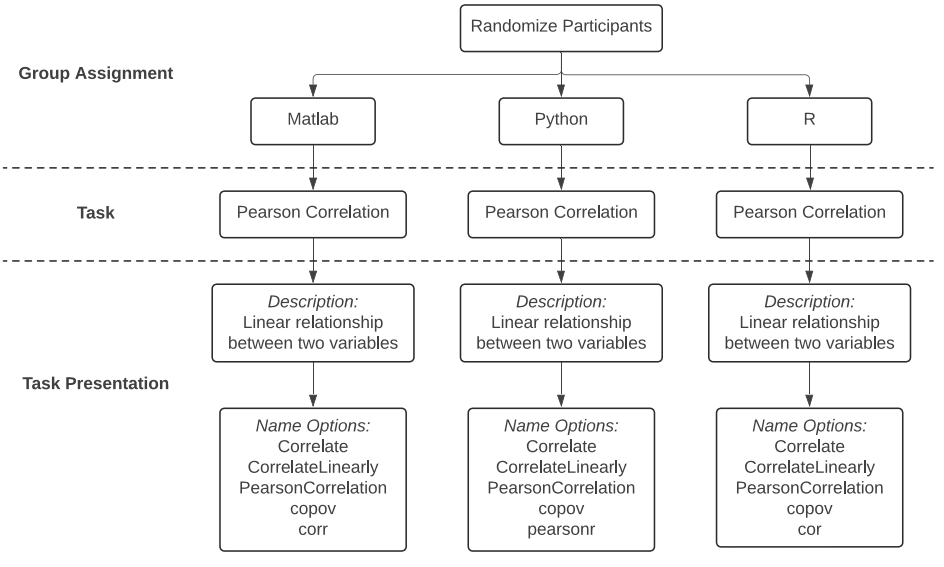

Figure 1. A block diagram that represents the varying Name Options available per task based on an individual participant's group assignment between matlab, Python and R.

shows consistent ordering of options between the 3 group assignments, when the tasks are presented, the ordering of name options would be randomized.

The purpose for separating participants into three separate groups and presenting them with only one of the **Existing** options was based on participant feedback in a pilot study where all three **Existing** options were presented. The feedback suggested that the participants felt biased towards selecting one of the three **Existing** options specifically because they saw three near-identical names and made the assumption that the name must be "correct" rather than what they found to personally be the most intuitive match with the description. To avoid the biasing of repetitious choices, we elected to split the design into three separate groups which allows us to gather a comparison with several popular existing languages without biasing the participant responses toward the **Existing** method names.

For the design of the survey, we weighed the available options of a rating system, such as Likert scale, or a ranking system. Although rating systems are fairly common, some researchers find them to be flawed (Jamieson, 2004). Jamieson et al. discusses the inherent problems of a Likert scale in that it provides an ordinal rating but does not give any context as to the distance between the ratings and also leaves room for preferences to be tied (Jamieson, 2004). There is also the problem of "non-differentiators", where a respondent simply answers the same way for every single item (Krosnick & Alwin, 1988). While it is possible to detect these response types and filter them out, it complicates the analysis.

A ranked choice survey is able to accommodate the problem of ties by forcing the respondent to make a choice while directly comparing the options. For example, given a set of five items, a rating system would allow for each to be rated as "3", but in a ranked choice system, they would be put in order from "1" to "5" (Harzing et al., 2009).

We decided to use an experimental design of a weighted, ranked choice survey, similar to a voting system. Rather than place the list of 5 items in the rank order of preference, we provide a pool of points to allocate towards the users' preferred option. In this design, allocating more points would mean they have a higher preference for it. This disallows ties, so users are forced to provide a ranking and solves the problem with ordinal ranks.

Table 3. Examples of what task content consisted off. For a specific group assignment, they would be presented with the description and see the experimental, Experimental2, technical and random options, as well as the option for the assigned group.

Name	Description	Language	Method
Pearson Correlation	Linear relationship between two variables	Experimental	Correlate
		Experimental2	CorrelateLinearly
		Technical	PearsonCorrelation
		Random	copov
		R	cor
		Python	pearsonr
		Matlab	corr
Levene's Test	Check that several groups vary in the same way	Experimental	CompareVariances
		Experimental2	EqualVarianceTest
		Technical	LevenesTest
		Random	xorutu
		R	leveneTest
		Python	levene
		Matlab	Levenetest

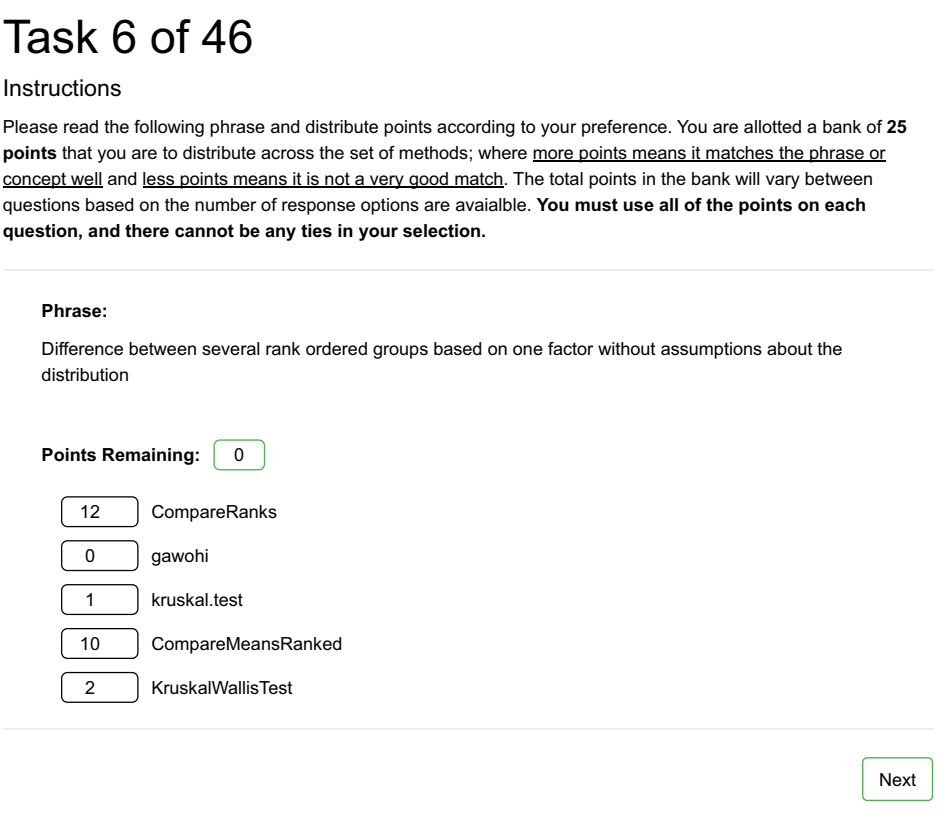

Figure 2. An example of a task that has all of the points filled in. There is a description of how to complete the task, a description of a data science concept, a counter of the remaining points that need to be allocated and 5 options of method names with 12, 0, 1, 10 and 2 points assigned.

For example, if they like the first option, they may allocate the majority of their points to it. Each question had a point pool sized at 5 times the number of choices (e.g. if there are five choices, they had 25 points to assign).

Table 3 contains 2 of the 46 tasks with accompanied method name choices. For brevity, only 2 tasks are shown, the full task list can be found in the replication packet as well as in the supplementary materials. Participants were presented the description and then distributed points amongst the available name options. For example, Figure 2 shows a sample task from the survey in which a participant in the R group was responding to the description of a "Kruskal-Wallis Test". In this example, the responses were 12, 0, 1, 10 and 2; which suggests that they preferred "CompareRanks" and "CompareMeansRanked" as compared to "KruskalWallisTest", "kruskal.test" and "gawohl".

2.6. Random assignment method

Stratified randomization was used to assign participants to their language group (Kang et al., 2008). After entering their experience level (e.g. college year, professional status, etc.), participants of a specific level were assigned to one of the three language groups (Matlab, Python, R) in a randomized round robin fashion. For

example, the first senior to participate would have a 33.3% chance to be assigned to any of the three groups and gets the R condition, the second senior would have a 50% chance to be assigned to the remaining groups and gets the Python condition, the third senior would then be assigned the Matlab condition and the fourth senior would have the same 33.3% chance of any condition as the first. This was done to ensure that within each experience level, there was an even distribution of each group assignment.

Additionally, for each participant, the 46 tasks were present in randomized order and within each specific task the available options were presented in randomized order. This was done to account for any learning effect between tasks and to try to keep the participant engaged in thinking critically about the available choices.

2.7. Statistical methods

The primary method used to compare groups on primary outcomes was a two-way ANOVA was performed to evaluate the effects of Language Group (Matlab, Python, R) and Method Name (Experimental, Experimental2, Existing, Technical, Random) on normalized preference ratings, and partial eta squared (η_p^2) indicated as effect size. After the study was completed and we had also built a statistical package based on our findings, we ran the statistical results in both the programming language R and in our new library which makes use of the nomenclature choices and organizational taxonomy found while creating this survey.

2.8. Data diagnostics

There were three types of post-data-collection exclusion of participants. There were 124 participants in total and all were hand-checked for problems. The first type of exclusion was for participants that did not complete the experiment. There were 4 participants that dropped out after signing the informed consent either partway through the demographic survey or within the first 4 questions. The second type was upon first inspection of the data, there was one participant that had some of some of their data mangled by an issue with the web server. Their data was unusable. The third type was for one participant that was found to have provided the exact same pattern of rankings for each task (e.g. a point allocation of 19, 3, 2, 1, 0 from top to bottom). This was detected by finding that the randomized word was allocated 19 points far more than any other participant. After these exclusions, we were left with a sample size of 118 participants.

2.9. Analytic strategy

To protect against family-wise error in the analysis of our primary outcomes, we will conduct a pairwise t-test (a test comparing all the pairs of means) with a Bonferroni correction (that accounts for a type of error called family-wise error).

3. Results

3.1. Statistics and data analysis

There were 46 total questions, 118 participants and a total of 27,052 responses. The responses by method name category for the Matlab group ($n = 40$) were Existing, Experimental, Experimental2, Technical and Random ($n = 1835$), where one participant did not respond to one task ("Critical Value (F)"). The responses by method name category for the Python group ($n = 38$) were Existing ($n = 1705$), Experimental, Experimental2, Technical and Random ($n = 1743$), where Existing had one task without an available method name ("Test of equal or given proportions") and one participant did not respond to five tasks ("ANCOVA", "Binomial Test", "Huynh-Feldt Correction", "T Distribtuion" and "Z-Score"). The responses by method name category for the R group ($n = 40$) were Existing, Experimental, Experimental2, Technical and Random ($n = 1840$). For the Matlab group, considering the method names by category, Experimental had the highest mean ($M = .31$, SD = .2), followed by Experimental2 ($M = .26$, SD = .19), Technical ($M = .22$, SD = .18), Existing ($M = .15$, SD = .14) and Random ($M = .06$, SD = .1). For the Python group, considering the method names by category, Experimental had the highest mean ($M = .32$, SD = .19), followed by Experimental2 ($M = .27$, SD = .18), Technical ($M = .22$, SD = .17), Existing ($M = .14$, SD = .14) and Random ($M = .06$, SD = .08). And finally, for the R group, considering the method names by category, Experimental had the highest mean ($M = .3$, SD = .18), followed by Experimental2 ($M = .26$, SD = .17), Technical ($M = .23$, SD = .16), Existing ($M = .16$, SD = .14) and Random ($M = .05$, SD = .08). Table 4 shows the mean and standard deviations in a more concise format and Figure 3 shows a boxplot representation of preference ratings separated by language group and method name categories.

A two-way ANOVA was performed to analyze the effect of Language Group (i.e. the between-subjects independent variable of which existing language method would be presented) and Method Name (i.e. the categorized options as Experimental, Experimental2, Existing, Technical and Random) on normalized preference rating. This two-way ANOVA revealed that there was a statistically significant interaction between the effects of Language Group and Method Name ($F(8, 27037) = 6.22$, $p < .001$)($\eta_p^2 = .002$). An effect size of this magnitude is extremely small.

Simple main effect analysis showed that Language Group did not have a statistically significant effect on normalized preference rating ($F(2, 27037) = .05$, $p = .95$)($\eta_p^2 = .001$). This really just means that the experimental group, like getting the R name instead of the Matlab one, had no impact on how a participant rated the other items. For the other main

Table 4. The mean and standard deviation of the normalized method name preference ratings separated by language group and categorized method names.

	Matlab $\bar{x}(s)$	Python $\bar{x}(s)$	R $\bar{x}(s)$
Experimental	.31(.2)	.32(.19)	.3(.18)
Experimental2	.26(.19)	.27(.18)	.26(.17)
Existing	.15(.14)	.14(.14)	.16(.14)
Random	.06(.1)	.06(.08)	.05(.08)
Technical	.22(.18)	.22(.17)	.23(.16)

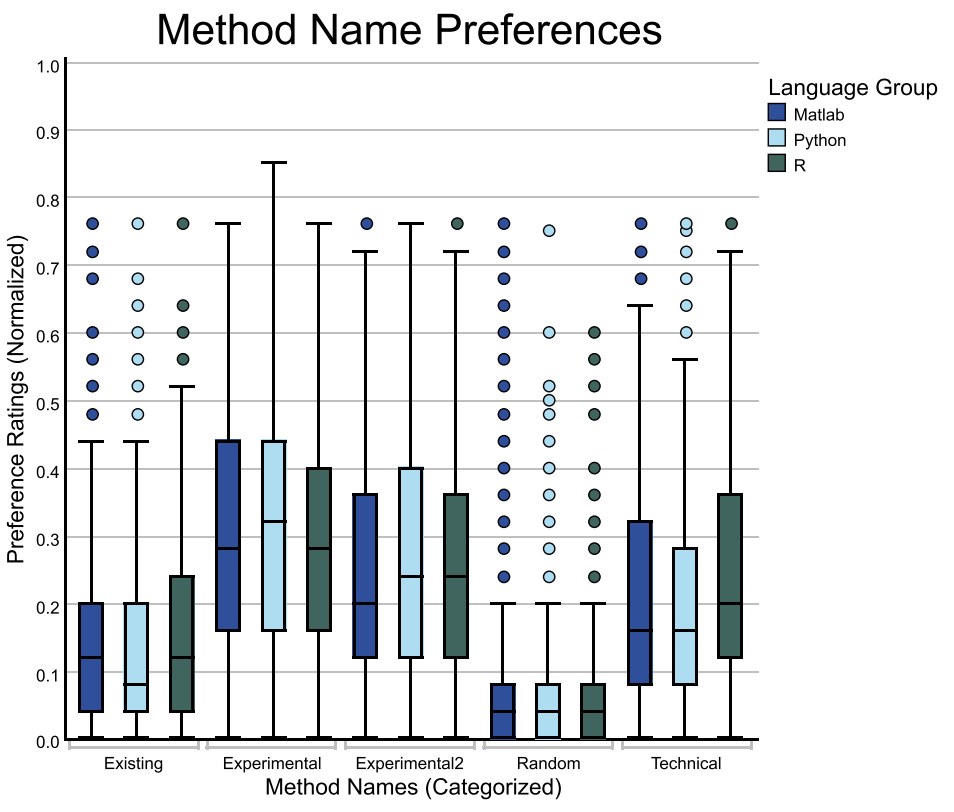

Figure 3. A boxplot comparing normalized method name preference ratings separated by language group and categorized method names.

Table 5. Pairwise t-tests of the main effect language group with a Bonferroni correction. Statistically significant differences are denoted with asterisk (*).

	Matlab	Python
Python	1.00	-
R	1.00	1.00

Table 6. Pairwise t-tests of the main effect method names with a Bonferroni correction. Statistically significant differences are denoted with asterisk (*).

	Experimental	Experimental2	Existing	Random
Experimental2	<.001*	–	–	–
Existing	<.001*	<.001*	–	–
Random	<.001*	<.001*	<.001*	–
Technical	<.001*	<.001*	<.001*	<.001*

effect analysis showed that Method Name did have a statistically significant effect on normalized preference rating ($F(4, 27037) = 2211.23$, $p < .001$)($\eta_p^2 = .247$).

Next, a pairwise comparison using t-tests was conducted, with a Bonferroni correction to provide a conservative adjustment for family-wise error. The p-values for all of the pairwise comparisons can be found in Tables 5 and 6 with the statistically significant differences at significance level $\alpha = 0.05$ denoted with an asterisk (*).

Lastly, to investigate whether self-reported general programming experience, statistics experience, Matlab, Python or R experience had an impact on our results, we separately ran an ANCOVA using 5 covariates. In this alternative model, each covariate was non-significant with all p-values >0.999 and effect sizes near zero. Our conclusion from this is that if such an effect exists, which it might on a different sample of people, we found no evidence of it in our sample. In this alternative model, all other p-values and effect sizes remain nearly identical to the previous model.

4. Discussion

4.1. Interpretation

With regards to **RQ1**, based on the results of the two-way ANOVA, participants with minimal statistical background did have a preference in what method names for data science related operations broadly. Whether the single language word included came from Matlab, Python or R appeared to make no difference in the overall result. The results were statistically significant for Method Name ($p < .001$) with an effect size of ($\eta_p^2 = .247$). Overall, there was clear preference between Experimental, Experimental2 and Technical words. Intuitively, this makes sense. Names designed to state what the test does or is tend to be preferred.

Some comments from the post-survey feedback follows this same logic. One participant reported "some of [the options] were hard to understand, mainly the abbreviations". And followed up with "The ones that were maybe one or two general words made more sense to me, [but] the overly long syntax just confused me overall". Similar sentiments were expressed by several participants. When they refer to abbreviations, this could be referring to the randomized words or existing method calls like "chi2gof" or "fpdf" (used to for "Chi-Square Goodness of Fit" and "F Distribtuion" respectively). Their comments largely matched their ratings. The comment about overly long syntax should be kept in mind, as there were several cases where the concept was complex and needed more words to properly articulate the method's use. This also highlights that word choice studies alone do not, and cannot, provide enough guidance to build a statistical library in practice. They can guide, but they cannot prescribe.

For **RQ2**, the results of the two-way ANOVA showed that there was no statistically significant difference between the Language Groups ($p = .95$), and, while there was a statistically significant difference between the interaction effect of Language Group and Method Name ($p < .001$), the effect size accounted for less than 0.2% of the variance ($\eta_p^2 = .002$). Essentially, while statistically significant, it is of minimal practical significance. While this was so, because it was significant, it led us to consider why it might exist at all. To investigate, we ran a pairwise comparison using t-tests with a Bonferroni correction on each individual concept for each of the language groups, and found that for the Matlab,

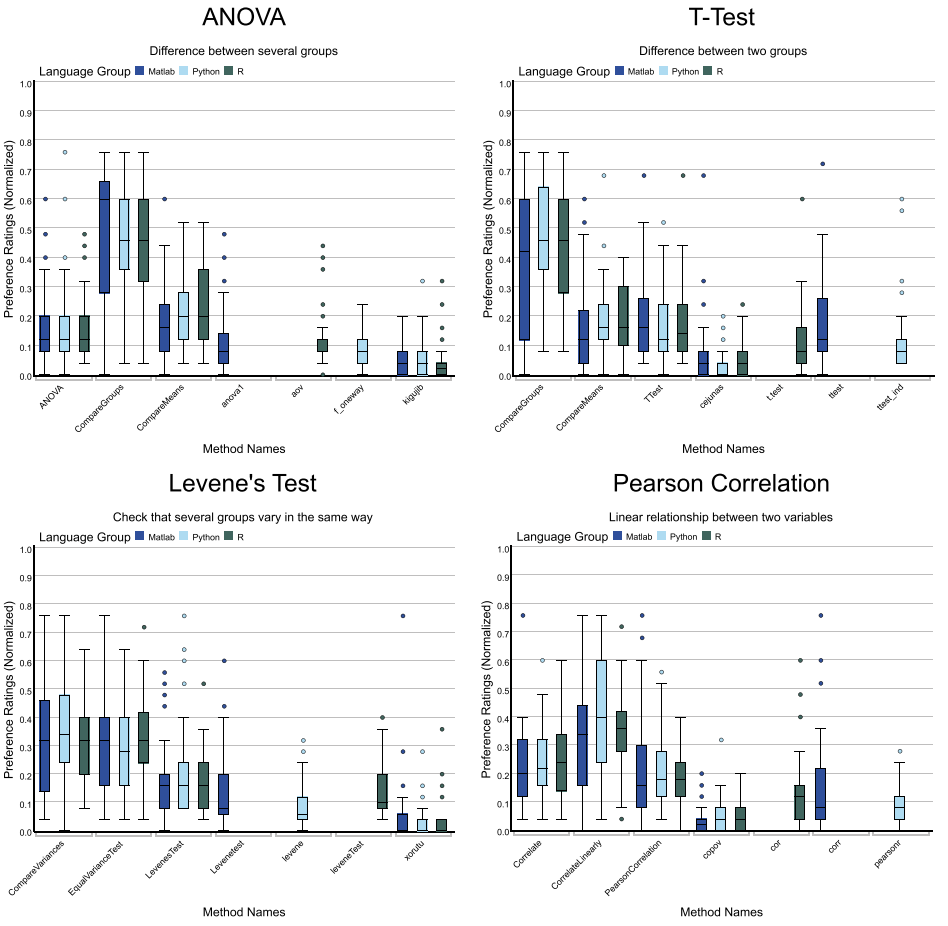

Figure 4. Four boxplots showing a comparison of method name choices to normalized preference ratings for 4 of the 46 concepts tested (ANOVA, t-test, Levene's test and Pearson correlation).

Python and R groups the existing method name was not statistically significantly different from the random option in 63.0%, 62.2% and 30.4% of concepts respectively. Put another way, the ratings were not much different, but R did just a bit better than random words.

Lastly, for **RQ3**, in most cases, the top choice is the same across all three group assignments; one of the two experimental names which provide a concise set of words that match the concept rank at the top. Figure 4 shows a few examples of participant responses on a per concept basis ("ANOVA", "T-Test", "Levene's Test" and "Pearson Correlation"). While the experimental names were typically the top choices, the technical name with the same formatting style ranked well in some cases. For example, the method name rankings for "T Distribution" in the Matlab group were "HeavyTailNormalDistribution" ($M = .33$, $SD = .16$), "FatTailDistribution" ($M = .26$, $SD = .15$), "TDistribution" ($M = .23$, $SD = .16$), "dt" ($M = .13$, $SD = .15$) and "wukeb" ($M = .04$, $SD = .06$). While the experimental method names were ranked first and second, the technical name came in a close third. It is hard to say why exactly, but we hypothesize the word distribution made sense to people and they may have plausibly just ignored the

letter T. The reader of this paper may not even be aware of how T varies from F or other distributions, as it is not obvious unless one knows the math. If we were to attempt to describe the T Distribution in simple terms, it is characterized by a symmetric bell-shaped curve with a shorter peak and heavier tails; as compared to an F Distribution which could be described as a representation of the ratio between two variances.

4.2. Threats to validity

It is important to note that some threats to validity were discovered during post-data-collection analysis. In particular, it is difficult to recommend a set of rankings on a per concept basis due to how the experimental method names were generated. The first round of method names were generated based on the phrases that were presented. This introduced some possible bias; for example, the phrase presented for "T-Test" was "Difference between two groups", the Experimental method name was "CompareGroups" and the Experimental2 method name was "CompareMeans". These were ranked 1 and 2 respectively for both the Python and R groups and 1 and 4, respectively, for the Matlab group, with "CompareGroups" being rank 1 across the board. While it seems evident that participants have a preference for plain English, full words with camel-case formatting, "CompareGroups" may have just been the best option for the given phrase due to it also having the word "group". The use of the word group was accidental in the description, despite conducting several pilots before data collection. We point it out, though, because others replicating our work should likely change the wording and document the difference. Another example from a different category was for the "Repeated Measures ANOVA", which used the phrase "Difference between several groups when there are repeated measures" and had the Technical option, "RepeatedMeasuresANOVA" ranked highest for each group while both experimental options, "CompareGroups" and "CompareMeans" lacked the "repeated measures" terminology. Our point is, even how to describe these tests to people in a completely fair way is not obvious and will need more experimentation.

Next, looking at Figure 3, it is apparent that there are some outliers. Note that the total responses for each of those plots ranges from 1705 to 1840 data points. The reason that there appear to be so many outliers for the Existing, Random and, to some degree, Technical categories is simply because the majority of responses were quite low for those categories. Without being able to reach out and get further feedback from individuals, we can only make guesses at why they would rate certain method names high comparatively to the popular rating for that particular name. We were able to filter out some problematic data, as discussed in Section 2.8, but these outliers could potentially be due to randomly assigned task responses or a true preference for a method name that does not follow the trend of the rest of the participants and thus we did not feel that we could remove them arbitrarily. Instead, we rely on gathering a larger sample size and the robustness of the statistical tests to handle the outliers that remain, but it is worth noting this issue so it can be taken into consideration.

Another potential threat to validity lies in which library or package name was used within an existing language. For example, we used "pearsonr" from the scipy library as the Python example to go along with the description for a "Pearson Correlation", but there are several other libraries and packages with other method name choices that could be used

to run the same statistical test. There is a threat to validity in that each of our participants could only be familiar with one of the alternative libraries (e.g. numpy, pandas) and the associated alternative method name choice within said library (e.g. "corrcoef", "corr"), and thus rated what was available to be low due to not have familiarity with that particular method name. While we elected to use just one method name choice that was a valid, compilable option as the Existing option, we failed to collect demographic information on how much experience individuals had in each library or package which could have provided some insight into if this was an issue.

Next, while we did gather some post-experiment feedback based around their opinion of the word choices, descriptions and experiment overall, we did not gather any metrics or ask direct questions about the level of cognitive load or anxiety induced while participating in the experiment. It stands to reason that with a large set of tasks, many choices to make and the potential lack of statistical knowledge; some of the participants may have felt overwhelmed and this had the potential to impact their responses. Unfortunately, we did not gather any specific metrics to measure their cognitive load or anxiety, whether that be done broadly at the start and end of the study or on a task-by-task basis, it is definitely something that we will consider looking into with future research.

Lastly, there were several cases with duplicate options presented to the participant. This was somewhat of an oversight that did not come up in piloting, as the pilot participants did not mention this in the post feedback survey. For example, a participant with the Matlab or Python group assignment, when presented with the definition of "Z-Score", would have had the option of "ZScore" as the Technical name using similar naming convention formatting as our experimental names, but the Existing option would have been "ZScore" for both Matlab and Python. In the post feedback survey, participants were able to comment about the study and the tasks to explain their thought process or any issues they had. Several participants reported that they were confused when this situation came up and simply ranked the duplicate method names similarly, but due to the random ordering and requirement of disallowing ties, there was forced weighted rankings between these values options and some participant may have elected to rank one of those option highly and not provide a high rank to the second instance of that choice. Our expectation was that while the choices are nearly identical, the difference in naming convention would sway their preference towards one or the other even if the ratings were very close. Unfortunately, this was not the gut reaction that some participants expressed in the post survey feedback.

4.3. Limitations

Naming conventions used in statistics in the real-world, where experts have a variety of tasks to do (e.g. analyze data, think through complex problems, reporting), are only one small part of a bigger picture. As such, our study hardly represents some kind of general purpose test across any data science or statistics tasks. For example, beyond naming, there are other considerations in organizing a scientific computing library. Examples at least include the parameter list, order of operations in code, how to report out what a test means (e.g. F-values, p-values), among many other factors. While this study does seem to show that our *say what you do* style philosophy is plausible, we did not evaluate how these choices impact program comprehension, writing code, recollection, or

interpretation. We are thinking about names and organization as a guide, but clearly that is one small part of a bigger picture when it comes to re-thinking statistical analysis in general.

Additionally, the sample consisted exclusively of college students enrolled in Computer Science courses and primarily majoring in Computer Science. The target population of people that uses statistics, in the general sense, is much broader than this. Other degree fields that make use of these concepts (e.g. statistics, psychology, education, law, other scientific fields) may have a more or perhaps less foundational education in statistics compared to these students. As such, tests like ours should be interpreted cautiously, as many factors could influence these ratings (e.g. age, gender, disability, expertise). Further, this study had an inclusion criteria that participants must be 18 years of age or older and no other specific exclusion criteria. While a college-aged participant with little statistical training can provide their views on what could be beneficial to lowering the barrier for entry for novice data science programmers, much could potentially be learned through similar tests with middle or high school aged students. Finally, one large motivator for including an easy to use replication packet is that we hope other scholars will be open to thinking about the broader issue here: statistics libraries today are *arguably* esoteric, difficult to understand, and cause anxiety (Rode & Ringel, 2019b; Vigil-Colet et al., 2008). As scholars, even selfishly, we may be able to make our own work, and this paper is no exception, easier to think about with a bit of careful redesign.

In regard to the naming conventions themselves, while we are investigating a few options for each statistical concept and have gathered some evidence on specific name choices, we are hardly suggesting that there is one word to rule them all (Stefik & Hanenberg, 2014). If anything, we included two experimental names to investigate partially whether small differences, even if both were arguable saying what they do, would make much of a difference. We found that our two options were quite comparable. While we have provided them in this fashion, there could be a plethora of others where some perform slightly better or worse depending on the background of the sample and other factors. The implication of this study is thus that there is evidence that novice data science users do have a preference for names that say what they do, but this does not mean there is a single name that is always best for a particular test.

While we are not suggesting there should be one word to rule them all, we also think it is important to consider the status quo. We suspect software developers, perhaps with no training in psychology or human factors, just guessed how to design these libraries, which seems rational given how inconsistent they are. Just as it is important to at least try and choose your words carefully in an academic paper, words matter in statistics as well and we think there may be societal benefits by making statistics libraries less esoteric and more intuitive. So, for example, while it stands to reason that there may be communities that have different naming convention preferences, scientific research should be done evaluating where and to whom. Instead of just letting designers of these libraries guess, where in some cases the words chosen appear to do little better than random words, we should be careful and say what we mean in the design of statistics libraries. Put another way, the primary goal of this line of research is to try and evaluate from first principles what it might look like to use statistics libraries that were meaningfully organized, named to imply what everything does, and structured to present the information that is most

important to the kinds of people using them. Given data science is increasingly being taught, at least in the U.S., in high school or younger settings, it feels salient to consider the humans that ultimately must use these tools.

4.4. Implications

We recognize that today, one may also be able to search for statistical terms, or use large language models like ChatGPT, to garner more information about data analysis. These tools may provide benefits in figuring out how to write or structure an experiment or statistical analysis. Such tools are hardly perfect, however, and today there is still significant infrastructure in many academic departments where we teach such methods. For example, just because ChatGPT can regurgitate statistical procedures or equations does not mean statistics departments or research methods courses are going away anytime soon. We see the work here not in competition to such methods, but complementary. Perhaps such terminology can help organize our students' thinking about statistics, an area chronically known to cause anxiety (Macher et al., 2012; Onwuegbuzie & Wilson, 2003; Rode & Ringel, 2019a). This paper also, however, serves as a ponderance: is it necessary for students around the world to memorize a series of mathematician's names, and single letter variables, or could we come up with a system that is more organized? It might be that search and language model tools can be part of the picture, but it seems unlikely they will entirely replace data science done by humans anytime soon.

The findings presented here suggest that what is currently available to use today, and is commonly used in both a learning and professional environment, use naming choices that are not preferred by novice users when presented with alternatives. In general, the results show that novice users preferred method names which followed a consistent naming convention and used a few concise, easy-to-read words that described what the method does, rather than a set of abbreviations or names. While these results do not investigate the preferences of data science or statistics experts that have had the training and knowledge of the technical test names, it does show that there could be a beneficial alternative to the nomenclature used to help new data science students.

While these results on user preferences are a step towards organizing scientific computing at the top level, this is only the first step towards investigating a reorganization of scientific computing. This survey is a useful tool in finding evidence towards what naming convention is preferred and makes more sense to novice data science users and have used those user preferences and grouped similar tests in a reorganized, uniform way of interacting with a statistical library, but future work will involve investigating how these changes impact the use of a language in practice.

As a final note, if a programming language were to adopt and permanently use the naming conventions mentioned for statistics here, the long-term impact is unclear. It could be the case that students feel it is inauthentic. Or, it could feel inauthentic, but still be beneficial. Or, still, it might just make it easier to understand. Similarly, consider flipping the authenticity argument from the lens of fake history. Imagine statistics had instead chosen these names from the beginning (e.g. "CompareMeans", "CompareRelatedRankedMeans"). Then, a new language sprouted that changed them

to authors names or random names (e.g. "T-Test", "Wilcoxon Signed Rank"): would this be better or more authentic? Our paper hardly proves the issue one way or another, but we imagine significant more investigation is needed and our paper is just one investigation into a much larger story.

5. Contribution 2: attempt to put statistics into practice

After completing our survey, we wanted to obtain a "gut check" for how we might apply our naming conventions. We are obviously fully aware that names are just one part of statistics. Further, in looking at previous work, we felt there may be room in the statistical ecosystem to re-think how such systems are put together. After all, there is no hard requirement that we must organize such a system by the names of mathematicians and single letter variables. Academia has simply chosen to do so through convention and tradition over time.

As such, we created our own statistical library in Quorum, an evidence-oriented programming language. Our primary goal was to force ourselves not just to use the names, but to build every statistical test by hand mathematically to ensure we both understood it and that our organization system reflects what we believed was the underlying purpose. The idea here is that our survey provided us some initial feedback, but actually building it flipped our perspective on how such systems should be designed, as it ratchets up the complexity level and forces us to re-think it all – beyond naming.

To further force us to think through usability issues, after completing version 1 of our system, we then wrote documentation for all of the statistical package and placed it online so it could be runnable in a browser or on desktop. Inside this initial prototype library, we implemented a wide variety of statistical tests, including functions for general usage (e.g. loading files), data transformations (e.g. null removal, wide or narrow conversions, filtering), descriptive statistics (e.g. mean, median, mode, skew, kurtosis), a series of kinds of charts (e.g. bar, scatter), and a wide variety of statistical tests (e.g. multi-variate ANOVA, repeated measures, t-tests, Brown-Forsythe, Levene, and a host of univariate and multi-variate corrections). Finally, while not in our survey, we also implemented a library for factor analysis, in part because we were curious about potentially including it in a future survey.

The mathematics behind the tests is nothing short of daunting in many cases and we encountered a series of issues. For example, in an early version, we used Apache Commons to implement the T-Distribution, only to quickly discover that the p-values for the library were incorrect. They were not incorrect by simply a small margin, but by many orders of magnitude. We noticed the issue while specifically building a Tukey HSD test, which requires you consider degrees of freedom values that approach infinity, which is where the problem was. We contacted the team and they fixed the issue after approximately a year of back and forth. Similarly, we found that in many statistical tests, we were either unable to find the original equations published in the literature or textbooks provided oversimplified versions of the equations that are computed differently in practice.

For example, take the Spearman correlation. We noticed quickly that this test was not actually implemented by Spearman (his version does not account for ties). As such, statistical packages typically do a Pearson test with a rank transform. Similarly, for factor

analysis, many of the metrics languages like R output are interesting, but we had to derive our own equations to make them match what the package reported. Finally, for effect sizes, we found different statistical packages either do not agree or do not give the same answer and this matters on a practical level in practice. Different statistical packages also disagreed on the type of sums of squares used for ANOVA and this is not a trivial decision. It can demonstrably change the answer to a research question, depending on the data and the type.

Finally, one limitation of our approach that became all too apparent in building a prototype library was that we needed more evidence on what statistical libraries should output. Academic conventions are terse and bizarre and we suspect that they too, including the very descriptions used in this paper, should be questioned for clarity. In our first version of the library, we copied the style of the R programming language, philosophically, although we find it highly unlikely that this reporting is intuitive. We did this only in the first version to ensure we were getting the same answers as other statistical packages mathematically. Just as an example, APA style uses old conventions, like df, F, t, and p, to represent meaning. Perhaps, with some creativity and careful testing,

```
library(car)
library(lsr)

data = read.csv("data/results2023.csv", header = TRUE, stringsAsFactors = TRUE)

data$normalized_response <- (data$response / data$max)

model = aov(normalized_response ~ group * method_choice, data = data)
Anova(model, type=2)
etaSquared(model)

pairwise.t.test(data$normalized_response, data$group, paired = FALSE, p.adjust.
    method = "bonferroni")
pairwise.t.test(data$normalized_response, data$method_choice, paired = FALSE, p.
    adjust.method = "bonferroni")
```

Figure 5. A sample of R code used to run the two-way ANOVA and pairwise comparison using t-tests with a Bonferroni correction in the results section.

```
use Libraries.Compute.Statistics.DataFrame
use Libraries.Compute.Statistics.Tests.CompareMeans
use Libraries.Compute.Statistics.Tests.CompareMeansPairwise

DataFrame frame
frame:Load("data/results2023.csv")

frame:AddColumn("normalized_response", "response / max")
frame:SetSelectedColumns("normalized_response")
frame:SetSelectedFactors("group,method_choice")

CompareMeans compareMeans = frame:CompareMeans()
output compareMeans:GetFormalSummary()

CompareMeansPairwise compareMeansPairwise = frame:CompareMeansPairwise()
output compareMeansPairwise:GetPairwiseSummary()
```

Figure 6. A sample of our statistics library code used to run the two-way ANOVA and pairwise comparison using t-tests with a Bonferroni correction in the results section.

we can derive conventions that are more meaningful to the people that need to read them.

To help the reader understand the benefits and limitations of our contribution, including the limitations of naming, we thus provide two ways to replicate the raw statistics in this work. We first provide a script in R that will execute statistics on the anonymized raw data. Second, we provide a script in our tool. Figure 5 and Figure 6 show the source code for both scripts for one of the tests in this paper. Charts and other systems are similarly doable in both, but are not listed. These libraries are written in different programming languages, have a different style, and have different words. Our study did not encapsulate whether the new style was actually helpful to people, as we needed to organize and then create the library first before we can begin such testing, but we offer these examples as a gut check for one possible approach to how these names could be used. We imagine, given only naming data, that this is only the beginning of the debate and that reasonable people might come to different conclusions. Even in our own case, while the study was informative, when actually implementing we made a series of executive decisions on naming that can be brought into doubt in future replications.

Our hunch is that if the community were to create a statistical analysis library really designed to be easy to use and understand, it could plausibly help a group larger than "just" academics. However, our study was scoped to investigate novice data science programmers and gathered evidence only for those with a Computer Science background and minimal statistical training. Thus, we created a replication system where people can, hopefully, easily modify the survey that we have created here and run it on different kinds of people, with different questions, or for different goals. We discuss this replication package next and we call for others to help us make statistics easier to understand and use.

5.1. Contribution 3: future research

Our research here suffers from at least two limitations that we think would benefit from participation by the broader community. The first, already described, is that names alone are not sufficient to actually build a statistical library in practice. The second, described here, is that our study realistically needs a wider demographic to know where and to whom the evidence applies. For example, it could be the case that undergraduates generalize, but broadly it seems implausible. Middle- or high-school aged students, undergraduate computer scientists, and trained statisticians are all just very different people. We think that collecting data in a community organized and a consistent manner, where we track demographics, could have value in narrowing down how best to describe statistical concepts. Along that same line of reasoning, we elected to use a variety of existing programming languages in this experiment rather than selecting just one. This was to give existing languages a fair shot, and when replicating this with other groups, it is possible that they have more experience or an affinity towards a particular existing language over another group with a different background.

Thus, after running our study, we first conducted a round of revisions to the concept descriptions. This fixes issues we discovered in piloting related to overlapping words with any of the method name options. Second, we dockerized our replication packet and will make it publicly available. The survey is built as a self-hosted experiment website under

version control on Github; this provides the flexibility to customize our experiments how we see fit, such as customizing randomization methods or using less common question formats like a weighted, ranked choice survey. In addition to this, it simplifies the process of replication by creating a centralized location for submitting replications for code review and keeping everything tracked under version control which promotes transparency. Finally, while there are many commercial software solutions for creating a survey, we found this to be a simple way to bundle the entire study process from data collection to data analysis into one distributed package, and ideally this same process can be used in future research, building on this dockerized tech stack to include programming tasks or other methodology design, which are not quite as easily fit into a commercial survey host's format, with the same benefits seen here of being able to spin up the website, collect data and analyze the results. This section describes how to use our dockerized system to run a separate version of the study. We encourage other research groups to change the questions, get groups of different demographics, but also to keep some aspects of the survey constant so others can compare. This approach, to vary some pieces and keep others the same, may allow us to sort out issues like generalizability.

Our replication packet uses a web stack consisting of NGINX, MySQL and PHP. To spin up your own instance of the web-based survey, first you must clone the repository from https://github.com/qorf/DataScience. Next, you must make a copy of the file containing the environment variables ("cp env-dev.env") and update it with your preferred passwords and ports. Finally, you can spin up the docker instance by running the command ("docker-compose up -d") after installing Docker Desktop. The URL to begin the experiment will be [baseurl]/nomenclature, so for the local instance you would navigate to localhost/nomenclature. Further steps can be taken to include self-signed certificates, and details can be found in the repository's README file. Thus, to conduct an identical replication with no changes to the existing experimental protocol or tasks on a new sample, it will only require a few simple steps to get everything up and running.

By default, the docker-compose command will use the.sql file found in the/sqlinit directory to initialize the database. The tasks in this web-based survey are database-driven and all tasks are included. You can add additional tasks to the experiment by inserting into the TEST and METHOD tables. Each task is made up of a TEST entry and all of the method name options are derived from METHOD entries that are joined by the TEST. test_id primary key. To add your own additional statistical tests, you can insert them to the TEST table (the name of the concept and a description) and then use the generated ID and insert your method name choices in the METHOD table. You could also add additional method choices to existing tasks or alter descriptions in a similar fashion. All tasks, when presented to a participant, are randomized in the order they are presented from all possible tasks in the TEST table, and the method name choices per task have a randomized order as well. An example of how to do this can be found in Figure 7. Here, we have inserted a new concept and phrase into the TEST table, found the test_id that was generated and then inserted a few method choices into the METHOD table.

While the benefits of replication are generally understood, our purpose here is that we suspect that no single research group could quickly obtain information from all of the types of people and questions that might be needed to really understand how to describe an area as complex as statistics. Further, even if it could, there is an existing social contract around statistics. We simply accept today that if we say "Bonferroni" or "F" or "Generalized

```
INSERT INTO TEST (concept, phrase)
VALUES ('New Test', 'A test phrase that will be presented to the user');

SELECT test_id  FROM TEST WHERE concept = 'New Test';

INSERT INTO METHOD (test_id, name, language)
VALUES ([returned test_id], 'PresentToUser', 'Experimental'),
       ([returned test_id], 'ptu', 'R'),
       ([returned test_id], 'pres2user', 'Python'),
       ([returned test_id], 'present_to_user', 'Matlab');
```

Figure 7. An example of how to add a test and set of methods to the list of available tasks on the web-based survey.

Eta" that the reader just has to go look that up and hope they can figure out what it means. Perhaps with community participation and study, we can improve the situation and make it less esoteric. For example, the broad pattern we derived from this study to CompareMeans (t-test or ANOVA) or CompareVariances (Levene), or Correlate (Pearson) vs CorrelateRanks (Spearman) feels more intuitive, but various demographics may disagree.

Our hunch for the community to consider is that we can target sampling in a variety of groups that use data science and statistical languages, in addition to others using data science today. For example, high-school students, policy makers, medical patients, or many others may benefit from less esoteric reporting and as such, we should conduct tests with them as participants. A recent example of this can be found in Lappi et al.'s replication study, which took a study designed to find evidence-based research on programming language syntax intuitiveness and replicated it with non-native (predominately in Finland) English speakers to see how the results differ (Lappi et al., 2023). In that case, the original result generalized, but there is no guarantee that will be true across the board.

Replication studies are fairly common in other STEM fields resulting in the possibility of meta-analyses of bodies of research by inspecting multiple formal, quantitative studies and providing a broader understanding of the results. There has been a recent push to formalize the steps that could make meta-analysis possible in Computer Science education research by normalizing replication studies and establishing guidelines for registered reports (Brown et al., 2022). Similar efforts are being made in Software Engineering research with special issues and guidelines being setup in several journals and conferences in the space (Ernst & Baldassarre, 2023). While this study did not include a registered report, we have made an effort to follow a set of reporting guidelines; using the American Psychology Association Journal Article Reporting Standards (APA JARS) (American Psychology Association, n.d.) as a starting point, and provide the means for the study to be replicated through the full research pipeline from data collection to data analysis.

6. Conclusion

In this paper, we have presented a randomized controlled trial with 118 participants in which we measured weighted, ranked choice preferences for method name choices of statistical tests and data science related operations. In particular, we compared three existing languages' (Matlab, Python and R) naming choices, two options we derived by

using peer reviewed literature and empirical evidence as a guideline, the technical name for the concept using the same naming convention and a randomly generated control choice. Our derivations were based around the benefits of using names that map more intuitively to the concept they are used for.

Our findings suggest that novice data science users prefer a consistent naming convention and that uses a few concise, easy-to-read words that describe what the method does, as compared to existing scientific computing libraries which largely make use of abbreviated letters and mathematicians' names. We found that these names, currently used in practice, largely were not even preferred, at a statistically significant level, more than the randomized control words by novices when presented with alternatives. When actually building a statistics package based on these findings, we encountered significant challenges, but were able to create a library based on these organizing principles. These principles can be organized into real tools and languages. Lastly, we have provided a replication package and call on the broader community to replicate the work in their own context to try and help all of us make statistical libraries that are, perhaps, a bit easier to understand for our students (and ourselves).

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the National Science Foundation under [Grant numbers 2048356 and 2106392].

ORCID

Timothy Kluthe http://orcid.org/0009-0005-7856-7909
Hannah Stabler http://orcid.org/0000-0001-7254-0856
Amelia McNamara http://orcid.org/0000-0003-4916-2433
Andreas Stefik http://orcid.org/0000-0002-2554-5862

References

American Psychological Association. (n.d.). https://apastyle.apa.org/jars

Anderson, J. R. (1995). *Cognitive psychology and its implications* (W.H. Freeman).

Bartlett, F. (1932). *Remembering: A study in experimental and social psychology Cambridge University press*. Cambridge (England) and Melbourne (Australia).

Becker, B. A., Denny, P., Pettit, R., Bouchard, D., Bouvier, D. J., Harrington, B. & Prather, J. (2019). Compiler error messages considered unhelpful: The landscape of text-based programming error message research. *Proceedings of the working group reports on innovation and technology in computer science education* (pp. 177–210). Association for Computing Machinery. New York, NY, USA. https://doi.org/10.1145/3344429.3372508.

Binkley, D., Davis, M., Lawrie, D., Maletic, J., Morrell, C., & Sharif, B. (2013, April). The impact of identifier style on effort and comprehension. *Empirical Software Engineering*, *18*(2), 219–276. https://doi.org/10.1007/s10664-012-9201-4

Bower, G. H. (1972). Mental Imagery and Associative Learning L. W. Gregg. *Cognition in Learning and Memory*, (Wiley), 51–88.

Brown, N. C. C., Marinus, E., & Hubbard Cheuoua, A. (2022). Launching registered report replications in computer science education research. *Proceedings of the 2022 acm conference on international computing education research - volume 1* (pp. 309–322). Association for Computing Machinery. New York, NY, USA. https://doi.org/10.1145/3501385.3543971.

Burke, A., Okrent, A., & Hale, K. (2022, January). *The state of U.S. Science and engineering 2022*. National Science Foundation | National Science Board. https://ncses.nsf.gov/pubs/nsb20221/u-s-and-global-research-and-development

Chattopadhyay, S., Prasad, I., Henley, A. Z., Sarma, A., & Barik, T. (2020). What's wrong with computational notebooks? pain points, needs, and design opportunities. *Proceedings of the 2020 chi conference on human factors in computing systems* (pp. 1–12). Association for Computing Machinery. New York, NY, USA. https://doi.org/10.1145/3313831.3376729.

Deese, J. (1959). On the prediction of occurrence of particular verbal intrusions in immediate recall. *Journal of Experimental Psychology*, *58*(1), 17. https://doi.org/10.1037/h0046671

Deissenboeck, F., & Pizka, M. (2006, September). Concise and consistent naming. *Software Quality Journal*, *14*(3), 261–282. https://doi.org/10.1007/s11219-006-9219-1

Easing the burden of code review. (2018).

Ernst, N. A., & Baldassarre, M. T. (2023, March 11). Registered reports in software engineering. *Empirical Software Engineering*, *28*(2), 55. https://doi.org/10.1007/s10664-022-10277-5

Godfrey, A. J. R., Murrell, P., & Sorge, V. (2018). An accessible interaction model for data visualisation in statistics. In K. Miesenberger & G. Kouroupetroglou (Eds.), *Computers helping people with special needs* (pp. 590–597). Springer International Publishing.

Harzing, A.-W., Baldueza, J., Barner-Rasmussen, W., Barzantny, C., Canabal, A., Davila, A. & Zander, L. (2009). Rating versus ranking: What is the best way to reduce response and language bias in cross-national research? *International Business Review*, *18*(4), 417–432. http://www.sciencedirect.com/science/article/pii/S0969593109000353

Jamieson, S. (2004). Likert scales: How to (ab)use them. *Medical Education*, *38*(12), 1217–1218. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1365-2929.2004.02012.x

Kaijanaho, A.-J. (2015). *Evidence-based programming language design : A philosophical and methodological exploration* (University of Jyväskylä).

Kang, M., Ragan, B. G., & Park, J.-H. (2008). Issues in outcomes research: An overview of randomization techniques for clinical trials. *Journal of Athletic Training*, *43*(2), 215–221. https://doi.org/10.4085/1062-6050-43.2.215

Kapur, S., Craik, F., Tulving, E., Wilson, A., Houle, S., & Brown, G. (1994, April). Neuroanatomical correlates of encoding in episodic memory: Levels of processing effect. *Proceedings of the National Academy of Sciences of the United States of America*. United States (Vol. *91*). pp. 2008–2011.

Krosnick, J. A., & Alwin, D. F. (1988). A test of the form-resistant correlation hypothesis: Ratings, rankings, and the measurement of values. *Public Opinion Quarterly*, *52*(4), 526–538. http://www.jstor.org/stable/2749259

Kross, S., Peng, R. D., Caffo, B. S., Gooding, I., & Leek, J. T. (2020). The democratization of data science education. *American Statistician*, *74*(1), 1–7. https://doi.org/10.1080/00031305.2019.1668849

Lappi, V., Tirronen, V., & Itkonen, J. (2023). A replication study on the intuitiveness of programming language syntax. *Software Quality Journal*, *31*(4), 1211–1240. https://doi.org/10.1007/s11219-023-09631-7

Lawrie, D., Morrell, C., Feild, H., & Binkley, D. (2006, June). What's in a name? A study of identifiers. *14th IEEE International Conference on Program Comprehension (ICPC'06)* Athens, Greece (pp. 3–12).

Macher, D., Paechter, M., Papousek, I., & Ruggeri, K. (2012). Statistics anxiety, trait anxiety, learning behavior, and academic performance. *European Journal of Psychology of Education*, *27*(4), 483–498. https://doi.org/10.1007/s10212-011-0090-5

Noble, W. S., & Lewitter, F. (2009, July). A quick Guide to organizing computational biology projects. *PLoS computational biology*, *5*(7), e1000424. https://doi.org/10.1371/journal.pcbi.1000424

O'Connor, G. (2014). Moore's law gives way to bezos's law. *GigaOm.com* https://old.gigaom.com/2014/04/19/moores-law-gives-way-to-bezoss-law/

Onwuegbuzie, A. J., & Wilson, V. A. (2003). Statistics anxiety: Nature, etiology, antecedents, effects, and treatments: A comprehensive review of the literature. *Teaching in Higher Education*, *8*(2), 195–209. https://doi.org/10.1080/1356251032000052447

Pardilla-Delgado, E., & Payne, J. (2017, January). The Deese-roediger-McDermott (drm) task: A simple cognitive paradigm to investigate false memories in the laboratory. *Journal of Visualized Experiments*, *2017*(119). https://doi.org/10.3791/54793

Rafalski, T., Uesbeck, P. M., Panks-Meloney, C., Daleiden, P., Allee, W., Mcnamara, A., & Stefik, A. (2019, September 23). A randomized controlled trial on the Wild Wild West of scientific computing with Student learners. *Proceedings of the 2019 ACM Conference on International Computing Education Research - ICER '19* (pp. 239–247). ACM Press. http://dl.acm.org/citation.cfm?doid=3291279.3339421

Rendulic, P., & Terrell, S. (2012). Anxiety toward statistics and the use of computers: A study of graduate level education majors. *Journal of Computing in Higher Education*, *11*(2), 104–120. https://doi.org/10.1007/BF02940893

The Rockefeller Foundation. (2023, February). https://data.org/organizations/data-science-4-everyone/

Rode, J. B., & Ringel, M. M. (2019a). Statistical software output in the classroom: A comparison of R and SPSS. *Teaching of Psychology*, *46*(4). 319–327. https://doi.org/10.1177/0098628319872605

Rode, J. B., & Ringel, M. M. (2019b, October). Statistical software output in the classroom: A comparison of R and SPSS. *Teaching of Psychology*, *46*(4), 319–327. https://doi.org/10.1177/0098628319872605

Roediger, H., & McDermott, K. (1995, July). Creating false memories: Remembering words not presented in lists. *Journal of Experimental Psychology: Learning, Memory & Cognition*, *21*(4), 803–814. https://doi.org/10.1037//0278-7393.21.4.803

Scanniello, G., & Risi, M. (2013, September). Dealing with faults in source code: Abbreviated vs. full-word identifier names. *2013 ieee international conference on software maintenance*. Eindhoven, the Netherlands (pp. 190–199).

Stefik, A., & Hanenberg, S. (2014). The programming language wars: Questions and responsibilities for the programming language community. *Proceedings of the 2014 acm international symposium on new ideas, new paradigms, and reflections on programming & software* (pp. 283–299). Association for Computing Machinery. New York, NY, USA. https://doi.org/10.1145/2661136.2661156.

Stefik, A., & Siebert, S. (2013, November). An empirical investigation into programming language syntax. *ACM Transactions on Computing Education*, *13*(4). 1–40. https://doi.org/10.1145/2534973 .

Stefik, A., Siebert, S., Stefik, M., & Slattery, K. (2011). An empirical comparison of the accuracy rates of novices using the quorum, perl, and randomo programming languages. *Proceedings of the 3rd acm sigplan workshop on evaluation and usability of programming languages and tools* (pp. 3–8). Association for Computing Machinery. New York, NY, USA https://doi.org/10.1145/2089155.2089159.

Tibon, R., Gronau, N., Scheuplein, A.-L., Mecklinger, A., & Levy, D. A. (2014). Associative recognition processes are modulated by the semantic unitizability of memoranda. *Brain & Cognition*, *92*, 19–31. https://doi.org/10.1016/j.bandc.2014.09.009

Vigil-Colet, A., Lorenzo-Seva, U., & Condon, L. (2008). Development and validation of the statistical anxiety scale. *Psicothema*, *20*(1), 174–180.

White, E., Baldridge, E., Brym, Z., Locey, K., Mcglinn, D., & Supp, S. (2013, January). Nine simple ways to make it easier to (re)use your data. *Ideas in Ecology and Evolution*, *6*(2), https://doi.org/10.4033/iee.2013.6b.6.f

Wilson, G., Aruliah, D. A., Brown, C. T., Hong, N. P. C., Davis, M., Guy, R. T. & Wilson, P. (2014, January). Best practices for scientific computing. *PLoS Biology*, *12*(1), e1001745. https://doi.org/10.1371/journal.pbio.1001745

Zong, J., Lee, C., Lundgard, A., Jang, J., Hajas, D., & Satyanarayan, A. (2020). Rich Screen Reader Experiences for Accessible Data Visualization. *Computer Graphics Forum (Proc. EuroVis)*. Retrieved from http://vis.csail.mit.edu/pubs/rich-screen-reader-vis-experiences.