reVISit: Supporting Scalable Evaluation of Interactive Visualizations

Yiren Ding
Worcester Polytechnic Institute

Jack Wilburn
University of Utah

Hilson Shrestha
Worcester Polytechnic Institute

Akim Ndlovu
Worcester Polytechnic Institute

Kiran Gadhave
University of Utah

Carolina Nobre
University of Toronto

Alexander Lex*
University of Utah

Lane Harrison†
Worcester Polytechnic Institute

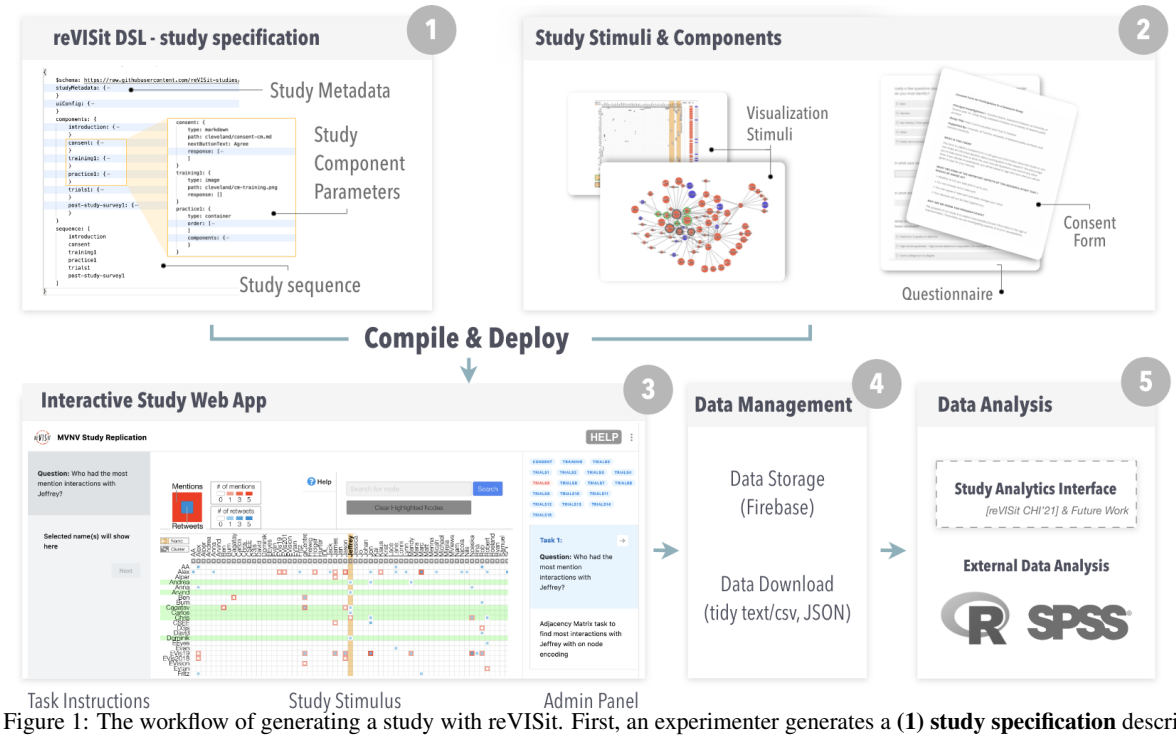

Figure 1: The workflow of generating a study with reVISit. First, an experimenter generates a **(1) study specification** describing the study using the reVISit DSL. They also create **(2) study stimuli and components** such as consent forms and trainings. Stimuli and components can be specified as HTML pages, markdown, static images, React components, etc. The specification and the study components are then compiled to an **(3) interactive web app**, which can be deployed to the web. The web app includes an admin interface for quickly browsing and debugging a study. Data from the study can either be **(4) downloaded** at the end of a trial, or **stored on a server**. Collected data can be **(5) analyzed** with external software such as R and SPSS, or examined and analyzed using the reVISit analytics interface [15].

Abstract

reVISit is an open-source software toolkit and framework for creating, deploying, and monitoring empirical visualization studies. Running a quality empirical study in visualization can be demanding and resource-intensive, requiring substantial time, cost, and technical expertise from the research team. These challenges are amplified as research norms trend towards more complex and rigorous study methodologies, alongside a growing need to evaluate more complex interactive visualizations. reVISit aims to ameliorate these challenges by introducing a domain-specific language for study set-up, and a series of software components, such as UI elements, behavior provenance, and an experiment monitoring and management interface. Together with interactive or static stimuli provided by the experimenter, these are compiled to a ready-to-deploy web-based experiment. We demonstrate reVISit's functionality by re-implementing two studies — a graphical perception task and a more complex, interactive study. reVISit is an open-source community project, available at `https://revisit.dev/`.

Index Terms: Human-centered computing—Software prototype—Visualization systems and tools—Empirical Study

1 Introduction

Well-designed visualization experiments help us understand how people and visualizations interact, allowing us to gain insight into the cognitive processes underlying data visualization use. They are also a primary method of evaluating emerging visualization techniques, comparing competing visualizations across tasks, and helping us understand and validate choices when communicating data at scale.

Conducting visualization studies can be challenging, however. A research team often requires one or more members with significant technical expertise to create a robust visualization study. Visualization systems or experiment stimuli may also use complicated visualization frameworks, client-server architectures, front-end toolkits, databases, among other technologies. There are also significant complexities in crafting experiments: randomization, attention checks, handling invalid operations, data tracking, etc. Moreover, an ex-

*e-mail: alex@sci.utah.edu
†e-mail: ltharrison@wpi.edu

periment will likely have multiple variations for piloting, demo, or validation purposes. These efforts require rich experience in both software engineering and experiment design. Visualizations themselves are also becoming more complex. For instance, stimuli are changing from static pictures or simple charts [1, 4, 11, 12, 20, 25] to animated or even interactive visualizations [2, 9, 16]. Experiments may also blend qualitative and quantitative approaches, requiring more effort in development.

Fortunately, progress in visualization theory, practice, and technology shows a path forward. Web-based visualization frameworks and web technology are entering a mature era. Visualization specification languages and grammars help define and create visualizations. Recent developments in visualization experiment practice bring better study designs, methodologies, statistical approaches, and tools for analyzing study data, while simultaneously pushing for more transparency. These advances provide a promising baseline to explore new possibilities. Nevertheless, to date, public and open visualization study infrastructure has not kept pace with these developments. Research teams often use ad-hoc solutions for their experiments or recycle one-off prototypes from other labs. General alternatives such as Qualtrics or SurveyMonkey offer solutions to some (but not all) of these problems, yet are closed, commercial environments, hence are only accessible to well-resourced research teams. The nature of these platforms makes setups difficult to share, hampering reproducibility.

To remedy these issues, our team is developing **reVISit as a community-focused open-source effort to build tools for creating visualization studies**. In this paper, we introduce a first reVISit prototype with core functionalities. reVISit's expressive low-code study specification schema supports a range of visualization experiments. reVISit's framework and reusable components aim to reduce the time and effort to refine study prototypes and final designs. reVISit's integrated interaction provenance data tracking produces ready-to analyze datasets of participant behavior, allowing researchers to revisit individual moments and patterns in study activity. reVISit's experiment monitoring interface gives real-time overviews of study progress. Finally, reVISit studies can be packaged and shared supporting transparency and reproducibility.

Our contribution is in the spirit of an applications note —– a short paper with the goal of informing the visualization research community of newly developed technologies addressing common challenges of broad interest. We believe this applications note is consistent with calls for research and approaches that are necessary to tackle the increasing complexity of empirical studies in the visualization community while also addressing practical pain points for research teams in our community.

We demonstrate the utility of reVISit by recreating Cleveland & McGill's graphical perception comparison task [4] and a multivariate network visualization evaluation [16].

2 Background

We first give an overview of representative studies that could benefit from reVISit and then discuss other related frameworks.

2.1 Empirical Studies in Visualization

Empirical studies in visualization involve conducting experiments to understand how people interact with visual encodings of data, with the aim of understanding human behavior, perception, cognition, or improving the design and effectiveness of visualizations. Many experiments present stimuli, followed by a corresponding question per trial. For instance, Cleveland & Mcgill's graphical perception experiment contains comparison tasks to measure how well people can retrieve data from different visual channels. Related studies targeting perceptual issues in visualization are common e.g. [11, 20, 22, 24]. Likewise, Borkin et al. investigated what makes a visualization memorable by presenting participants with a sequence of static images and subsequently asking if they had seen an image before [1]. The trial format also has been adopted by many visualization literacy assessments [3, 8, 12]. Notably, many recent studies are conducted as online experiments, placing a high technical burden on the experimenters. Also, while some examples use standard form fields for responses, others require custom widgets for providing answers such as for drawing bars [13] or icon arrays [18].

Over the last decade, advances in web-based visualization frameworks have facilitated the development of interactive data visualizations, opening up many new opportunities for research on interaction methods and more complex tasks [26]. For instance, Nobre et al. compare interactive multivariate node-link diagrams and adjacency matrices in a crowdsourced experiment [16]. For experiments like these, the technical burden of developing interactive, validated stimuli is significant, which is aggravated by a lack of integrated participant tracking and experimentation frameworks.

2.2 Experiment Frameworks

Empirical studies are not unique to data visualization; here we discuss both, generic experiment frameworks as well as those that are explicitly designed for visualization studies. A small fraction of simple studies can be done using Google Forms. A more advanced framework is SurveyMonkey, a commercial cloud-based software covers the development, deployment, and data collection phase of survey-style experiments [21]. Qualtrics is a similar commercial service that supports more complex experiment logic and provides data analysis functions [17]. Such systems can accommodate custom interactive stimuli, but lack integration between the stimuli and the experiment framework, which is often desirable for debugging studies and analyzing data. Using commercial GUI-based systems also often results in limited reproducibility (e.g., the experimental setup can't be easily submitted with a publication), and are an impediment to replication studies, for the lack of having the setup available and having access to the commercial tool. JsPsych is a mature open-source JavaScript framework that provides a collection of tools for creating behavioral experiments in JavaScript but requires substantial programming skills to use [6]. Experimentr (developed by our team) is a similar effort in the visualization community, but also can require substantial web development expertise for module development [10]. reVISit differs in focus by supporting more complex interactive visualizations and developing a low-code specification language in addition to library components. reVISit is orthogonal to participant recruiting platforms such as Prolific, Amazon Mechanical Turk and Lab in the Wild [19], and can be used with any of these.

3 The reVISit Study Framework

reVISit is envisioned to be an end-to-end visualization experiment support framework, including both the **study framework**, that enhances experiment development with integrated features such as study provenance tracking and reusable components, as well as debugging and analysis capabilities. In this paper, we focus on the study framework only (Figure 1.1-5). We previously developed a study data analytics interface, also called reVISit [15], which is distinct from the study framework we introduce here, but will be integrated in the future (see Figure 1.5).

reVISit introduces several innovations, including a domain-specific language (DSL) for defining experiments, reusable components for common experiment features (e.g., consent forms, literacy tests), and administrative interfaces for managing experiments. These are supported by the integration of a provenance tracking library [5] and data management capabilities (e.g. database options, tidy data export, etc.). Finally, reVISit is designed to be self-contained for portability and transparency.

The basic workflow of generating a study with reVISit is shown in Figure 1. First, experimenters design the experiment flow and

generate a **specification** describing the study using the DSL. In their scaffolding script, they specify study components. **Components** can be selected from pre-existing modules, such as consent forms or standardized tests, or they can be custom, as would commonly happen for study stimuli. Components can be created based on a markdown file, forms, images, HTML/JavaScript, and React components. Finally, the specification and stimuli are compiled into an interactive web app, which can be deployed to the web. Once a study is compiled, it can be viewed in an admin interface, which can be used to browse and debug the study quickly.

After participants complete the study, data can either be **downloaded** at the end of a trial (for serverless deployment, e.g., in a lab or field study) or **stored on a server**. We provide interfaces and examples for storing the data in Firebase, but Firebase can be easily replaced with custom data stores. Once data was collected, it can be analyzed using standard tools, such as R, or explored with the reVISit analytics interface [15].

3.1 A DSL for Study Configuration

reVISit introduces **reVISit.spec** — a declarative DSL [14] for specifying visualization experiments. reVISit.spec uses Hjson[1], a human-readable JSON-like data format that compiles to JSON. reVISit is designed to be a flexible framework that allows experimenters to easily define study structures, trials, and different components using key-value associations. It supports diverse customizable components, including consent, training, practice, trials, and post-study surveys. To help experimenters get started quickly, reVISit provides templates and example replications of various studies that can be easily adapted (e.g., reordering trial sequences, adding/deleting components, or changing any content like stimuli and questions). This specification is parsed and compiled into a working experiment, including core reVISit data management and tracking by default.

The flexible reVISit.spec, paired with the stimulus formats, enables the framework to support various experiment structure types common in visualization studies, including multi-block trials like those used in graphical perception and color studies (e.g. [4, 22]), as well as more complex visualizations used in task-based studies (e.g. [16]). Listing 1 shows a partial example specifying a dynamic stimulus (the Cleveland-McGill bar chart) implemented as a React component. The values of the bar chart and the selected bars are specified in a data object that is passed to the React component. The correct answer and the task prompt are also specified. In this case, a response field used for the task is specified, including prompt and parameters for data validation. The complete example is available via the reVISit website.

We also provide commonly used UI elements for survey questions that can be specified in the reVISit.spec. Data can be collected via form elements defined in the specification or through callbacks and hooks when using JS or React to implement complex stimuli.

reVISit.spec configurations can be written manually, or generated through an external automated process (e.g., generating simulated data in R), or within the logic of a component itself. reVISit currently supports the explicit ordering of components, but randomization strategies are an immediate next milestone.

3.2 reVISit Interface Components

As a visualization experiment framework, reVISit includes necessary interfaces and interface components to help experimenters in the design and deployment process.

3.2.1 Study Interface

reVISit provides a set of common experiment interface components that can be configured via reVISit.spec. Default layouts for experiments include a left sidebar that provides study instructions,

[1] https://hjson.github.io/

```
components: {
    consent: {
        type: markdown
        path: cleveland/consent-cm.md
        nextButtonText: Agree
    }
    trials1: {
        type: container
        order: [ "bubbleChart1", "stackedBarChart1" ]
        components: {
            bubbleChart1: {
                instruction: What percentage is the smaller value?
                type: react-component
                path: BubbleChart.tsx
                parameters: {
                    data: [
                        { name: "A", value: 30 }
                        { name: "B", value: 40 }
                    ]
                    selectedIndices: [0, 1]
                }
                response: [{
                        id: cm-response1
                        prompt: Your answer
                        required: true
                        location: sidebar
                        type: numerical
                        placeholder: Answer 0-100
                        max: 100
                        min: 0
                    }]
                nextButtonLocation: sidebar
                instructionLocation: sidebar
            }
            // More stimuli would be defined here
        }
    }
}
```

Listing 1. A snippet of a reVISit.spec configuration for the Cleveland & McGill experiment, using a React stimulus.

inputs, legends, or any other additional content (see Figure 1.3). The experiment header provides study navigation, awareness features, as well as help functionality for participants. These UI components can be displayed or hidden based on their configuration.

The interface also provides rich form fields that are automatically tracked as participants complete a study. For example, the specification in Section 3.1 defines a numeric input, which is rendered as a text box that also performs input validation.

These pre-built features are intended to speed-up the design and development of studies while being flexibly deployed or used on-demand.

3.2.2 Admin Interface

Debugging and pilot testing are essential when developing a new study. reVISit supports such efforts through an administrative sub-interface as part of the main study interface (see Figure 1). Driven by the configuration file, this view provides a compact visualization of the study layout (Figure 1.3), which doubles as a navigation bar. Individual tasks fill the sidebar, allowing experimenters to navigate to study stages of interest (rather than the common method of rapidly taking the survey to reach a particular page). Task/trial cards are designed to be bound to a single participant's data when available, also indicating the performance on a task.

3.3 Provenance Tracking and Data Management

One challenge with many experiments is deciding what to track. Logging every mouse movement, for example, may lead to increased data size, with potentially little benefit. Conversely, logging too little can lead to ambiguous and difficult-to-interpret results during design

and final deployment/analysis. reVISit tightly integrates trrack [5], a visualization provenance library developed and maintained by our team, which we have used before to track study meta-data and detailed interaction provenance in two experiments [7, 16].

In any experiment built using the reVISit framework, all responses and experiment metadata, such as progress, attributes of questions, and sequence are automatically recorded. Researchers seeking task-level provenance data, such as mouse movements or keystrokes, are able to integrate such provenance tracking inside stimuli. This can be achieved through either utilizing the trrack library or implementing a custom solution in the stimuli.

reVISit provides both **online and offline data storage solutions**. Researchers have the freedom to pass the data to a cloud storage system (our demo uses Firestore), store it on a private server, or download the experiment data after a trial is complete (which is useful for laboratory studies or debugging sessions). Provenance data are stored in JSON format and can be exported to a tidy CSV format [23] for analysis in popular tools.

3.4 Deployment

Deploying reVISit is an essential part of a researcher's workflow. Our suggested deployment steps for a reVISit study involve forking the GitHub repository, modifying the configuration files, and thereafter, committing and pushing changes to the forked version of the repository. This process activates a built-in GitHub Actions deployment workflow, significantly reducing the intricacies associated with hosting the site on GitHub Pages. Support is also provided for custom deployment, e.g., on university servers or other cloud infrastructures.

4 Experiment Examples

To illustrate the capabilities of reVISit, we re-implemented two experiments, in addition to several simple instructive examples. A demo site `https://revisit.dev/study/`, hosts these experiments and examples. The reVISit.spec, stimuli, and components are available at `https://github.com/revisit-studies/study/`.

4.1 Cleveland and McGill

Building on our example thus far, we describe our re-implementation of the graphical perception experiments [4]. The components and configuration reside in a repository deployed using GitHub pages, ready to collect data. It includes four chart components that can be reused to produce multiple stimuli, each with distinct data parameters sourced from the configuration. Each component can generate visualizations and marks from data in the spec, so that the same implementation can be used in many variations of the experiment.

4.2 Multi-Variate Network Visualization (MVNV)

One goal of reVISit is to support complex visualization experiments. The original MVNV study [16] included 32 tasks across multivariate node-link and matrix visualizations, including questions that must be answered by interacting with the visualization. We include all 16 tasks from the adjacency matrix condition to illustrate advanced reVISit functionality. Similar to the simpler demonstration, tasks are defined in a configuration file, which link to specific stimuli, defined in HTML/JavaScript modules in this case, which render as iFrames. This experiment also makes use of the left sidebar, similar to the original MVNV study (see Figure 1).

5 Discussion

The reVISit project aims to develop infrastructure that will democratize (i.e., make available to broader audiences), accelerate, and advance our ability to conduct web-based and crowdsourced studies. In its current form, reVISit is a suite of modular but compatible tools to design and debug studies of interactive visualization techniques. While there are various commercial and some open/academic frameworks that support these workflows for survey-based research, there is no mature infrastructure that supports experiments based on custom, complex stimuli which are commonly used in visualization research.

Importantly, reVISit is open source, meaning that if the specification and stimuli are published, anyone can easily re-run an experiment exactly as it was originally conducted. reVISit experiments also compile to standard, serverless websites, avoiding potential security issues and complicated setups, which ensures their longevity. This makes research more transparent, reproducible (e.g., enabling replication studies), and open to scrutiny in a manner that can be difficult or impossible with other tools.

While experimenters using reVISit have to edit the specification to customize it to their needs, there is no programming required to create and launch basic experiments (e.g. using images and rating scales only). Advanced experimenters with programming expertise are able to focus on more complex stimuli development. Our dedication to streamlining the user experience allows researchers to concentrate on harnessing the potential of our tool for their projects without the burden of complex deployment procedures.

6 Future Work

reVISit is an NSF funded infrastructure project that is in the prototype phase of its development. This paper focuses on the core components for setting up and running a study. We also sketch our vision below for debugging, analytics, and data collection.

Our approach to **debugging** studies and study stimuli is based on (a) the ability to rapidly view/edit/deploy a study and (b) to view pilot data and detailed user behavior. We plan to create an interface that facilitates the quick construction of a reVISit.spec configuration. This configuration can be further refined for a comprehensive setup through JSON editing. Based on our integration of provenance tracking, we will be able to, for example, identify participants who did not correctly complete a task and step through every interaction they performed to identify any potential problems, as we have demonstrated in a technology probe [15].

For **analytics**, we will use a two-pronged approach. First, we will continue making it easy to export data in useful formats so experimenters can run statistical analysis with their preferred tools. Second, we will enable more advanced analysis of participant usage behavior based on event sequences [15]. For example, in the analytics interface, we will enable experimenters to stratify their data based on usage patterns, look at outcomes for the subgroups, and explore the steps of individual participants.

Finally, for **qualitative data collection**, we intend to make it easier for experimenters to run qualitative studies, such as think-aloud experiments online. We will make recording participant audio possible, automatically transcribing and annotating it, and associating findings, text, and audio snippets with events and/or outcomes.

At the same time, we are actively seeking input from the community so that they can participate in shaping the future of interactive, online experiments.

7 Conclusion

We have introduced reVISit, a new open research infrastructure that enables experimenters to create reproducible online studies efficiently. We have described reVISit.spec, a domain-specific language for defining all aspects of a study, from consent, to training, to stimuli, to data collection modalities. reVISit is modular, easy to deploy, and ready to use by the community. In the future we will continue to develop reVISit capabilities, and seek community input when developing the roadmap for the coming years.

ACKNOWLEDGMENTS

We are grateful to Cindy Xiong Bearfield, Lace Padilla, and Danielle Albers Szafir for advice on the requirements of a study platform. reVISit is funded by the National Science Foundation (CNS 213756 and 2213757).

REFERENCES

[1] M. A. Borkin, A. A. Vo, Z. Bylinskii, P. Isola, S. Sunkavalli, A. Oliva, and H. Pfister. What Makes a Visualization Memorable? *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2306–2315, Dec. 2013. doi: 10.1109/TVCG.2013.234

[2] J. Boy, L. Eveillard, F. Detienne, and J.-D. Fekete. Suggested interactivity: Seeking perceived affordances for information visualization. *IEEE transactions on visualization and computer graphics*, 22(1):639–648, 2015.

[3] J. Boy, R. A. Rensink, E. Bertini, and J.-D. Fekete. A principled way of assessing visualization literacy. *IEEE transactions on visualization and computer graphics*, 20(12):1963–1972, 2014.

[4] W. S. Cleveland and R. McGill. Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984. doi: 10.1080/01621459.1984.10478080

[5] Z. T. Cutler, K. Gadhave, and A. Lex. Trrack: A Library for Provenance Tracking in Web-Based Visualizations. In *IEEE Visualization Conference (VIS)*, pp. 116–120. IEEE, 2020. doi: 10.1109/VIS47514.2020.00030

[6] J. R. De Leeuw. jspsych: A javascript library for creating behavioral experiments in a web browser. *Behavior research methods*, 47:1–12, 2015.

[7] K. Gadhave, J. Görtler, Z. Cutler, C. Nobre, O. Deussen, M. Meyer, J. M. Phillips, and A. Lex. Predicting intent behind selections in scatterplot visualizations. *Information Visualization*, 20(4):207–228, Oct. 2021. doi: 10.1177/14738716211038604

[8] L. W. Ge, Y. Cui, and M. Kay. Calvi: Critical thinking assessment for literacy in visualizations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pp. 1–18, 2023.

[9] H. Guo, S. R. Gomez, C. Ziemkiewicz, and D. H. Laidlaw. A case study using visualization interaction logs and insight metrics to understand how analysts arrive at insights. *IEEE transactions on visualization and computer graphics*, 22(1):51–60, 2015.

[10] L. Harrison. codementum/experimentr, Jan. 2022. original-date: 2013-04-01T16:27:54Z.

[11] L. Harrison, F. Yang, S. Franconeri, and R. Chang. Ranking visualizations of correlation using weber's law. *IEEE transactions on visualization and computer graphics*, 20(12):1943–1952, 2014.

[12] S. Lee, S.-H. Kim, and B. C. Kwon. Vlat: Development of a visualization literacy assessment test. *IEEE transactions on visualization and computer graphics*, 23(1):551–560, 2016.

[13] C. M. McColeman, L. Harrison, M. Feng, and S. Franconeri. No mark is an island: Precision and category repulsion biases in data reproductions. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1063–1072, 2020.

[14] A. M. McNutt. No Grammar to Rule Them All: A Survey of JSON-style DSLs for Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):160–170, Jan. 2023. doi: 10.1109/TVCG.2022.3209460

[15] C. Nobre, D. Wootton, Z. Cutler, L. Harrison, H. Pfister, and A. Lex. reVISit: Looking Under the Hood of Interactive Visualization Studies. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–13. ACM, Yokohama Japan, May 2021. doi: 10.1145/3411764.3445382

[16] C. Nobre, D. Wootton, L. Harrison, and A. Lex. Evaluating Multivariate Network Visualization Techniques Using a Validated Design and Crowdsourcing Approach. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–12. ACM, 2020. doi: 10.1145/3313831.3376381

[17] Qualtrics International Inc. Qualtrics. `https://www.qualtrics.com/`, 2023. Accessed: 2023-07-23.

[18] N. Rakotondravony, Y. Ding, and L. Harrison. Probablement, wahrscheinlich, likely? a cross-language study of how people verbalize probabilities in icon array visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 1912.

[19] K. Reinecke and K. Z. Gajos. LabintheWild: Conducting Large-Scale Online Experiments With Uncompensated Samples. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '15, pp. 1364–1378. Association for Computing Machinery, New York, NY, USA, Feb. 2015. doi: 10.1145/2675133.2675246

[20] R. A. Rensink and G. Baldridge. The perception of correlation in scatterplots. In *Computer Graphics Forum*, vol. 29, pp. 1203–1210. Wiley Online Library, 2010.

[21] SurveyMonkey Inc. Survey Monkey. `https://www.surveymonkey.com/`, 2023. Accessed: 2023-07-23.

[22] D. A. Szafir. Modeling color difference for visualization design. *IEEE transactions on visualization and computer graphics*, 24(1):392–401, 2017.

[23] H. Wickham. Tidy Data. *Journal of Statistical Software*, 59:1–23, Sept. 2014. doi: 10.18637/jss.v059.i10

[24] C. Xiong, A. Sarvghad, D. G. Goldstein, J. M. Hofman, and Ç. Demiralp. Investigating perceptual biases in icon arrays. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2022.

[25] C. Xiong, C. Stokes, Y.-S. Kim, and S. Franconeri. Seeing what you believe or believing what you see? belief biases correlation estimation. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):493–503, 2022.

[26] K. Xu, A. Ottley, C. Walchshofer, M. Streit, R. Chang, and J. Wenskovitch. Survey on the analysis of user interactions and visualization provenance. In *Computer Graphics Forum*, vol. 39, pp. 757–783. Wiley Online Library, 2020.