CausalSynth: An Interactive Web Application for Synthetic Dataset Generation and Visualization with User-Defined Causal Relationships

Zhehao Wang* UNC-Chapel Hill
Arran Zeyu Wang† UNC-Chapel Hill
David Borland‡ RENCI, UNC-Chapel Hill
David Gotz§ UNC-Chapel Hill

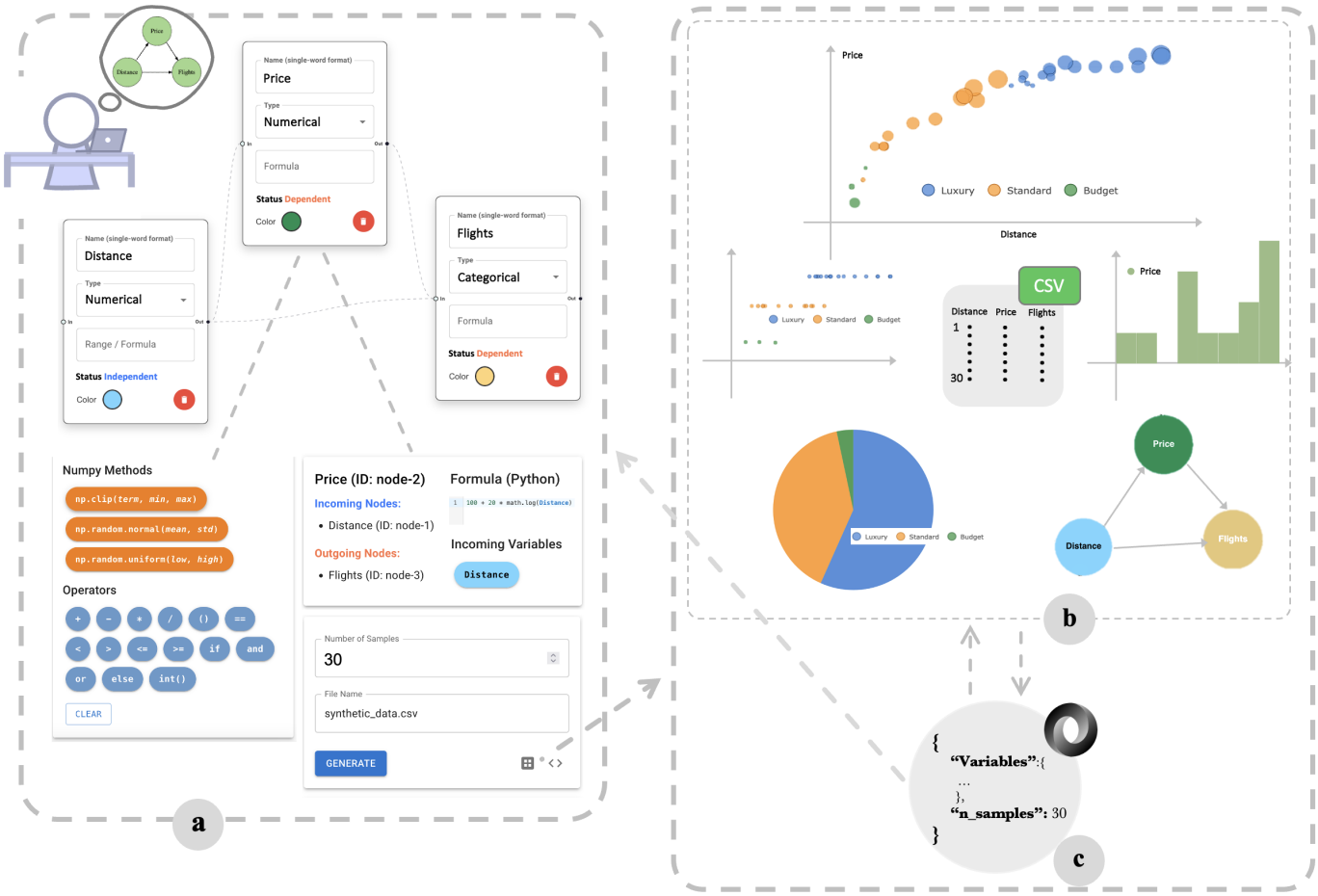

Figure 1: An overview of *CausalSynth*'s workflow with an example causal model, showcasing (a) variable definition and causal relationship configuration, (b) built-in visualization tools, and (c) generated data in JSON format.

Abstract

Understanding and inferring causal relationships between variables is a fundamental task in visualization and visual analysis. However, it can be challenging to verify inferences of causal relationships from traditional observational data because they often lack a ground truth causal model, complicating the evaluation of visual causal inference tools. To address this challenge, we introduce *CausalSynth*, an interactive web application designed to generate synthetic datasets from user-defined causal relationships. *CausalSynth* enables users to define acyclic causal graphs via a user-friendly graphical interface, establish interrelationships between variables, and produce datasets that reflect these desired causal interactions. The application also includes built-in tools for visualizing the generated datasets, facilitating deeper insights into the user-defined causal structure and aiding the validation of the generated data. By providing a user-friendly interface for synthetic data generation and visualization based on ground truth causal models, *CausalSynth* helps support more meaningful evaluations of visual causal inference technologies.

*e-mail: zhehaow24@gmail.com
†e-mail: zeyuwang@cs.unc.edu
‡e-mail: borland@renci.org
§e-mail: gotz@unc.edu

1 Motivation

The primary motivation behind the development of *CausalSynth* stems from the significant challenges associated with causal inference in observational data. One major issue is the presence of confounding variables, which can obscure true causal relationships and lead to biased estimates [1]. Observational data often lacks the controlled conditions of randomized experiments, making it difficult to draw definitive conclusions about causality [4].

Moreover, the lack of public benchmark datasets for causal inference further complicates the evaluation of visual causal inference tools. Unlike associative techniques, which can be easily tested and compared using widely available datasets, causal inference techniques are perhaps most meaningfully validated using data for which explicit causal relationships are known. However, these causal ground truths are rarely available [5]. This scarcity hinders the development and validation of new visual causal inference methodologies.

Additionally, real-world data is often incrementally available and non-stationary, posing further difficulties in maintaining accurate causal models over time. The need to continuously update models with new data without compromising previous estimations is a complex task that requires sophisticated methods for continual learning and adaptation [2].

Approaches like CausalVis [3] tried to better visualize such causal graphs, however, there's still limited effort in visually assisting dataset generation. To address these challenges, *CausalSynth* aims to provide a robust and efficient tool for generating and validating (via visualization) synthetic datasets with user-defined causal relationships. These can serve as ground truth datasets for the evaluation of visual causal inference technologies.

2 Design and Implementation

The design of *CausalSynth* emphasizes user experience, featuring a clean and intuitive interface for generating, inspecting, and downloading synthetic datasets. Central to the user experience is the acyclic graph definition area. Implemented using React Flow, this area enables users to visually define new variables and manage the relationships between them. Users can assign different colors to specific variables to enhance visual clarity. These colors are consistently reflected elsewhere in the application.

When a user clicks on a node, detailed information about the node is displayed. This includes options for an explicit Python-based formula used to generate data for the node's corresponding variable, or higher-level parameters such as range, categorical values, and corresponding probability values that can be used to generate data for built-in distribution types. Similarly, clicking on an edge reveals details about the variable relationship it represents, including the source and target of the connection. This helps users understand the structure of the causal relationships represented within the graph.

The formula box in which users can define explicit formulas includes syntax highlighting and other features for ease of use.For instance, a virtual keyboard with buttons containing common NumPy methods and Python operators is featured. In addition, an "Incoming Variables" section includes buttons representing the incoming variable names for the selected node, making it easy to incorporate causal relationships to connected variables. These are color-coded to match the user-selected colors in the causal graph.

After defining the graph, including all variables' attributes and interrelationships, a JSON file is generated to document the specification. This JSON file can be viewed and saved by the user, and can be imported back into the web application to restore the same configuration at a later time. The same JSON is processed by the system's Flask-based backend to generate a corresponding synthetic data set. A CSV file containing the generated data is returned to the browser to be viewed within the application or downloaded for subsequent analysis. Within the application users can generate various visualizations, including scatter plots, bubble plots, histograms, pie charts, and causal graphs. These features enable users to quickly inspect the data generated by their causal graph and make adjustments if necessary. The source code of *CausalSynth* will be released via GitHub following the review cycle, enabling users to freely leverage these tools to create ground truth datasets to aid in the evaluation of their own visual causal inference software.

3 Example Usage

A typical workflow for *CausalSynth* is illustrated in Fig. 1. The process starts in Fig. 1 (a), where three variables are presented: **Distance**, **Price**, and **Flights**, with each assigned a distinct color for clarity. **Distance** is defined as a numerical, independent variable with a specified range of 100 to 3000. **Price** is a numerical, dependent variable calculated using a formula involving **Distance**: `100 + 20 * math.log(Distance) + 10 * np.random.random()`. **Flights** is a categorical variable that depends on both **Price** and **Distance**, categorized as one of "Budget," "Standard," "Premium," or "Luxury" based on thresholds set for **Distance** and **Price**.

Interdependence between these variables is visually represented in the acyclic graph with edges between nodes that the user connects. In Fig. 1 (a), **Price** depends on **Distance**, and **Flights** depends on both **Price** and **Distance**, while **Distance** has no incoming nodes. The user clicks on the **Price** node to display the variable's incoming node (**Distance**) and outgoing node (**Flights**), as well as the formula used to calculate **Price** and the incoming variable name **Distance** highlighted in the formula box. While this example includes a relatively simple graph, much more complex models can be quickly created in a similar way.

After defining the variables and their relationships, the user specifies the number of desired samples to be generated, in this case 30 as shown in the bottom section of (a). A JSON file (c) capturing the full configuration is created and can be viewed, saved, and imported back into the web app at a later time. Meanwhile, a CSV file is generated containing the requested samples.

The data in the CSV file can be inspected in the application using various visualizations as shown in (b). In the bubble plot shown, the size of the data points is set to represent their respective price. The colors are set to represent different flight categories. Upon closer inspection of the plot, the user can see a notable logarithmic relationship between **Price** and **Distance**; additionally, as distance and price increase, flights tend to be categorized into more luxurious categories. This example demonstrates how *CausalSynth* enables users to define causal relationships, generate synthetic data, and visualize it to validate that the variables are interacting in ways that appear to reflect the intended underlying causal model.

4 Conclusion

CausalSynth can quickly and intuitively generate synthetic datasets based on user-defined causal relationships while providing ground-truth data for the development, evaluation, and comparison of new visual causal inference technologies. The JSON-based specification also enables the persistence of generative causal model data for provenance and reuse. It can be a valuable tool in support of this important and active area of visualization research. Future work should explore more effective visual representations and user-friendly interactions through real-world user studies.

Acknowledgments

This research is made possible in part by NSF Award #2211845.

References

[1] D. Borland, A. Z. Wang, and D. Gotz. Using counterfactuals to improve causal inferences from visualizations. *IEEE CG&A*, 44(1):95–104, 2024.

[2] Z. Chu, R. Li, S. Rathbun, and S. Li. Continual causal inference with incremental observational data. *IEEE ICDE*, 2023.

[3] G. Guo, E. Karavani, A. Endert, and B. C. Kwon. Causalvis: Visualizations for causal inference. In *ACM CHI*, pp. 1–20, 2023.

[4] L. G. Hemkens, H. Ewald, F. Naudet, and et al. Interpretation of epidemiologic studies very often lacked adequate consideration of confounding. *Journal of Clinical Epidemiology*, 93:94–102, 2018.

[5] A. Z. Wang, D. Borland, and D. Gotz. Beyond correlation: Incorporating counterfactual guidance to better support exploratory visual analysis. *IEEE TVCG (Proc. IEEE VIS 2024)*, 2025.