CausalPlayground: Addressing Data-Generation Requirements in Cutting-Edge Causality Research

Anonymous CVPR submission

Paper ID *****

Abstract

Research on causal effects often relies on synthetic data due to the scarcity of real-world datasets with ground-truth effects. Since current data-generating tools do not always meet all requirements for state-of-the-art research, ad-hoc methods are often employed. This leads to heterogeneity among datasets and delays research progress. We address the shortcomings of current data-generating libraries by introducing CausalPlayground*, a Python library that provides a standardized platform for generating, sampling, and sharing structural causal models (SCMs).* CausalPlayground *offers fine-grained control over SCMs, interventions, and the generation of datasets of SCMs for learning and quantitative research. Furthermore, by integrating with* Gymnasium*, the standard framework for reinforcement learning (RL) environments, we enable online interaction with the SCMs. Overall, by introducing* CausalPlayground *we aim to foster more efficient and comparable research in the field. All code is available at* `https://anonymous.4open.science/r/CausalPlayground-D1B2/`*. The API documentation will be released after de-anonymization.*

1. Introduction

Ever since the formalization of causality [18] the field has gained significant attention for improving inference from data, scientific discovery, and others. Progress in causality research relies heavily on data collection about the scenario that is being investigated. However, a major challenge is the lack of real-world data with known causal ground truth [6]. As an example, even after decades of research, only a few strong real-world benchmark datasets are available for causal structure discovery [6, 32]. Consequently, many state-of-the-art methods use synthetically generated data, such as in causal representation learning [16], causal discovery [9], among others.

Naturally, each research comes with its specific requirements for the data-generating process that is being investigated. In causality particularly though, some requirements can be observed that are common among many instances. We identify them as requirements for a causal data-generating library to be useful for a broad variety of research questions as follows:

R1 Interventional data generation: Intuitively, interventions are experiments in an environment that are crucial for identifying causal effects [4]. Therefore, a general causal data-generation framework must facilitate the sampling of interventional data.

R2 Interaction with the causal model: It is becoming increasingly clear, that interacting with the causal model is beneficial for many tasks. For example, intervening after every sample can improve sample efficiency for causal discovery methods [22, 23] and causal inference [29]. This requirement is further highlighted by the popularity of interactive frameworks like Causal World [1] and various interactive causal models created ad-hoc for specific research questions [15, 29, 35].

R3 Fine-grained control over the causal model: As causal inference and discovery often investigate settings with clearly defined assumptions on the data, like the popular linear functions with additive with non-Gaussian noise [19] assumption, a general data-generation library must provide detailed control over the functional relations of the causal model.

R4 Causal model generation for quantitative results: Recent research in causal methods, such as [9, 17, 22, 23], emphasize the need for training and evaluation not merely on a single causal model, but rather on data-sets of models. This results in the requirement for a general causal data-generating library to be able to easily create many causal models.

As our comparison in Sec. 2 shows existing tools fall short of addressing all current requirements within one framework. This leads to many researchers having to rely on ad-hoc data generation processes, introducing undesirable heterogeneity of datasets amongst methods, more difficult comparisons of results, and generally obstructing the research progress of the field.

CVPR #*****

	interventional data	DAG-Generation	SCM-Generation	interactive	SCM-controlled	API-Documentation	Language
Do-Why [25]	✓	X	X	X	X	✓	P
TETRAD [21, 24]	X	✓	X	X	X	✓	J/P/R
cause2e [8]	✓	✓	X	X	X	✓	P
Lawrence et al. [14]	X	✓	X	X	$\sim$	X	P
MANM-CS [11]	✓	✓	X	X	$\sim$	X	P
BNlearn [27]	$\sim$	X	X	X	X	✓	P
causaldag [5]	$\sim$	✓	X	X	X	✓	P
gCastle [33]	X	✓	X	X	$\sim$	✓	P
CDT [12]	X	✓	X	X	$\sim$	✓	P
SCModels [2]	✓	X	X	X	✓	X	P
R6causal [13]	✓	X	X	X	✓	✓	R
CausalWorld [1]	✓	X	X	✓	X	✓	P
SynTReN [31]	X	✓	X	X	X	X	J
pcalg [3]	X	✓	X	X	X	✓	R
JustCause [10]	X	X	X	X	✓	✓	P
CausalPlayground	✓	✓	✓	✓	✓	✓	P

Table 1. Overview of software libraries that can be used for data-generation for causality research. Symbols ✓, X, and $\sim$ indicate whether a requirement is met, not met, or partially met, respectively. J, P, R refer to Java, Python, and R, respectively. We evaluate the methods on the ability to sample interventional data, generate directed acyclic graphs (DAG), generate SCMs, interact with the SCM, the availability of a detailed API documentation, and the programming language that it is written in.

To offer a solution to this problem we created *CausalPlayground*, a Python library for generating synthetic data and providing generation processes for causal models. Our library enables fine-grained control over the models, integrates processes within the popular RL framework *Gymnasium* [30] to accommodate the requirement for interacting with the causal model, and implements the generation of many causal models at once to enable learning and quantitative research. It thereby constitutes a framework for causal data generation that allows for more standardized and easy-to-share causal data-generation processes.

In the remainder of the paper we will compare current causal data-generating libraries in Sec. 2, introduce fundamental notions of causality more formally in Sec. 3, provide an overview of *CausalPlayground* in Sec. 4, outline a simple use-case in Sec. 5, and give an outlook on future versions of this library in Sec. 6.

2. Related Work

Based on the requirements we identified in Sec. 1, we investigate the current landscape of general purpose causal data-generation tools. We restrict our comparison to packages specifically designed for causality-related research, acknowledging that other general-purpose tools have been used, such as common RL environments like MuJoCo [28]. The comparison is summarized in Tab. 1.

Regarding interventional data-generation (R1), many available packages, including Do-Why [25], cause2e [8], MANM-CS [11], SCModels [2], R6causal [13], and CausalWorld [1], allow for sampling from interventional distributions for specifiable functional interventions. BNlearn [27] and causaldag [5] offer less versatile methods for intervening, and facilitating interventions by changing the node conditional distributions or the graphical structure, respectively.

Considering interaction with causal models (R2), only CausalWorld [1] provides out-of-the-box functionality for actively interacting with the model, highlighting an opportunity for useful new tools to meet this requirement.

With respect to generating causal models with specific assumptions on the causal functions (R3) SCModels [2], R6causal [13], and JustCause [10] give full control over functional relations and noise distributions, while Lawrence et al. [14], MANM-CS [11], gCastle [33], and CDT [12] provide coarse-grained control that simultaneously applies to all functions and distributions.

Lastly, to the best of our knowledge, no general-purpose framework for causal data-generation provides methods to generate sets of causal models with specific functional relations and noise distributions (R4).

Overall, this comparison shows a gap in libraries that simultaneously address some of the most important requirements for causal data-generation. This opens the opportunity to develop a tool that is relevant to a broader set of research questions.

3. Fundamental Concepts of Causality

In this section, we provide an overview of the necessary formal concepts for our approach. Let an SCM M be a tuple $M = (\mathcal{X}, \mathcal{U}, \mathcal{F}, \mathcal{P})$, where $\mathcal{X}$ and $\mathcal{U}$ are sets of endogenous and exogenous random variables, respectively, $\mathcal{F}$ is a set of functions f_i that map the direct causes of $X_i \in \mathcal{X}$ to X_i, and $\mathcal{P}$ is a set of pairwise independent probability distributions P_i s.t. U_i follows distribution P_i with $U_i \in \mathcal{U}$. Furthermore, we denote interventions as $do(X_i = g(...))$, where g is an arbitrary function that takes as input a subset of $\mathcal{X} \cup \mathcal{U}$. Such an intervention replaces function f_i with function g. Applying a set of N arbitrary interventions $a_t = \{do_0(\ldots), \ldots, do_N(\ldots)\}$ to an SCM, means replacing all corresponding functions simultaneously. Furthermore, each SCM M induces a distribution $P^M(\mathcal{X}, \mathcal{U} \mid a_t)$ that we can sample via ancestral sampling.

4. CausalPlayground

In this section we provide an overview of the *CausalPlayground* library. By implementing the functionalities that address the requirements R1 - R4, we provide an off-the-shelf causal data-generation platform that can be used for state-of-the-art causality research.

4.1. Fine-grained control over the SCM

For each endogenous variable X_i, the structural equation f_i can be arbitrarily defined. Similarly, the distribution of the exogenous variables P_j can be arbitrarily defined. Notably, this entails the possibility of defining pixel-based environments. An example for creating the SCM with $\mathcal{F} = \{A \leftarrow U + 5, Effect \leftarrow 2A\}$ and $U \sim Uniform(3, 8)$ is provided below:

```
scm = StructuralCausalModel()
scm.add_endogenous_var('A',
    lambda noise: noise+5,
    {'noise': 'U'})
scm.add_exogenous_var('U',
    random.randint,
    {'a': 3, 'b': 8})
scm.add_endogenous_var('Effect',
    lambda x: x*2,
    {'x': 'A'})
```

This level of control over the SCM is a significant improvement compared to other libraries, which often provide more coarse-grained control.

4.2. Arbitrary interventions

Interventions can be applied to existing SCMs using arbitrary functions g. The only restriction is that the new function does not induce cyclic causal relations. In the example code snippet:

```
scm.do_interventions([("Effect",
                      (lambda a: a+1,
                      {'a':'A'}))])
```

the intervention sets the value of the variable `Effect` to the value of $A + 1$, using a lambda function that takes `a` as an argument and a dictionary that maps `a` to the variable name `A`. This, effectively, applies $do(Effect = A + 1)$ to the SCM.

4.3. Interaction with the SCM

Users can intervene actively in existing SCMs. At each step, a set of interventions a_t can be applied, after which a sample of the endogenous and exogenous variables is drawn from the induced distribution. The interventions are undone before the next step. The interaction is enabled by wrapping the SCMs in the popular *Gymnasium* environment, facilitating standardized integration with (deep) RL methods [20]. In the example code snippet:

```
env = SCMEnvironment(scm,
        possible_interventions=
         [("A", (lambda: 5, {})),
          ("Effect", (lambda a: a+1,
                     {'a':'A'}))])
```

An `SCMEnvironment` object is created by passing the `scm` object and a list of two possible interventions with the same syntax as described before. The resulting `env` object can be used to interact with the SCM using the standard Gymnasium environment interface, allowing for interactive exploration and experimentation with the causal model.

More specifically, calling `env.step(action)` applies the interventions defined in the action via their index, samples the intervened SCM, determines the new observation, termination flag, truncated flag, and reward, and, finally undoes the interventions. Importantly, the precise observation, termination, truncated, and reward functions can be specified by the user.

4.4. SCM generation

CausalPlayground's SCM generation procedure provides a flexible approach to creating datasets of SCMs. Users can specify the number of endogenous variables and exogenous variables, the presence of confounders, and define the class of possible causal relations. This is exemplified by the following code:

```
gen = SCMGenerator(all_functions=
```

CVPR #*****

```
                    {'linear': f_linear})
scm_unconfounded = gen.create_random(
            possible_functions=["linear"],
            n_endo=5, n_exo=4,
            exo_distribution=random.gauss,
            exo_distribution_kwargs=
              {mu=0, sigma=1},
            allow_exo_confounders=
              False)[0]
```

that creates a random SCM with the causal relations following a predefined `f_linear` function, 5 endogenous variables, 4 exogenous variables, a Gaussian exogenous distribution with $\mu = 0, \sigma = 1$ for each exogenous variable, and no confounders. Similarly, an SCM can also be generated from a given causal structure.

This comprehensive control over the SCM generation process facilitates the systematic evaluation and comparison of causal models.

4.5. Further Implementation Details

CausalPlayground is implemented in Python with five main classes. The `StructuralCausalModel` class represents an SCM and provides methods for manipulation and sampling. The `SCMGenerator` creates random SCMs based on specified parameters, while the `CausalGraphGenerator` and `CausalGraphSetGenerator` generate random causal graphs and sets of unique causal graphs, respectively. The `SCMEnvironment` class, allows interaction with an SCM. Furthermore, *CausalPlayground* relies on external classes such as `networkx.DiGraph`, `pandas.DataFrame`, `Gymnasium.Box`, and `Gymnasium.Sequence` for various functionalities. All code is published under the open-source MIT license and we highly welcome contributions.

5. Use-Case: Comparison of Causal Discovery Algorithms

As one of many use-cases, we outline how to use the *CausalPlayground* library to compare different causal discovery algorithms. We generate synthetic data using the library's functions and evaluate the performance of the algorithms on both confounded and unconfounded, automatically generated SCMs. We use the causal discovery algorithms provided by the gCastle [33] library. The detailed implementation can be found in the repository examples.

First, we generate distinct confounded and unconfounded causal graphs using the `CausalGraphSetGenerator` class from *CausalPlayground*. Each of the sets has 4 endogenous variables and 4 exogenous variables. Based on these graphs we generate the SCMs for evaluation using the `SCMGenerator` class.

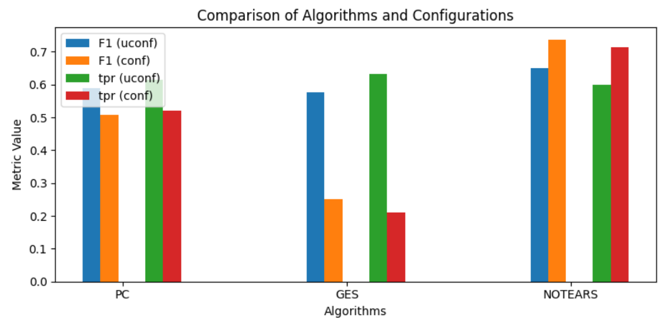

Figure 1. Experimental results of the causal discovery algorithm comparison use-case.

The functional relations are determined by specifiable functional relations. For our experiment we use linear additive functions and functions that add all causes and multiply the value of two randomly selected causes of a variable. More specifically, for every endogenous variable, the function is drawn from either of the two classes.

We then iterate over the 30 generated SCMs and draw 100 samples per SCM. We evaluate the performance of each causal discovery algorithm on both confounded and unconfounded datasets. More specifically, we compare PC [26], GES [7], and NOTEARS [34] and measure F1 score and the true-positive rate with gCastle [33]. The results can be seen in Fig. 1.

This use-case provides a glimpse into the usefulness of *CausalPlayground* for causality research, not the least because the data-generating processes could now be shared among experiments and researchers.

6. Conclusion and Outlook

In this work, we identified the core requirements for data-generation in causal research and found that they are unsatisfactorily met by the current landscape of data-generation tools. To offer a solution, we introduced *CausalPlayground*, a Python library that allows for sampling of, generation of, and online interaction with SCMs. By providing functionality to address some of the main requirements of causality research, this library can enable a broad research community to easily share their SCMs and more rigorously compare its methods.

For future versions of this library, we are anticipating the integration of more diverse causal models such as Bayesian networks, and optimizations regarding parallelization for fast deployment on GPUs. Ultimately, we envision that our proposed library contributes to faster scientific progress in fields that rely on causality.

References

[1] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Manuel Wüthrich, Yoshua Bengio, Bernhard Schölkopf, and Stefan Bauer. CausalWorld: A Robotic Manipulation Benchmark for Causal Structure and Transfer Learning, 2020. 1, 2

[2] Michael Aichmüller. Structural Causal Models, 2023. 2

[3] Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13:2409–2464, 2012. 2

[4] Elias Bareinboim, Juan D. Correa, Duligur Ibeling, and Thomas Icard. On Pearl's Hierarchy and the Foundations of Causal Inference. In *Probabilistic and Causal Inference*, pages 507–556. ACM, New York, NY, USA, 2022. 1

[5] Chandler Squires. causaldag: creation, manipulation, and learning of causal models, 2018. 2

[6] Lu Cheng, Ruocheng Guo, Raha Moraffah, Paras Sheth, K Selçuk Candan, Huan Liu, Hong Kong, and Huan Liu are. Evaluation Methods and Measures for Causal Learning Algorithms. *IEEE Transactions on Artificial Intelligence*, 3(6), 2022. 1

[7] David Maxwell Chickering. Optimal Structure Identification with Greedy Search. *J. Mach. Learn. Res.*, 3(null):507–554, 2003. 4

[8] Daniel Grünbaum. cause2e: A Python package for end-to-end causal analysis., 2021. 2

[9] Tristan Deleu, Mizu Nishikawa-Toomey, Jithendaraa Subramanian, Nikolay Malkin, Laurent Charlin, and Yoshua Bengio. Joint Bayesian Inference of Graphical Structure and Parameters with a Single Generative Flow Network. In *Advances in Neural Information Processing Systems*, pages 31204–31231. Curran Associates, Inc., 2023. 1

[10] Maximilian Franz, Florian Wilhelm, and Tanmay Kulkarni. JustCause, 2020. 2

[11] Johannes Huegle, Christopher Hagedorn, Lukas Böhme, Mats Pörschke, Jonas Umland, and Rainer Schlosser. MANM-CS: Data generation for benchmarking causal structure learning from mixed discrete-continuous and nonlinear data. In *WHY-21 at NeurIPS 2021*, 2021. 2

[12] Diviyan Kalainathan and Olivier Goudet. Causal Discovery Toolbox: Uncover causal relationships in Python. 2019. 2

[13] Juha Karvanen. R6causal: R6 Class for Structural Causal Models, 2023. 2

[14] Andrew R. Lawrence, Marcus Kaiser, Rui Sampaio, and Maksim Sipos. Data Generating Process to Evaluate Causal Discovery Techniques for Time Series Data. 2021. 2

[15] Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M Asano, Taco Cohen, and Stratis Gavves. Citris: Causal identifiability from temporal intervened sequences. In *International Conference on Machine Learning*, pages 13557–13603, 2022. 1

[16] Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M Asano, Taco Cohen, and Efstratios Gavves. BISCUIT: Causal Representation Learning from Binary Interactions. In *The 39th Conference on Uncertainty in Artificial Intelligence*, 2023. 1

[17] Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized Inference for Causal Structure Learning. In *Advances in Neural Information Processing Systems*, pages 13104–13118. Curran Associates, Inc., 2022. 1

[18] Judea Pearl. *Causality*. Cambridge university press, 2009. 1

[19] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017. 1

[20] Aske Plaat. *Deep reinforcement learning*. Springer, 2022. 3

[21] Joseph Ramsey and Bryan Andrews. Py-Tetrad and RPy-Tetrad: A New Python Interface with R Support for Tetrad Causal Search. *In Causal Analysis Workshop Series PMLR*, pages 40–51, 2023. 2

[22] Andreas W M Sauter, Erman Acar, and Vincent Francois-Lavet. A Meta-Reinforcement Learning Algorithm for Causal Discovery. In *Proceedings of the Second Conference on Causal Learning and Reasoning*, pages 602–619. PMLR, 2023. 1

[23] Andreas W M Sauter, Nicolò Botteghi, Erman Acar, and Aske Plaat. CORE: Towards Scalable and Efficient Causal Discovery with Reinforcement Learning. *arXiv preprint arXiv:2401.16974*, 2024. 1

[24] Richard Scheines, Peter Spirtes, Clark Glymour, Christopher Meek, and Thomas Richardson. The TETRAD project: Constraint based aids to causal model specification. *Multivariate Behavioral Research*, 33(1):65–117, 1998. 2

[25] Amit Sharma and Emre Kiciman. DoWhy: An End-to-End Library for Causal Inference. 2020. 2

[26] Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000. 4

[27] Erdogan Taskesen. Learning Bayesian Networks with the bnlearn Python Package., 2020. 2

[28] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. *IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033, 2012. 2

[29] Christian Toth, Lars Lorch, Christian Knoll, Andreas Krause, Franz Pernkopf, Robert Peharz, and Julius von Kügelgen. Active Bayesian Causal Inference. In *Advances in Neural Information Processing Systems*, pages 16261–16275. Curran Associates, Inc., 2022. 1

[30] Mark Towers, Jordan K Terry, Ariel Kwiatkowski, John U Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G Younis. Gymnasium, 2023. 2

[31] Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor, and Kathleen Marchal. SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7(1):43, 2006. 2

[32] Matthew J. Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya Like DAGs? A Survey on Structure Learning and Causal Discovery. *ACM Computing Surveys*, 55(4): 1–36, 2023. 1

[33] Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. gCastle: A Python Toolbox for Causal Discovery. 2021. 2, 4

[34] Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P Xing. DAGs with NO TEARS: Continuous Optimization for Structure Learning. 2018. 4

[35] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal Discovery with Reinforcement Learning. *ArXiv*, abs/1906.04477, 2019. 1