Journal of Machine Learning Research 27 (2026) 1-42 Submitted 5/24; Published 6/26

Better Simulations for Validating Causal Discovery with the DAG-Adaptation of the Onion Method

Bryan Andrews ANDR1017@UMN.EDU
Institute for Health Informatics, University of Minnesota

Erich Kummerfeld ERICHK@UMN.EDU
Institute for Health Informatics, University of Minnesota

Editor: Jin Tian

Abstract

The number of methods for learning causal models from data is growing rapidly, as evidenced by the exponential increase in causal discovery publications each year. Due to a lack of real-world datasets with known causal ground truth, most "causal discovery" or "causal structure learning" algorithms are primarily validated with simulation studies. However, the simulation study designs being used (i) vary from one study to another, (ii) have never been formally characterized, and (iii) have not been proven to have desirable properties. As one might expect, publications often report conflicting performance statistics—even for simulation results on purely linear models. Appropriately, several manuscripts have now criticized the most common simulation designs.

We propose to resolve the above-mentioned challenges with a novel simulation design: the DAG-adaptation of the Onion (DaO) method. The DaO method takes a directed acyclic graph (DAG) as input and randomly generates coefficient and error-variance parameters of a linear model whose additive error terms may follow any distribution with finite variance. DaO simulations are fundamentally different from existing simulations because the DaO method prioritizes sampling the distribution of correlations rather than the distribution of linear effects. Specifically, the DaO method uniformly samples the space of correlation matrices Markov to a DAG. We compare the DaO method against two alternative simulation designs and provide implementations of the DaO method in Python and R: `https://github.com/bja43/DaO_simulation`; the Python implementation is available on PyPI: `https://pypi.org/project/daosim`. We posit that the DaO method is uniform, complete, well characterized, and fair, and should be adopted as a field simulation standard.

Keywords: Causal Discovery, DAG Models, Correlation, Structural Equation Models, Simulation, Validation, Uniform Sampling

1. Introduction

Learning causal relationships is central to many areas of scientific research, including medicine, climate science, economics, and psychology. Due to the difficulties of setting up proper randomized experiments and the growth in available non-experimental datasets, causal relationships are increasingly learned from observational data (Pearl, 2020). Another factor contributing to this trend is the increasing volume of new causal discovery algorithms (CDAs) published every year (Spirtes et al., 1993; Spirtes, 2010; Eberhardt, 2017; Malinsky and Danks, 2018; Glymour et al., 2019). CDAs output graphs intended

to accurately represent the causal relationships between the variables in a dataset, but lack finite-sample guarantees without unrealistic assumptions (Kalisch and Bühlman, 2007; Spirtes and Zhang, 2014; Uhler et al., 2013; Wang and Spirtes, 2022). Moreover, almost all observational datasets lack a comprehensive causal ground truth, making simulation studies the primary means of CDA validation (Ramsey and Andrews, 2017; Ramsey et al., 2020; Rios et al., 2021; Kummerfeld et al., 2023).

Unfortunately, CDA performance can heavily depend on simulation design and the field currently has no simulation standards. Without field standards: (i) algorithm developers can cherry-pick simulations that emphasize their method's strengths or their competitor's weaknesses and (ii) arbitrarily designed simulations may randomly favor certain CDAs over others for unknown reasons. As a consequence, several manuscripts warn that careless simulation practices have biased many well-cited studies (Reisach et al., 2021; Kaiser and Sipos, 2022; Reisach et al., 2023; Ng et al., 2024).

In this paper, we present a CDA simulation design as a potential field standard in the linear case with additive (not necessarily Gaussian) error: the DAG-adaptation of the Onion (DaO) method. The DaO method takes as input a directed acyclic graph (DAG) $\mathcal{G}$, and outputs a correlation matrix R. The matrix R is sampled uniformly at random over the space of all correlation matrices whose implied partial correlations satisfy the global Markov property for $\mathcal{G}$. The specific approach we use is adapted from the Onion method of Ghosh and Henderson (2003, 2009) by modifying it for DAG models. The correlation matrix R and DAG $\mathcal{G}$ define a recursive structural equation model (SEM) from which we can efficiently generate data. The graphs resulting of running CDAs on the simulated data can then be compared to $\mathcal{G}$ to evaluate performance.

The DaO method: (i) avoids prioritizing any correlation matrix over another, (ii) ensures that all possible correlation matrices are represented, and (iii) does not have tuning parameters that could be used to selectively benefit some CDAs over others. Altogether, this means that DaO offers a fair method for evaluating CDAs. These benefits necessarily come at a cost: the DaO method produces a distribution of models that may not reflect real-world causal systems. For example, data generated from DaO simulations do not resemble brain imaging data, climate science data, omics data, or electronic health record data. The DaO method is intended for generic domain-free simulations. This method could plausibly be extended to specific data domains in the future, but such an extension is outside the scope of this paper.

Paper Organization. We first summarize the contributions of this paper, consider the motivation for uniformly sampling the space of correlation matrices, and briefly review previous work on problematic simulation study designs for validating CDAs. Section 2 provides some necessary notation, definitions, and other background information. Section 3 presents the DaO method and relevant correctness proofs. Section 4 presents an empirical evaluation that (i) compares models produced by the DaO method against models produced by two other simulation designs, (ii) demonstrates pitfalls encountered by (at least one of) the other simulation designs, and (iii) demonstrates that the DaO method avoids these pitfalls. Section 5 demonstrates how a variety of CDAs perform on DaO simulations compared to the other two simulation designs. The paper concludes with a brief discussion in Section 6.

1.1 Contributions

Our novel contributions include the following:

1. We present a new method for simulating data from a given DAG: the DAG-adaptation of the Onion (DaO) method. The DaO method avoids several known problems with standard DAG model simulation designs.

2. We prove that the DaO method randomly samples correlation matrices uniformly over the space of all correlation matrices that satisfy the Markov property for the provided DAG.

3. We argue, based on DaO's proven theoretical properties, that the DaO method should be adopted as a field simulation standard.

4. We present simulation results that empirically demonstrate advantages of the DaO method over alternative designs.

5. We provide a novel explanation for why the simulation design used by many continuous optimization CDA publications report performance statistics in conflict with those from other simulation designs.

1.2 Previous Work

Other recently proposed simulations for validating CDAs have focused on modeling violations (Montagna et al., 2023), generating time-series data (Lawrence et al., 2021), and matching data from real-world systems (Tu et al., 2019). This paper targets the simple linear case where all modeling assumptions are met. Even in this simple case, there is disagreement on how to simulate data for validation. We intend DaO simulations to enable CDA method developers to establish a solid baseline for their algorithm(s).

Reisach et al. (2021) introduced the concept of *varsortability*. They observed that a common method for simulating data from a DAG popularized by Zheng et al. (2018) produces data where the causal order of the variables correlates with the marginal variance of the variables. Throughout this paper we refer to this simulation design by the authors of the paper that popularized it: Zheng, Aragam, Ravikuma, and Xing (ZARX). This is problematic because the marginal variance of a variable is scale dependent, making it an arbitrary quantity that should contain no information about the DAG. However, since the ZARX simulation design produces data with varsortability, marginal variance could in principle be used to gain information about the topological order of the DAG. Many CDAs are unaffected by marginal variance and thus would not make use of this information. Some CDAs, however, appear to be responsive to marginal variance. Such algorithms would have unrealistically strong performance in simulation studies the produce data with varsortability.

While varsortability can be removed by standardizing the data, Reisach et al. (2023) show in a followup paper that the ZARX simulation design has another sortability property that is not affected by rescaling the data: R^2*-sortability*. For R^2-sortability, the statistic of interest is, "the fraction of a variable's variance explained by all others, as captured by the coefficient of determination R^2." Many simulation designs produce data such that a

variable's ranking based on R^2 values is associated with its ranking in the DAG's topological order. Whether real world causal systems also exhibit R^2-sortability is an empirical question, however it is certainly plausible that common simulation designs have significantly more R^2-sortability than is realistic or reasonable.

These sortability concepts have also been discussed in relation to a recently developed class of DAG-learning algorithms that use continuous optimization to solve a differentiable loss function. In the first paper to actively criticize continuous optimization for learning DAGs, Kaiser and Sipos (2022) focused on the Non-combinatorial Optimization via Trace Exponential and Augmented lagRangian for Structure learning (NOTEARS) method (Zheng et al., 2018). They demonstrated through examples and theory that NOTEARS is not scale-invariant. In other words, the output of NOTEARS depends on the units of measurement for the variables in the dataset. This suggests that NOTEARS makes use of varsortability to achieve good performance. They also show that when the simulation design removes or reverses the varsortability, the performance of those methods can drop dramatically.

Ng et al. (2024) also investigated how the apparent performance of continuous optimization DAG-learning algorithms depends on simulation design. While they are critical of methods such as NOTEARS, they point to different issues than those raised by Kaiser and Reisach. In terms of simulation design, they indicate the use of equal versus non-equal variances in the simulated independent error terms as being responsible for the unusual simulation results (Peters and Bühlmann, 2014).

These works highlight a general problem with several existing simulation designs for evaluating DAG-learning algorithms. In particular, these designs can produce biased results by producing data distributions that associate irrelevant statistics with the DAG's structure (i.e., simulation artifacts). These designs elevate methods that can use the simulation artifacts to improve their DAG-learning performance, and punish methods that are neutral to or even negatively impacted by the simulation artifacts.

1.3 Why Simulate from DAGs by Sampling Correlation Matrices Uniformly?

The theoretical benefits of the DaO method fall into two categories: (1) advantages of directly sampling from the space of correlation matrices; (2) advantages of sampling uniformly at random. Collectively, these benefits support the conclusion that the DaO method is fair. Throughout this section, it is assumed that we wish to simulate data from some known DAG.

1.3.1 Advantages of Directly Sampling the Correlation Matrix

Existing linear simulation designs almost exclusively sample from prespecified distributions of edge weights and independent error terms. The sampled coefficient and error-variance parameters imply a (typically unknown or uncharacterized) distribution of correlation matrices. The DaO method takes the opposite approach: we prioritize sampling from a known distribution over the space of correlation matrices. The sampled correlation matrix implies a set of coefficient and error-variance parameters. This has some advantages, including the following.

First, a correlation matrix and sample size are sufficient statistics for the input to many CDAs. Variance is dependent on the units used for measurement, and as such has nothing to do with causality. As a consequence, for linear models any reasonable CDA should respond identically to a covariance matrix as it would to its corresponding correlation matrix or any other rescaling. As such, for a given DAG, it makes sense to consider how CDAs perform across the space of possible (i.e. Markov) correlation matrices. By prioritizing our control over the distribution of correlation matrices, we can directly interrogate performance across the space of possible CDA inputs.

Second, directly sampling correlation matrices will immediately and trivially prevent simulation artifacts like varsortability. A correlation matrix characterizes a distribution where all variables have unit variance, making varsortability impossible.

Third, the parametric models produced by directly sampling a correlation matrix are inherently standardized. Almost all other simulation study designs sample coefficient and error-variance parameters in a way that does not produce a standardized model. This can lead to situations where effect sizes (in terms of signal versus noise) are much stronger or weaker than one might expect, potentially even creating near-deterministic relationships; see Figures 3, 4, and 5. The only existing method we are aware of that directly samples standardized models has several restrictions which the DaO method does not have (Kummerfeld et al., 2023), such as requiring all of the coefficients in a model to be equal.

1.3.2 Advantages of Sampling Uniformly

After establishing a paradigm of directly sampling correlation matrices, it is intuitive to sample uniformly from this space. Uniform sampling across correlation matrices also has several benefits, including the following.

First, sampling from the uniform distribution of correlation matrices means that there are no input parameters or other design choices to make beyond specifying a DAG. This is convenient, prevents users from cherry-picking simulations, and ensures that simulations from different studies are consistent.

Second, uniform sampling across all correlation matrices means that no correlation matrices are neglected. All previous simulation designs that we are aware of omit portions of the possible correlation matrices and, as such, provide no findings about how CDAs perform on parts of the problem space. This could also be accomplished by another distribution with full support.

Third, for assessment of learning tasks, it is common to assume a generic condition where we lack knowledge about what to expect. When evaluating CDAs in a generic context (i.e., where we do not have a precise topic for which the CDA is intended), it is hard to say that having good performance on any particular correlation matrix is more important than another. This lack of knowledge in a generic context is accurately captured by a uniform distribution.

Altogether, sampling uniformly from the space of correlation matrices is fair and embodies a valuable benchmark. The CDAs that perform best in these simulations will be those that have the best average performance across all inputs. This is a desirable property for any general purpose CDA to have. No other distribution of linear models can be used to evaluate DAG-learning methods by that standard.

2. Background

This section provides background information on the relevant notation, models, and theory. The topics covered include directed acyclic graphs (DAGs), structural equation models (SEMs), the original Onion method, and the multivariate Pearson type II distribution.

Throughout this paper $V = \{1, 2, \dots\}$ denotes a non-empty finite set of positive integers that act as indices for collections of random variables and as vertices in graphical contexts. We denote the set of the first n positive integers as:

$$[\, n \,] = \{1, 2, \dots, n\}$$

and will sometimes refer to the elements of $[\, n \,]$ with the elements of V interchangeably. For matrices and vectors, we use the following notation:

- $\mathbb{R}^{V \times V}$: the space of $|V| \times |V|$ real-valued matrices;
- $\mathbb{S}^V_{++}$: the space of $|V| \times |V|$ symmetric positive definite matrices.

Let the following denote the space of all positive definite correlation matrices over V:

$$\text{Corr}^V_{++} := \{R \in \mathbb{S}^V_{++} \ : \ R_{i,i} = 1 \text{ for all } i \in V\}$$

2.1 Directed Acyclic Graphs

A *directed acyclic graph* (DAG) is an ordered pair $\mathcal{G} = (V, E)$ consisting of a finite vertex set V and an edge set of ordered pairs $E \subseteq V \times V$ with no *directed cycles*.[1] Ordered pairs of vertices in the edge set of a DAG are typically described using familial relationships. For $i \in V$:

$$\text{pa}_{\mathcal{G}}(i) \equiv \{j \in V \ : \ (i, j) \in E\} \qquad \text{ch}_{\mathcal{G}}(i) \equiv \{j \in V \ : \ (j, i) \in E\}$$

are the *parents* and *children* of i, respectively. Moreover, i is a *source* if it has no parents:

$$|\text{pa}_{\mathcal{G}}(i)| = 0.$$

A *total order* on V is a *reflexive*, *transitive*, *antisymmetric*, and *strongly connected* binary relation $\preceq : V \times V \to \{0, 1\}$; see Davey and Priestley (2002). The relationships between vertices in $\mathcal{G}$ can (partially) characterize a total order. A total order $\preceq$ is *consistent* with respect to $\mathcal{G}$ if for all $i, j \in V$:

$$i \in \text{pa}_{\mathcal{G}}(j) \quad \Rightarrow \quad i \preceq j.$$

In other words, $\preceq$ is consistent if every vertex precedes its children. A total order $\preceq$ is *source-first* with respect to $\mathcal{G}$ if for all $i, j \in V$:

$$|\text{pa}_{\mathcal{G}}(i)| = 0 \ \text{ and } \ |\text{pa}_{\mathcal{G}}(j)| \neq 0 \quad \Rightarrow \quad i \preceq j.$$

In other words, $\preceq$ is source-first if every source precedes every non-source. When it is obvious from context, we omit "with respect to $\mathcal{G}$" when describing an order as consistent and source-first. We adopt the following convention throughout this paper:

1. A *directed cycle* is a vertex sequence $\langle v_1, \dots, v_k \rangle$ $(k > 1)$ where $v_1 = v_k$ and $(v_i, v_{i+1}) \in E$ for all $i < k$.

Convention 1 *For any DAG $\mathcal{G} = (V, E)$ where $p = |V|$, we fix a total order $\preceq$ on V that is both source-first and consistent, and we label the vertices so that:*

$$V = [\,p\,] = \{1, \ldots, p\}, \qquad 1 \preceq 2 \preceq \cdots \preceq p.$$

Consequently,

$$i \preceq j \quad \Leftrightarrow \quad i \leq j.$$

This labeling is chosen separately for each graph and imposes no restrictions on the class of graphs considered.

2.2 Directed Acyclic Graph Models

A *DAG model* is a probabilistic models whose conditional independence relationships are explicitly represented by a DAG. In particular, DAGs graphically encode conditional independence relationships between their vertices, which can be read off using d-separation (Pearl, 1988; Verma and Pearl, 1990). A Markov property defines a subset of these relationships—usually by a simple graphical criterion—that is sufficient to imply all of conditional independence relationships represented by the DAG.

The *ordered Markov property* does so using a consistent total order (Lauritzen et al., 1990).[2] Accepting Convention 1, a probability distribution over V satisfies the ordered Markov property with respect to a DAG $\mathcal{G} = (V, E)$ if and only if:

$$i \perp\!\!\!\perp [\,i-1\,] \setminus \mathrm{pa}_{\mathcal{G}}(i) \mid \mathrm{pa}_{\mathcal{G}}(i) \quad \text{for all} \quad i \in V$$

where the ternary relation $A \perp\!\!\!\perp B \mid C$ for disjoint sets $A, B, C \subseteq V$ denotes that A and B are independent conditioned on C (Dawid, 1979). When a distribution satisfies the ordered Markov property with to respect $\mathcal{G}$, we say the distribution and its corresponding parametric model are *Markov to* $\mathcal{G}$. The distributions considered in the work belong to a family of DAG models called recursive structural equations models (SEMs) which are parameterized by covariance/correlation (Bollen, 1989).[3]

2.3 Recursive Structural Equation Models

A *recursive structural equation model* is a linear system of equations with no feedback loops or correlated error terms (Bollen, 1989). Let $X = \{X_i : i \in V\}$ and $Z = \{Z_i : i \in V\}$ be collections of random variables indexed by V that denote the model variables and additive error terms, respectively. The matrix $B = (\beta_{ij}) \in \mathbb{R}^{V \times V}$ contains the linear coefficients of the system and the additive errors are distributed according to $\mathcal{E}(\Omega)$, where $\mathcal{E}$ has finite covariance represented by the matrix $\Omega = (\omega_{ij}) \in \mathbb{S}^{V}_{++}$.

A recursive SEM for $\mathcal{G}$ is a distribution characterized by the system of linear equations:

$$X = XB^{\top} + Z \qquad\qquad Z \sim \mathcal{E}(\Omega) \tag{1}$$

2. Lauritzen et al. (1990) originally called this Markov property the local well-numbering Markov property.
3. General SEMs can model latent variables and feedback. These models can have more complex independence structures not compatible with DAGs (Bollen, 1989; Drton, 2018).

where

$$\begin{aligned} j \notin \text{pa}_{\mathcal{G}}(i) \quad &\Rightarrow \quad B_{i,j} = 0 \\ i \neq j \quad &\Rightarrow \quad \Omega_{i,j} = 0 \end{aligned} \tag{2}$$

for all $i, j \in V$. Let I denote the $|V| \times |V|$ identity matrix and accept Convention 1. The linear system in Equation 1 is uniquely solved by $X = Z\left(I - B\right)^{-\top}$ and has covariance:

$$\Sigma = \left(I - B\right)^{-\top} \Omega \left(I - B\right)^{-1}$$

Conversely, for $i \in V$ and $J = [\, i-1 \,]$, the linear coefficients and additive error variance are computed:

$$B_{i,J} = \Sigma_{i,J} \left(\Sigma_{J,J}\right)^{-1} \tag{3}$$

$$\Omega_{i,i} = \Sigma_{i,i} - \Sigma_{i,J} \left(\Sigma_{J,J}\right)^{-1} \Sigma_{J,i} \tag{4}$$

respectively. Typically one takes $J = \text{pa}_{\mathcal{G}}(i)$, however since partial correlation and conditional independence are equivalent for recursive SEMs, the ordered Markov property implies that Equations 3 and 4 are valid for any J satisfying $\text{pa}_{\mathcal{G}}(i) \subseteq J \subseteq [\, i-1 \,]$.

Let $\mathcal{G} = (V, E)$ be a DAG and $R \in \text{Corr}^{V}_{++}$ be a correlation matrix over a V. We say R is "Markov" with respect to $\mathcal{G}$ if there exist $B \in \mathbb{R}^{V \times V}$ and $\Omega \in \mathbb{S}^{V}_{++}$ such that:

$$R = \left(I - B\right)^{-\top} \Omega \left(I - B\right)^{-1}$$

and Equation 2 holds for all $i, j \in V$. Let the following denote the space of positive definite correlation matrices Markov to $\mathcal{G}$:

$$\mathcal{M}(\mathcal{G}) := \{R \in \text{Corr}^{V}_{++} \ : \ R \text{ is Markov with respect to } \mathcal{G}\}$$

2.4 The Onion Method

Ghosh and Henderson (2003, 2009) proposed the Onion method for uniformly sampling positive definite correlation matrices. Their approach sets up a recurrence relation with base case:

$$R_1 = [\, 1 \,]$$

and where the $(i+1) \times (i+1)$ correlation matrix R_{i+1} is sampled by appending a column vector r_{i+1} and its transpose $r_{i+1}^{\top}$ to the rows and columns of an existing $i \times i$ correlation matrix R_i:

$$R_{i+1} = \begin{bmatrix} R_i & r_{i+1} \\ r_{i+1}^{\top} & 1 \end{bmatrix}$$

Ghosh and Henderson call r_{i+1} the the completion of R_i in R_{i+1} and derive a necessary and sufficient condition using Schur's complement for R_{i+1} to be positive definite:

Lemma 1 *If R_i is positive definite, then R_{i+1} is positive definite if and only if:*

$$1 - r_{i+1}^{\top} R_i^{-1} r_{i+1} > 0.$$

Ghosh and Henderson named their method the Onion method because it builds the sampled correlation matrix one layer at a time. Lewandowski et al. (2009) reformulated the Onion method in terms of elliptical distributions. Their reformulation relies on the multivariate Pearson type II distribution which we rely on as well.

2.5 Multivariate Pearson Type II

$W \sim mPII_d(\gamma)$ follows a multivariate Pearson type II distribution if it has density:[4]

$$f(w \mid \gamma) \propto (1 - w^\top w)^{\gamma - \frac{1}{2}} \tag{5}$$

where $\gamma > -\frac{1}{2}$ and $w \in \mathbb{R}^d$ such that $w^\top w \leq 1$. Alternatively, $W = Q^{\frac{1}{2}} U$ where:

$$Q \sim beta(\tfrac{d}{2}, \gamma + \tfrac{1}{2});$$
$$U \sim uniform(\{u \in \mathbb{R}^d \ : \ \|u\|_2 = 1\}).$$

In the alternative formulation, U is distributed uniformly on the surface of the unit d-sphere; for more details see Fang et al. (1990) and Khalafi and Azimmohseni (2014).

3. Methods

This section presents the primary content of this paper, the DaO method for sampling correlation matrices. This includes theoretical results that the distribution of matrices produced by the DaO method are uniformly sampled from the space of all correlation matrices Markov to an input DAG. We first present the intuition for generalizing the Onion method to the DaO method. Pseudocode and proofs of correctness for the DaO method are provided in Section 3.2.

3.1 Generalizing the Onion Method to DAGs

To create the DaO method, we adapt the Onion method to sample uniformly from the space of correlation matrices Markov to a given DAG. We also compute and store the corresponding B and Ω matrices, which are sampled in tandem with the correlation matrix.

Let $\mathcal{G} = (V, E)$ be a DAG and accept Convention 1 so that $V = \{1, \ldots, p\}$ and $\preceq$ is a source first consistent order. The DaO method starts from three $m \times m$ matrices, where $m = |\{i \in V \ : \ \text{pa}_\mathcal{G}(i) = \varnothing\}|$ is the number of source vertices in $\mathcal{G}$:

$$R_m = I_{m \times m} \qquad B_m = 0_{m \times m} \qquad \Omega_m = I_{m \times m}$$

The DaO method builds a correlation matrix (and corresponding B and Ω matrices) by adding one additional layer at a time:

$$R_{i+1} = \begin{bmatrix} R_i & r_{i+1} \\ r_{i+1}^\top & 1 \end{bmatrix} \qquad B_{i+1} = \begin{bmatrix} B_i & 0_i \\ b_{i+1}^\top & 0 \end{bmatrix} \qquad \Omega_{i+1} = \begin{bmatrix} \Omega_i & 0_i \\ 0_i^\top & \omega_{i+1} \end{bmatrix}$$

where $I_{m \times m}$ and $0_{m \times m}$ denote the $m \times m$ identity matrix and zero matrix respectively, and 0_i denotes a column vector of i zeros. Following Ghosh and Henderson, we call $b_{i+1}^\top$ the completion of B_i in B_{i+1} and ω_{i+1} the completion of Ω_i in Ω_{i+1}. By Equations 3 and 4:

$$b_{i+1}^\top = r_{i+1}^\top R_i^{-1} \tag{6}$$
$$\omega_{i+1} = 1 - r_{i+1}^\top R_i^{-1} r_{i+1} \tag{7}$$

Let R_i and $k = |\text{pa}_\mathcal{G}(i+1)|$. The completion r_{i+1} is sampled using a change of variable using the following:

4. The multivariate Pearson type II distribution is usually defined: $f(w \mid a) \propto (1 - w^\top w)^{a-1}$. We use $\gamma = a - \frac{1}{2}$ to simplify calculations in Section 3.1.

- Let $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ be an i-dimensional column vector where $w_1 \sim mPII_k$ is column vector of k draws from a multivariate Pearson type II distribution and $w_2 = 0_{(i-k)}$ is a column vector of $i-k$ zeros.
- Let P be an $i \times i$ permutation matrix such that the first k rows and columns correspond to the members of $\text{pa}_{\mathcal{G}}(i+1)$ and the remaining $i-k$ rows and columns correspond to the members of $[\,i\,] \setminus \text{pa}_{\mathcal{G}}(i+1)$. In other words, P permutes R_i so that parent vertices precede non-parent vertices.
- Let $LL^\top = P^\top R_i P$ be the Cholesky decomposition so that $R_i = PLL^\top P^\top$.

The completion of R_i in R_{i+1} is then sampled as $r_{i+1} = PLw$. Moreover, using the orthogonality of P, the completions defined in Equations 6 and 7 can be reformulated in terms of w:

$$\begin{aligned} b_{i+1}^\top &= r_{i+1}^\top R_i^{-1} \\ &= (PLw)^\top (PLL^\top P^\top)^{-1} \\ &= w^\top L^\top P^\top P L^{-\top} L^{-1} P^\top \\ &= w^\top L^{-1} P^\top \end{aligned} \tag{8}$$

$$\begin{aligned} \omega_{i+1} &= 1 - r_{i+1}^\top R_i^{-1} r_{i+1} \\ &= 1 - (PLw)^\top (PLL^\top P^\top)^{-1} (PLw) \\ &= 1 - w^\top L^\top P^\top P L^{-\top} L^{-1} P^\top PLw \\ &= 1 - w^\top w \end{aligned} \tag{9}$$

Notice the first line on the right hand side of Equation 9 is the same as the left hand side of the inequality in Lemma 1. Accordingly, the positive definite requirement from the lemma is equivalent to requiring that the additive error terms have positive variance ($\omega_{i+1} > 0$). By forbidding the variances of these terms from equaling zero ($\omega_{i+1} = 0$), the DaO method excludes recursive SEMs with deterministic relationships from its support. Indeed, the set of excluded models (those with deterministic relationships) are exactly the those that imply positive semi-definite correlation matrices that are not positive definite.

We apply the change of variable $w = (PL)^{-1} r_{i+1}$ to derive the density of r_{i+1}. Since $(1 - w_1^\top w_1) = (1 - w^\top w)$, we can express the density of w in the same form as Equation 5: $f(w \mid \gamma) \propto (1 - w^\top w)^{\gamma - \frac{1}{2}}$. Moreover, PL is invertible since P is an orthogonal permutation matrix and L is a postive definite Cholesky factor. Applying the change of variable $w = (PL)^{-1} r_{i+1}$ to Equation 5, r_{i+1} has density:

$$\begin{aligned} f(r_{i+1} \mid R_i \,;\, \mathcal{G}, \gamma_i) &\propto \left[1 - ((PL)^{-1} r_{i+1})^\top ((PL)^{-1} r_{i+1})\right]^{\gamma - \frac{1}{2}} \; \left|\det((PL)^{-1})\right| \\ &= \left[1 - r_{i+1}^\top (PLL^\top P^\top)^{-1} r_{i+1}\right]^{\gamma - \frac{1}{2}} \; \det(PLL^\top P^\top)^{-\frac{1}{2}} \\ &= (1 - r_{i+1}^\top R_i^{-1} r_{i+1})^{\gamma - \frac{1}{2}} \; \det(R_i)^{-\frac{1}{2}} \end{aligned} \tag{10}$$

The DaO method samples completions from the density above by initializing $\gamma_m = \frac{p-m}{2}$ and updating $\gamma_{i+1} = \gamma_i - \frac{1}{2}$.

3.2 The DAG-adaptation of the Onion Method

Algorithm 3 (`DaO`) provides pseudocode for the DaO method which employs two subroutines:

- Algorithm 1 (`mPII`) takes DAG $\mathcal{G}$, integer/vertex i, and parameter γ as input. Let $n = i - 1$ and $k = |\mathrm{pa}_{\mathcal{G}}(i)|$. The algorithm returns an n-dimensional column vector where the first k elements are drawn from a multivariate Pearson type II distribution and the remaining $n - k$ elements are zero.

- Algorithm 2 (`P-mat`) takes DAG $\mathcal{G}$ and integer/vertex i as input. Let $n = i - 1$ and $k = |\mathrm{pa}_{\mathcal{G}}(i)|$. The algorithm returns an $n \times n$ permutation matrix P such that the first k rows and columns correspond to the members of $\mathrm{pa}_{\mathcal{G}}(i)$ and the remaining $n - k$ rows and columns correspond to the members of $[\, i-1 \,] \setminus \mathrm{pa}_{\mathcal{G}}(i)$. In other words, P permutes R_{i-1} so that parent vertices precede non-parent vertices.

In Algorithm 1 (`mPII`): `beta`(a, b) samples a beta distribution with parameters a and b; `randn`(k) samples a k-dimensional standard normal distribution; and lines 4 and 5 together uniformly samples the surface of the unit k-sphere using a procedure outlined by Muller (1956).

Algorithm 1: `mPII`$(\mathcal{G}, i, \gamma)$

Input: `DAG` : $\mathcal{G}$ `idx` : i `param` : γ
Output: `vector` : w

1 $n \leftarrow i - 1$
2 $k \leftarrow |\mathrm{pa}_{\mathcal{G}}(i)|$
3 $q \sim$ `beta`$(\frac{k}{2},\ \gamma + \frac{1}{2})$ // $q \neq 1$
4 $y \sim$ `randn`(k) // $\|y\|_2 \neq 0$
5 $u \leftarrow \dfrac{y}{\|y\|_2}$
6 $w \leftarrow \begin{bmatrix} q^{\frac{1}{2}}\, u \\ 0_{n-k} \end{bmatrix}$

Algorithm 2: `P-mat`$(\mathcal{G}, i)$

Input: `DAG` : $\mathcal{G}$ `idx` : i
Output: `matrix` : P

1 $n \leftarrow i - 1$
2 $P \leftarrow 0_{n \times n}$; $j \leftarrow 1$
3 **foreach** $k \in \mathrm{pa}_{\mathcal{G}}(i)$ **do**
4 $\quad P_{j,k} \leftarrow 1$
5 $\quad j \leftarrow j + 1$
6 **foreach** $k \in [\, i-1 \,] \setminus \mathrm{pa}_{\mathcal{G}}(i)$ **do**
7 $\quad P_{j,k} \leftarrow 1$
8 $\quad j \leftarrow j + 1$

In Algorithms 1 and 2: $0_{n \times n}$ denotes the $n \times n$ zero matrix and 0_{n-k} denotes a column vector of $n - k$ zeros. We assume the support of `beta`(a, b) is $[\, 0, 1 \,]$ however we reject and redraw samples where $q = 1$. This would result in determinism and a non-positive definite (positive semi-definite) correlation matrix. Notably, sampling $q = 1$ occurs with Lebesgue measure zero and will not happen in practice. We do allow samples where $q = 0$ (these are unfaithful distributions) but note this also occurs with Lebesgue measure zero and will not happen in practice. Indeed, it is known that unfaithful distributions occur with Lebesgue measure zero in Gaussian DAG models (Meek, 1995). Similarly, sampling $y = 0_k$ (a column vector of k zeros) from `randn`(k) would result in a divide by zero error but occurs with Lebesgue measure zero and will not happen in practice. Accordingly, we reject and redraw samples where $y = 0_k$.

Algorithm 3: `DaO`$(\mathcal{G})$

Input: `DAG` : $\mathcal{G} = (V, E)$ s.t. $|V| = p$

Output: `corr` : R_p `coef` : B_p `err-var` : Ω_p

$m = |\{i \in V \ : \ \text{pa}_{\mathcal{G}}(i) = \varnothing\}|$

$R_m \leftarrow I_{m \times m}$; $B_m \leftarrow 0_{m \times m}$; $\Omega_m \leftarrow I_{m \times m}$; $\gamma_m \leftarrow \frac{p-m}{2}$

for $i \leftarrow m$ **to** $p-1$ **do**

$w \sim$ `mPII`$(\mathcal{G}, i+1, \gamma_i)$

$P \leftarrow$ `P-mat`$(\mathcal{G}, i+1)$

$LL^\top \leftarrow P^\top R_i \, P$ `// Cholesky decomposition`

$$R_{i+1} \leftarrow \begin{bmatrix} R_i & r_{i+1} \\ r_{i+1}^\top & 1 \end{bmatrix} \quad \text{where} \quad r_{i+1} = P\,L\,w$$

$$B_{i+1} \leftarrow \begin{bmatrix} B_i & 0_i \\ b_{i+1}^\top & 0 \end{bmatrix} \quad \text{where} \quad b_{i+1}^\top = w^\top L^{-1} P^\top$$

$$\Omega_{i+1} \leftarrow \begin{bmatrix} \Omega_i & 0_i \\ 0_i^\top & \omega_{i+1} \end{bmatrix} \quad \text{where} \quad \omega_{i+1} = 1 - w^\top w$$

$\gamma_{i+1} \leftarrow \gamma_i - \frac{1}{2}$

In Algorithm 3: $I_{m \times m}$ and $0_{m \times m}$ denote the $m \times m$ identity matrix and zero matrix respectively while 0_i denotes a column vector of i zeros. Let $\mathcal{G} = (V, E)$ be a DAG. Let the following denote the space of positive definite correlation matrices sampled by the DaO method given $\mathcal{G}$:

$$\mathcal{D}(\mathcal{G}) := \{R \in \text{Corr}_{++}^V \ : \ R \text{ can be sampled from } \texttt{DaO}(\mathcal{G})\}$$

Remark 2 *Lewandowski et al. (2009) give an extension of the onion method to sample correlation matrices proportional to a power of their determinant by choosing different values for the parameter we call γ. The same extension could be applied here. However, we prioritized minimizing the number of arguments and pick γ to sample correlation matrices uniformly; see Theorem 5.*

Figure 1 depicts the results of generating random correlation matrices using the DaO method for three DAGs. The DAGs make up the simplest non-trivial cases for the DaO Method and the dimensions of the plots represent the correlations in the matrix:

$$R = \begin{bmatrix} 1 & r_{12} & r_{13} \\ r_{12} & 1 & r_{23} \\ r_{13} & r_{23} & 1 \end{bmatrix}$$

We prove the correctness of the DaO method in two theorems. Theorem 4 shows that the DaO method samples the space of positive definite correlation matrices Markov to a DAG and Theorem 5 show that the DaO method samples that space uniformly. We use two lemmas to facilitate the proofs of these theorems: Lemma 1 gives a necessary and sufficient condition for a completion to produce a positive definite matrix and Lemma 3 gives a necessary and sufficient for the trailing zeroes in the column vector sampled from `mpii` to correspond to non-parents in the graph.

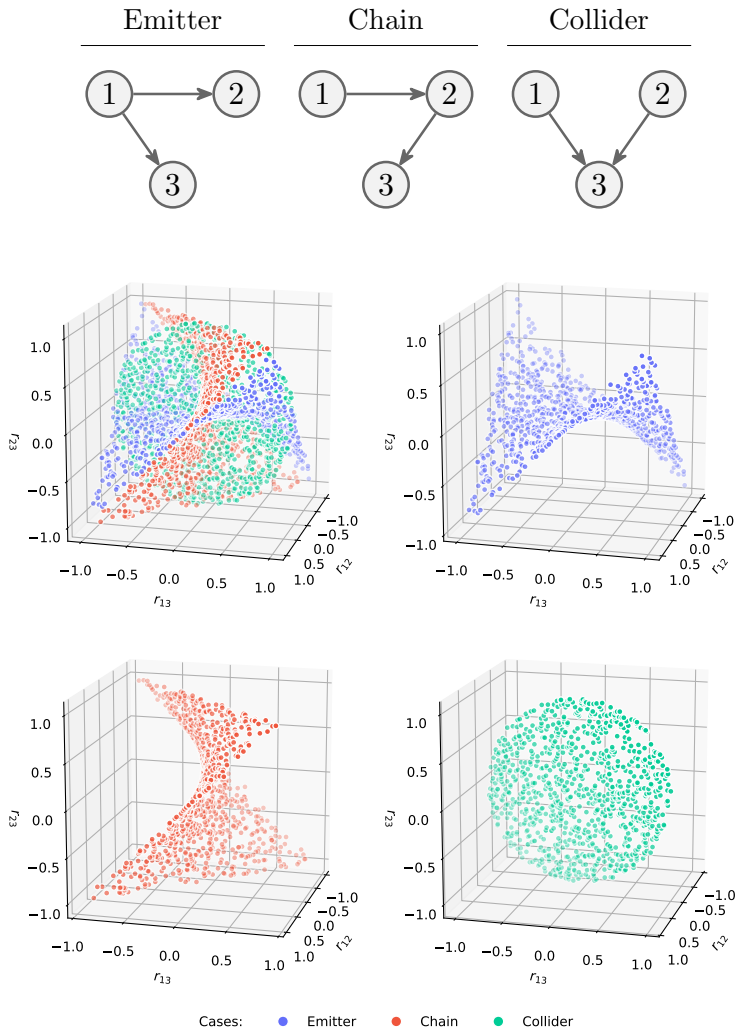

Figure 1: Uniformly sampled correlation matrices from DAGs with $|V| = 3$ and $|E| = 2$ with 100 repetition for each case.

Lemma 3 *Let $\mathcal{G} = (V, E)$ be a DAG and accept Convention 1. If w is the column vector sampled on line 4 of Algorithm 3 and $b_{i+1}^\top$ is the row vector sampled on line 8 of Algorithm 3, then the following are equivalent:*

- $(b_{i+1}^\top)_j = 0$ for all $j \notin \text{pa}_\mathcal{G}(i+1)$
- $(w^\top)_j = 0$ for all $j > |\text{pa}_\mathcal{G}(i+1)|$

Proof By Equation 8 and using the orthogonality of P:

$$b_{i+1}^\top P = w^\top L^{-1}$$

Moreover, because L^{-1} is lower triangular:

$$(L^{-1})_{k,\,j} = 0 \text{ for all } k < i$$

Accordingly, the j^{th} element of $b_{i+1}^\top P$ is computed:

$$(b_{i+1}^\top P)_j = \sum_{k=j}^{i} (w^\top)_k \, (L^{-1})_{k,\,j}$$

Note that L^{-1} is the inverse of a Cholesky factor and is therefore a positive definite matrix with real and positive diagonal elements. Recall P is a permutation matrix that permutes R_i so that parent vertices precede non-parent vertices. Likewise, P permutes $b_{i+1}^\top P$ so that parent vertices precede non-parent vertices. Accordingly, the following are equivalent:

- $(b_{i+1}^\top)_j = 0$ for all $j \notin \text{pa}_\mathcal{G}(i+1)$
- $(b_{i+1}^\top P)_j = 0$ for all $j > |\text{pa}_\mathcal{G}(i+1)|$

and we may prove the lemma by showing:

- $(b_{i+1}^\top P)_j = 0$ for all $j > |\text{pa}_\mathcal{G}(i+1)|$
- $(w^\top)_j = 0$ for all $j > |\text{pa}_\mathcal{G}(i+1)|$

The proof is completed using induction:

Base case: If $i = j$, then:

$$(b_{i+1}^\top P)_j = (w^\top)_j (L^{-1})_{j,j}$$

However, $(L^{-1})_{j,j} > 0$ which implies $(b_{i+1}^\top P)_j = 0$ if and only if $(w^\top)_j = 0$.

Inductive step: If $i > j$ and $(b_{i+1}^\top P)_k = (w^\top)_k = 0$ for all $k > j$, then:

$$(b_{i+1}^\top P)_j = \sum_{k=j}^{i} (w^\top)_k \, (L^{-1})_{k,\,j} = (w^\top)_j (L^{-1})_{j,j}$$

However, $(L^{-1})_{j,j} > 0$ which implies $(b_{i+1}^\top P)_j = 0$ if and only if $(w^\top)_j = 0$.
Accordingly, the lemma is proven by induction. ■

Theorem 4 *Let $\mathcal{G} = (V, E)$ be a DAG and accept Convention 1. The space of positive definite correlation matrices sampled by the DaO method given $\mathcal{G}$ and the space of positive definite correlation matrices Markov to $\mathcal{G}$ are equivalent: $\mathcal{D}(\mathcal{G}) \equiv \mathcal{M}(\mathcal{G})$.*

Proof The theorem is proven in two parts: $D(\mathcal{G}) \subseteq \mathcal{M}(\mathcal{G})$ and $\mathcal{M}(\mathcal{G}) \subseteq \mathcal{D}(\mathcal{G})$. Throughout this proof, if $A \in \mathbb{R}^{V \times V}$ is a matrix, then A_i denotes the upper-left $i \times i$ sub-matrix of A.

$D(\mathcal{G}) \subseteq \mathcal{M}(\mathcal{G})$:

Pick an arbitrary $R \in \mathcal{D}(\mathcal{G})$. We prove the first part of the theorem by showing $R \in \mathcal{M}(\mathcal{G})$ with induction:

Base case: Let $m = |\{i \in V \ : \ \mathrm{pa}_{\mathcal{G}}(i) = \varnothing\}|$ be the number of source vertices in $\mathcal{G}$ and construct:

$$R_m = I_{m \times m} \qquad B_m = 0_{m \times m} \qquad \Omega_m = I_{m \times m}$$

where $I_{m \times m}$ and $0_{m \times m}$ denote the $m \times m$ identity matrix and zero matrix respectively. $R_m = (I - B_m)^{-\top} \Omega_m (I - B_m)^{-1}$ is positive definite and Equation 2 holds for all $i, j \in [\, m \,]$ by construction.

Inductive step: Assume $R_i = (I - B_i)^{-\top} \Omega_i (I - B_i)^{-1}$ is positive definite and Equation 2 (replacing i, j with j, k) holds for all $j, k \in [\, i \,]$. Let w be the column vector sampled on line 4 of Algorithm 3, $r_{i+1}^{\top}$ be the column vector sampled on line 7 of Algorithm 3, $b_{i+1}^{\top}$ be the row vector sampled on line 8 of Algorithm 3, and $\omega_{i+1}^{\top}$ be the scaler sampled on line 8 of Algorithm 3. If we construct:

$$R_{i+1} = \begin{bmatrix} R_i & r_{i+1} \\ r_{i+1}^{\top} & 1 \end{bmatrix} \qquad B_{i+1} = \begin{bmatrix} B_i & 0_i \\ b_{i+1}^{\top} & 0 \end{bmatrix} \qquad \Omega_{i+1} = \begin{bmatrix} \Omega_i & 0_i \\ 0_i^{\top} & \omega_{i+1} \end{bmatrix}$$

where 0_i denotes a column vector of i zeros, then $R_{i+1} = (I - B_{i+1})^{-\top} \Omega_{i+1} (I - B_{i+1})^{-1}$ is positive definite by Lemma 1 and Equation 2 (replacing i, j with j, k) holds for all $j, k \in [\, i + 1 \,]$ by Lemma 3.

Accordingly, R is positive definite and Equation 2 holds for all $i, j \in V$ by induction which proves the first part of the theorem.

$\mathcal{M}(\mathcal{G}) \subseteq \mathcal{D}(\mathcal{G})$:

Pick an arbitrary $R \in \mathcal{M}(\mathcal{G})$ and construct B and Ω such that $R = (I - B)^{-\top} \Omega (I - B)^{-1}$ where the entries of B and Ω are uniquely solved for by:

$$B_{i,J} = R_{i,J} (R_{J,J})^{-1}$$
$$\Omega_{i,i} = R_{i,i} - R_{i,J} (R_{J,J})^{-1} R_{J,i}$$

for $i \in V$ and $J = [\, i - 1 \,]$ and equal to zero otherwise. We prove the second part of the theorem by showing $R \in \mathcal{D}(\mathcal{G})$ with induction:

Base case: Let $m = |\{i \in V \ : \ \mathrm{pa}_{\mathcal{G}}(i) = \varnothing\}|$ be the number of source vertices in $\mathcal{G}$ and note:

$$B_m = 0_{m \times m} \qquad \Omega_m = I_{m \times m}$$

by construction where $I_{m\times m}$ and $0_{m\times m}$ denote the $m \times m$ identity matrix and zero matrix respectively. Accordingly, running Algorithm 3 until it executes line 2 can produce B_m and Ω_m.

Inductive step: Assume running Algorithm 3 until the loop starting on line 3 has repeating the $i-m$ times can produce B_i and Ω_i. Let $J = [\,i+1\,]$, $b_{i+1}^\top = B_{i+1,J}$, and $\omega_{i+1} = \Omega_{i+1,i+1}$. Running Algorithm 3 until the loop starting on line 3 has repeating an additional time can produce:

$$B_{i+1} = \begin{bmatrix} B_i & 0_i \\ b_{i+1}^\top & 0 \end{bmatrix} \qquad\qquad \Omega_{i+1} = \begin{bmatrix} \Omega_i & 0_i \\ 0_i^\top & \omega_{i+1} \end{bmatrix}$$

by Lemmas 1 and 3 where 0_i denotes a column vector of i zeros.

Accordingly, the second part of the theorem is proven by induction.

Since $\mathcal{D}(\mathcal{G}) \subseteq \mathcal{M}(\mathcal{G})$ and $\mathcal{M}(\mathcal{G}) \subseteq \mathcal{D}(\mathcal{G})$ we have $\mathcal{D}(\mathcal{G}) \equiv \mathcal{M}(\mathcal{G})$. ■

Theorem 5 *Let $\mathcal{G} = (V, E)$ be a DAG and accept Convention 1. If $f : \mathcal{D}(\mathcal{G}) \to \mathbb{R}$ is the density induced by the DaO method, then $f \propto 1$.*

Proof Let $R_i \in \text{Corr}_{++}^{[i]}$ and note the following relation derived from Schur's compliment (Ouellette, 1981):

$$\begin{aligned} \det(R_{i+1}) &= \det(R_i)\det(1 - r_{i+1}^\top R_i^{-1} r_{i+1}) \\ &= (1 - r_{i+1}^\top R_i^{-1} r_{i+1})\det(R_i) \end{aligned}$$

The density of r_{i+1} conditioned on R_i in Equation 10 is reformulated using the relation above and by noting that $\gamma_{i+1} = \gamma_i - \frac{1}{2}$:

$$\begin{aligned} f(r_{i+1} \mid R_i\ ;\ \mathcal{G}, \gamma_i) &\propto (1 - r_{i+1}^\top R_i^{-1} r_{i+1})^{\gamma_i - \frac{1}{2}}\ \det(R_i)^{-\frac{1}{2}} \\ &= (1 - r_{i+1}^\top R_i^{-1} r_{i+1})^{\gamma_i - \frac{1}{2}}\ \det(R_i)^{-\frac{1}{2}}\ \frac{\det(R_i)^{\gamma_i}}{\det(R_i)^{\gamma_i}} \\ &= \frac{\left[(1 - r_{i+1}^\top R_i^{-1} r_{i+1})\ \det(R_i)\right]^{\gamma_i - \frac{1}{2}}}{\det(R_i)^{\gamma_i}} \\ &= \frac{\det(R_{i+1})^{\gamma_{i+1}}}{\det(R_i)^{\gamma_i}} \end{aligned}$$

Noting $f(R_{i+1}\ ;\ \mathcal{G}, \gamma_{i+1}) = f(r_{i+1} \mid R_i\ ;\ \mathcal{G}, \gamma_i)\ f(R_i\ ;\ \mathcal{G}, \gamma_i)$ and $f(R_m\ ;\ \mathcal{G}, \gamma_m) = 1$ since $R_m = I_{m\times m}$ where $m = |\{j \in V\ :\ \text{pa}_{\mathcal{G}}(j) = \varnothing\}|$ is constant:

$$\begin{aligned} f(R_p\ ;\ \mathcal{G}, \gamma_p) &= \prod_{i=m}^{p-1} f(r_{i+1} \mid R_i\ ;\ \mathcal{G}, \gamma_i)\ f(R_m\ ;\ \mathcal{G}, \gamma_m) \\ &\propto \prod_{i=m}^{p-1} \frac{\det(R_{i+1})^{\gamma_{i+1}}}{\det(R_i)^{\gamma_i}}\ \det(R_m)^{\gamma_m} \\ &= \det(R_p)^{\gamma_p} \end{aligned}$$

The DaO method sets $\gamma_m = \frac{p-m}{2}$ which makes $\gamma_p = 0$. Accordingly, $f \propto 1$. ■

4. Characterization of DaO and Two Other Simulation Designs

This section characterizes the distribution of models generated by the DaO method. In particular we describe the DaO method's output in terms of the distribution of beta coefficients; the distribution of error-variances parameters; and the R^2-sortability of the variables. In terms of graph structures, the following characterizations cover three different DAG types: Erdős Rényi (ER), scale-free in-degree (SFi), and scale-free out-degree (SFo).

Because DaO constructs standardized models, all variables in DaO models have unit marginal variance. As such, all beta coefficients represent variability transferred from one standardized variable to another standardized variable, and all independent error terms have variance equal to one minus the total variance induced by its ancestors in the model. By plotting the distribution of the beta coefficients and error-variance parameters, we can assess the distribution of statistical effect sizes that one is likely to find in DaO simulations. To provide additional context, we also characterize two other simulation designs in the same way. We standardize all models using Algorithms 8 and 9 for both comparison designs to ensure apples-to-apples comparisons; see Appendix A.3.

Since these are standardized designs, one outcome we evaluate is how often models contain extreme parameter values. Two types of extreme values are (i) betas that are near or above one, since they indicate extremely high statistical effect sizes where two variables are almost deterministically linked; and (ii) independent variances that are almost zero, since they indicate that a variable's values are almost deterministically set by its parents.

We do not consider betas near zero to be problematic, since that is just a matter of weak effect sizes that will require a larger sample size to detect. DaO does technically permit betas being sampled at precisely zero, as well as other distributions that would violate the Faithfulness assumption made by most CDAs. However, sampling any distribution that violates Faithfulness is a probability zero event. As such, this does not happen when producing finite models. All methods, including DaO and the comparison methods, have a nontrivial chance of sampling models that are almost Unfaithful (Uhler et al., 2013). This is normal, and is not problematic. Real-world situations are entirely capable of existing near Unfaithful distributions as well. Further, some methods are more capable of handling Unfaithful and near-Unfaithful distributions than others, and it is in part the purpose of simulations to help determine which methods are more robust against expected amounts and types of Unfaithfulness (Lam et al., 2022; Andrews et al., 2023).

4.1 Simulation Designs Used for Comparison

We compare the DaO method against two alternative simulation designs. Both designs also parameterize B and Ω directly, where B is a matrix of beta coefficients and Ω is a covariance matrix for additive error-variance parameters.

- Zheng, Aragam, Ravikuma, and Xing (ZARX) simulation (Zheng et al., 2018):[5]

$$B = (\beta_{i,j}) \ : \ \begin{cases} \beta_{i,j} \sim \textit{uniform}(\,[-2.0,\,-0.5\,] \cup [\,0.5,\,2.0\,]\,) & \text{if } j \in \text{pa}_{\mathcal{G}}(i) \\ \beta_{i,j} = 0 & \text{if } j \notin \text{pa}_{\mathcal{G}}(i) \end{cases}$$

$$\Omega = (\omega_{i,j}) \ : \ \begin{cases} \omega_{i,j} = 1 & \text{if } i = j \\ \omega_{i,j} = 0 & \text{if } i \neq j \end{cases}$$

- Tetrad simulation (Andrews et al., 2023):[6]

$$B = (\beta_{i,j}) \ : \ \begin{cases} \beta_{i,j} \sim \textit{uniform}(\,[-1.0,\,1.0\,]\,) & \text{if } j \in \text{pa}_{\mathcal{G}}(i) \\ \beta_{i,j} = 0 & \text{if } j \notin \text{pa}_{\mathcal{G}}(i) \end{cases}$$

$$\Omega = (\omega_{i,j}) \ : \ \begin{cases} \omega_{i,j} \sim \textit{uniform}(\,[\,1.0,\,2.0\,]\,) & \text{if } i = j \\ \omega_{i,j} = 0 & \text{if } i \neq j \end{cases}$$

We "standardize" the models produced by the alternative methods to facilitate a meaningful comparison. This ensures all variables have unit variance and that we compare scale independent versions of the models. The methods above are "standardized" using Algorithms 8 and 9; see Appendix A.3. Algorithm 8 (`Cov-to-Corr`) converts a covariance matrix to a correlation matrix and Algorithm 9 (`Cov-to-DAG`) takes a DAG and covariance/correlation matrix as input and returns the corresponding B and Ω parameters for the model—the parameters are standardized if the input is a DAG and a correlation matrix.

4.2 Comparison of Simulation Designs

Figure 2 plots the magnitude of the standardized betas against the standard deviation of the additive error term for every edge, with kernel density estimates for contours. Figures 3, 4, and 5 depict empirical cumulative distribution functions (ECDFs) of the absolute beta coefficients, absolute correlations, and standard deviations of 100 models generated from DAGs with 100 variables and average degree $\alpha = 10$. The values where the corresponding implied correlation is zero or one are omitted and vertical dotted lines indicate where the maximum value of each ECDF is obtained. Lastly, Zheng et al. (2018) recommend using 0.3 as a threshold for the absolute beta coefficients to determine if a non-zero value corresponds to an edge in continuous optimization-based methods—we mark this threshold with a vertical dashed line.

All three simulation methods generate substantially different distributions of parametric models. The ZARX method consistently produces more edges with large absolute beta coefficients relative to the other two methods, while DaO produces edges with absolute beta coefficients closer to zero. Similarly, the ZARX method produces the highest absolute

5. This simulation was originally used by Zheng et al. (2018) but many continuous optimization-based CDA publications use this design as well.
6. Tetrad is a Java application for causal discovery. Simulations in Tetrad are flexible, however, this design was used in a recent publication whose authors included Joseph Ramsey—the lead developer and maintainer of Tetrad.

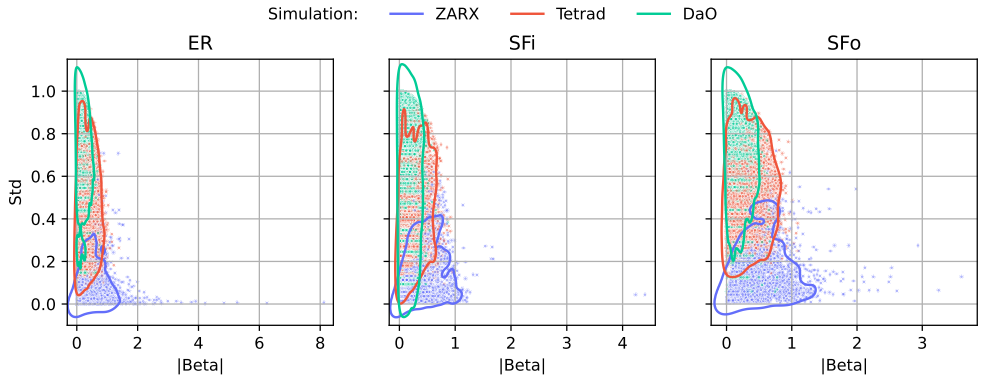

Figure 2: Properties of 10 models generated from ER/SFi/SFo-DAGs with $|V| = 100$ and $\alpha = 10$.

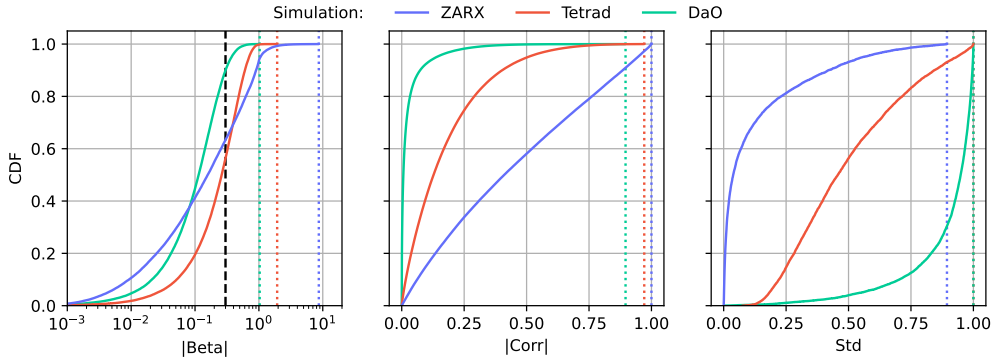

Figure 3: Properties of 100 models generated from ER-DAGs with $|V| = 100$ and $\alpha = 10$.

correlation values, while DaO produces the lowest absolute correlation values. Correspondingly, the standard deviations of the error terms in ZARX models are the smallest, while the standard deviations of error terms in DaO models are the largest.

The vertical dashed lines in these figures indicate that the standard threshold used by continuous optimization-based methods rules out over 50% of the true edges in the standardized versions of all three simulations. As such, all edges to the left of the vertical dashed lines can not be discovered by continuous optimization-based methods unless the user changes the threshold value—in Section 5 we lower the threshold to 0.1 (setting the threshold lower often results in cyclic models).

The ZARX method samples beta coefficients from a distribution bounded away from zero in an attempt to avoid (near-)violations of faithfulness and to generate models with a strong signal-to-noise ratio. Our experiments show that this is a misconception. After standardization, the beta coefficients produced by ZARX simulations are no longer bounded away from zero and there are many weak effects. In fact, the beta coefficients produced by Tetrad simulations tend to be further from zero than the beta coefficients produced by ZARX simulations. Moreover, ZARX simulations produce many additive error terms

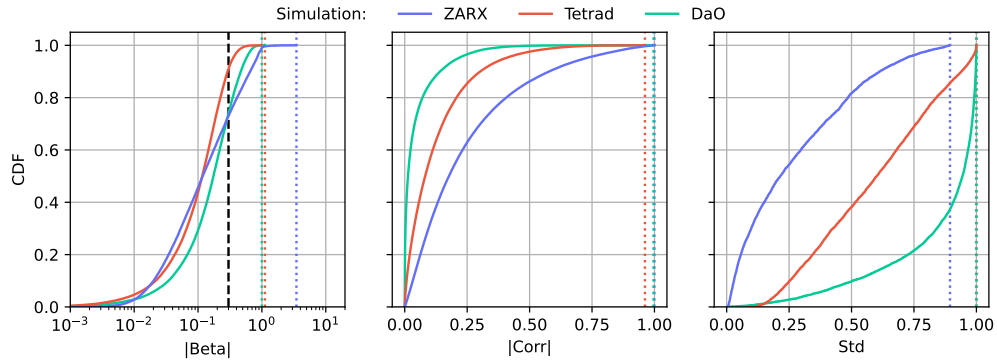

Figure 4: Properties of 100 models generated from SFi-DAGs with $|V| = 100$ and $\alpha = 10$.

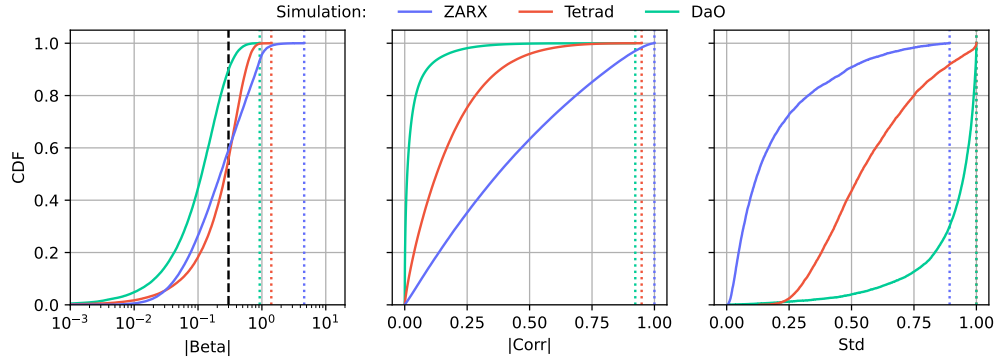

Figure 5: Properties of 100 models generated from SFo-DAGs with $|V| = 100$ and $\alpha = 10$.

with a standard deviation of approximately zero—this near determinism will induce (near-)violations of faithfulness.

Also puzzling is the presence of beta coefficients with magnitude approximately equal to 10 in the standardized ZARX models. This is only possible if two or more parents vertices are almost perfectly (anti-)correlated and together have a jointly canceling effect on a shared child vertex.

4.3 R^2-Sortability of Parametric Models

Unlike varsortability which can be prevented by standardization (Reisach et al., 2021), it is more difficult to prevent R^2-sortability (Reisach et al., 2023). In this section, we evaluate the R^2-sortability of the parametric models produced by the three simulation methods. We generated 100 models with 100 variables for each model structure: ER, SFi, and SFo; smaller models are considered in Appendix A.5. All models were generated with average degree $\alpha = 10$.

We quantified the R^2-sortability of each simulation method on each model structure type by calculating the rank correlation coefficient between each variable's R^2-sortability rank and its index from the variable order used to generate the DAG; see Section A.2.

Simulated R^2-sortable data will have correlations approaching plus or minus one, while, simulated non-R^2-sortable data will have correlations closer to zero.

	ZARX	Tetrad	DaO
ER	-0.873	-0.557	-0.386
SFi	-0.552	-0.336	-0.108
SFo	-0.497	-0.051	-0.084

Table 1: R^2 rank correlation on DAGs with $|V| = 100$ and $\alpha = 10$ over 100 repetitions

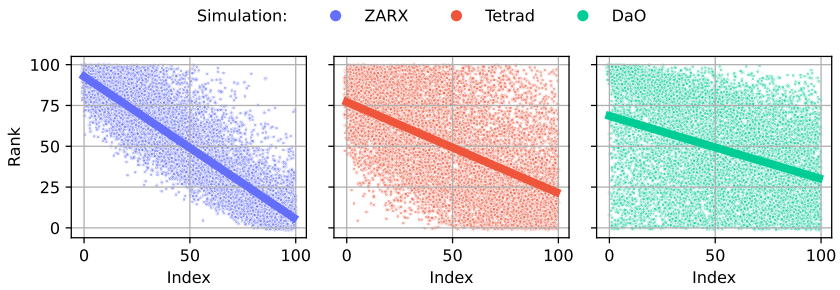

Figure 6: R^2-sortability of models generated from ER-DAGs with $|V| = 100$ and $\alpha = 10$ over 100 repetitions.

Table 1 tabulates the levels of R^2-sortability for the three simulation methods while varying model structure. The ZARX simulations have the highest R^2-sortability for all model structures, while R^2-sortability appears to be present in Tetrad simulations but at a weaker level than in ZARX simulations. Overall, the DaO method has lower R^2-sortability levels. Interestingly, we found that R^2-sortability varies substantially with model structure, as ER models typically had the highest R^2-sortability correlations, and SFo models typically had the lowest R^2-sortability correlations.

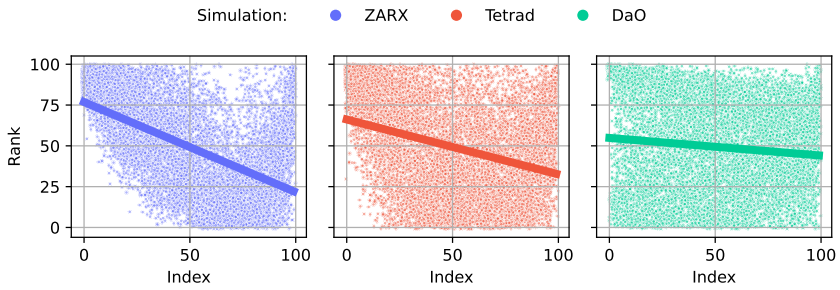

Figure 7: R^2-sortability of models generated from SFi-DAGs with $|V| = 100$ and $\alpha = 10$ over 100 repetitions.

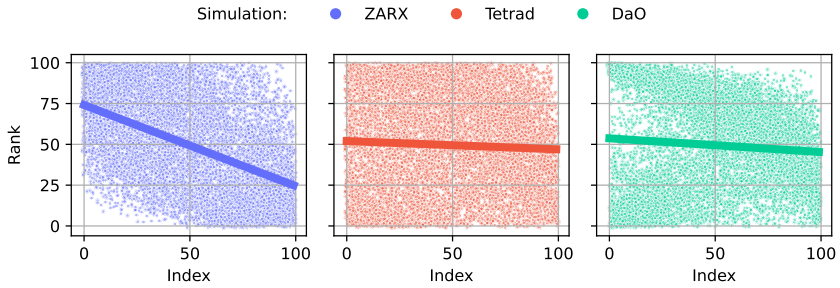

Figure 8: R^2-sortability of models generated from SFo-DAGs with $|V| = 100$ and $\alpha = 10$ over 100 repetitions.

4.4 Summary of Characterization Findings

Overall DaO simulations generate models with beta coefficients that tend to be smaller than the beta coefficeints generated by the ZARX or Tetrad simulations. The independent variances of DaO simulations also tend to be greater than those of the comparison methods. Major advantages of the DaO method are that, (i) unlike the comparison methods, DaO avoids creating problematic models that contain extreme betas with values near or above 1; (ii) DaO avoids creating variables that are nearly deterministic; (iii) R^2-sortability is low enough to be uninformative for DaO simulations.

5. Simulation Study

In this section, we report the results of a simulation study testing a variety of CDAs on data generated using the DaO method and two alternatives: the ZARX and Tetrad methods. The purpose of this study is to illustrate the differences between the DaO method and more traditional simulation designs like ZARX and Tetrad, and to highlight advantages of using the DaO method. The results of this study are summarized at the end of the section.

5.1 Simulation Details

Algorithm 4 details how the simulated datasets were generated from coefficient matrix B, additive error-variance matrix Ω, additive error ϵ, sample size n, and boolean flag δ. The flag δ is used to standardize variables immediately after they are sampled; see Appendix A.4.

In Algorithm 4: $0_{n\times p}$ denotes the $n \times p$ zero matrix while the subscripts of $B_{(p\times p)}$ and $\Omega_{(p\times p)}$ denote their dimensions. The second parameter of $\mathcal{E}$ gives the number of samples to draw.

We used the metrics tabulated in Table 2, which were originally proposed by Kummerfeld et al. (2023). The true DAG, rather than its Markov equivalence class, is used as the gold standard for these evaluation metrics because (i) algorithms such as Direct Linear Non-Gaussian Acyclic Mode (dLiNGAM) are not constrained to the Markov equivalence class and can in theory learn the complete DAG; (ii) from a practical perspective, restricting an algorithm to outputting an equivalence class rather than a complete DAG has the purpose of preventing false positive orientations and thereby increasing orientation precision; the

Algorithm 4: $\texttt{simulate}(B, \Omega, \mathcal{E}, n, \delta)$

Input: $\texttt{coef} : B_{(p\times p)}$ $\texttt{err-var} : \Omega_{(p\times p)}$ $\texttt{error} : \mathcal{E}$ $\texttt{samples} : n$ $\texttt{std-flag} : \delta$
Output: $\texttt{data} : X$

$X \leftarrow 0_{n\times p}$
for $i \leftarrow 1$ **to** p **do**
  $Z_i \sim \mathcal{E}(\Omega_{i,i}, n)$
  $X_{[n],i} \leftarrow X_{[n],[i]} \left(B_{i,[i]}\right)^\top + Z_i$
  if δ **then**
    $X_{[n],i} \leftarrow \left(X_{[n],i} - \texttt{mean}(X_{[n],i})\right) / \texttt{std}(X_{[n],i})$

advantages and purposes of doing such are obfuscated when the equivalence class itself is treated as the learning target.

All simulations were run with average degree $\alpha = 10$ over 20 repetitions at sample sizes: 200, 400, 600, 800, 1,000, 1,500, 2,000, 3,000, and 4,000. The number of variables were varied ($|V| = 20$ and $|V| = 100$). All algorithms were run on an Apple M1 Pro processor with 16G of RAM. Algorithms were dropped from the analysis in cases where they took longer than 20 minutes per run.

5.2 Tested Algorithms and Algorithm Settings

In the simulations, the causal discovery algorithms use the following tests and scores.

- Fisher Z test (Anderson, 1984; Spirtes et al., 1993) is a test of zero-partial correlation using Fisher's z-transform; we set the alpha parameter to 0.01.
- Bayesian Information Criterion (BIC) score (Schwarz, 1978; Haughton, 1988) is a penalized likelihood score that approximates the marginal likelihood for curved exponential families; we set the penalty discount parameter to 2.
- Bayesian Gaussian equivalent (BGe) score (Geiger and Heckerman, 2002; Kuipers et al., 2014) is a Bayesian score for a multivariate Gaussian distribution using conjugate priors; we use default parameters.

The simulations use multiple algorithms from multiple different software packages, outlined below. Notably, only algorithms σ-Sort, R^2-Sort, and dLiNGAM output DAGs; all other algorithms output (or have been modified to output) CPDAGs.[7]

- `https://github.com/kevinsbello/dagma`
  - Directed Acyclic Graphs via M-matrices for Acyclicity (DAGMA) (Zheng et al., 2018; Bello et al., 2022) is a continuous optimization-based algorithm using an L1 penalty and the Adam optimizer; we set the parameter thresholded = 0.1 (or smallest threshold > 0.1 that returns a DAG), use defaults for all other parameters, and convert the output to a CPDAG.

7. A CPDAG is a graphical summarization of a set of DAGs that imply the same conditional independence relationships. Many CDAs output a CPDAG rather than a DAG.

- `https://github.com/cmu-phil/tetrad` (Ramsey et al., 2018)
  - Best Order Score Search (BOSS) (Lam et al., 2022; Andrews et al., 2023) searches over permutations which are projected to DAGs using the BIC score; we set the parameter restarts = 10 and use defaults for all other parameters.
  - Fast Greedy Equivalence Search (fGES) (Chickering, 2002; Ramsey et al., 2017) is an optimized version of the greedy equivalent search which searches over equivalence classes of DAGs by iteratively adding then removing edges until the model cannot be improved according to the BIC score; we use default parameters.
  - Peter Clark (PC) (Spirtes et al., 1993; Colombo and Maathuis, 2014) explicitly tests conditional independence relationships to rule out incompatible hypothesis graphs using the Fisher Z test; we set the parameter stable = *true* and use defaults for all other parameters.
- `https://github.com/CausalDisco/CausalDisco`
  - σ-Sortability (σ-Sort) (Reisach et al., 2021) orders variables by iteratively minimizing a variance criterion; we use default parameters.
  - R^2-Sortability (R^2-Sort) (Reisach et al., 2023) orders variables by iteratively minimizing a partial correlation criterion; we use default parameters.
- `https://github.com/cdt15/lingam` (Ikeuchi et al., 2023)
  - Direct Linear Non-Gaussian Acyclic Model (dLiNGAM) (Shimizu et al., 2006, 2011) orders the variables by iteratively minimizing a non-Gaussian criterion; we use default parameters.
- `https://cran.r-project.org/web/packages/BiDAG/index.html` (Suter et al., 2023)
  - Iterative Markov Chain Monty Carlo (itMCMC) (Kuipers et al., 2022) uses MCMC to sample orders that are partially constrained by an adjacency matrix learned by PC using the Fisher Z test and BGe score; we set the parameter hardlimit = 20 and use defaults for all other parameters.
- `https://cran.r-project.org/web/packages/pchc/index.html` (Tsagris, 2023)
  - Max-Min Hill-Climbing* (MMHC*) is a continuous variant of the original MMHC algorithm (Tsamardinos et al., 2006; Tsagris, 2021). It initially grows Markov blankets by iteratively adding variables that maximize the minimum BIC score over subsets of the current Markov blankets. Explicit tests of conditional independence are then used to rule out incompatible hypothetical graphs using the Fisher Z test; we set the parameter restarts = 10 and use defaults for all other parameters.

$$\text{Precision} = \frac{\texttt{tp}}{\texttt{tp} + \texttt{fp}}$$

$$\text{Recall} = \frac{\texttt{tp}}{\texttt{tp} + \texttt{fn}}$$

$$\text{F1 Score} = \frac{\texttt{tp} + \texttt{tp}}{\texttt{tp} + \texttt{fp} + \texttt{tp} + \texttt{fn}}$$

True	Estimated	Adjacency	Orientation
$i \leftarrow j$	$i \leftarrow j$	tp	tp, tn
	$i \rightarrow j$	tp	fp, fn
	$i - j$	tp	fn
	$i \dots j$	fn	fn
$i \dots j$	$i \leftarrow j$	fp	fp
	$i \rightarrow j$	fp	fp
	$i - j$	fp	
	$i \dots j$	tn	

Table 2: Metrics

5.3 DaO Simulations

In the DaO simulations, graphs were generated using the ER-DAG; see Appendix A.2, moreover scale-free simulations are considered in Appendix A.6. The model parameters were passed to Algorithm 4 to generate two versions of the simulated data: Gaussian error and non-Gaussian exponential error. The Gaussian version of the simulated data was used to evaluate all CDAs with one exception. The non-Gaussian version of the simulated data was used to evaluate dLiNGAM due to its dependence on a non-Gaussian signal. Accordingly, dLiNGAM had access to additional information that the other algorithms did not, which is reflected in its superior performance. The σ-sortability algorithm was not evaluated on DaO simulations for a similar reason—there is no signal to sort the variables since all variances are one.

5.4 Alternative Simulations

In the alternative simulations, all graphs were generated using the ER-DAG method; see Appendix A.2. The model parameters were passed to Algorithm 4 to generate two versions of the simulated data: Gaussian error and non-Gaussian exponential error. The Gaussian version of the simulated data was used to evaluate all CDAs with one exception. The non-Gaussian version of the simulated data was used to evaluate dLiNGAM due to its dependence on a non-Gaussian signal. As in the DaO simulations, dLiNGAM had access to additional information that the other algorithms did not, which is reflected in its superior performance. The simulated data were not standardized before running any of the CDAs with one exception. DAGMA was run twice: once on the unstandardized data (DAGMA) and once on the standardized data (sDAGMA).

As noted in Section 4, ZARX simulations create near deterministic relationships between the variables. We see that methods such as PC and GES perform poorly in the presence of this near determinism. We conjecture this is because the near deterministic relationships result in (near-)violations of faithfulness and local optima. Moreover, the difference in performance between DAGMA and sDAGMA highlights the complains of several papers targeting the simulations commonly used to evaluate continuous optimization-based methods.

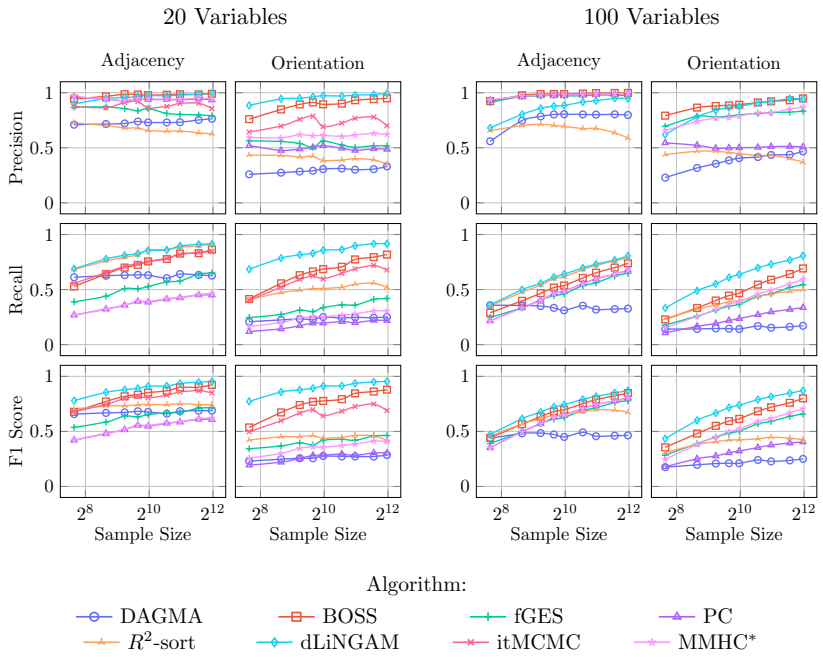

Figure 9: Mean performance of CDAs on data generated from ER-DAGs with $\alpha = 10$ using the DaO method for DAG Models over 20 repetitions.

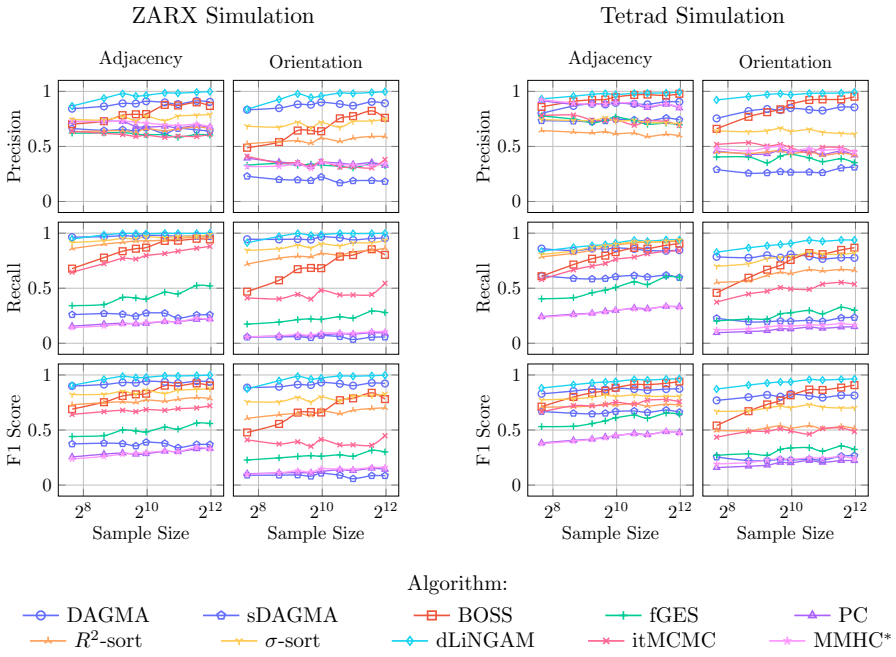

Figure 10: Mean performance of CDAs on data generated from ER-DAGs with 20 variables and $\alpha = 10$ over 20 repetitions.

5.5 Summary of Simulation Findings

DaO simulations demonstrated a greater ability to detect unusual CDA behavior. For example, while *a priori* σ-sort cannot perform well on DaO, σ-sort achieved reasonable performance on ZARX and Tetrad, and outperformed some more legitimate CDAs.

As another example, in DaO simulations most CDAs show increasing performance as sample size increases, however DAGMA and R^2-sort seem less responsive to DaO sample sizes. In contrast, ZARX and Tetrad simulations show nearly all of the CDAs being relatively unresponsive to sample size. This is likely due to the differences in the distribution of betas and independent variances reported earlier in Section 4.2. ZARX and Tetrad, by omitting all weak betas, mean that the advantages of larger sample sizes are significantly attenuated. By comparison, increasing sample sizes provides increasing power for detecting the small betas in DaO simulations.

Despite the DaO method sampling betas that are arbitrarily close to zero, higher performing CDAs are able to have good recall performance. Most of the tested CDAs show a trend that is approaching an adjacency recall of one as sample size increases. This is evidence that bounding betas away from zero is not necessary for evaluating CDAs, and in fact shows that not including such bounds has the advantage of being able to evaluate which CDAs are better able to detect weaker betas.

The DaO method gives performance evaluations of the various CDA methods are are quite different from the ZARX and Tetrad methods. CDA methods that had superior performance in DaO, namely dLiNGAM and BOSS, also had good performance in ZARX and Tetrad. The alternative is not true, however: some methods, like DAGMA and σ-sort, had superior performance on ZARX or Tetrad simulations, but had poor performance on DaO (or in the case of σ-sort, *a priori* cannot have good performance on DaO and as such was not even tested). This reflects the fact that DaO simulations cover the complete space of distributions, while ZARX and Tetrad simulations do not.

6. Discussion

We present the DaO method for uniformly sampling matrices from the space of correlation matrices satisfying the Markov property with respect to a DAG. DaO simulations should be adopted as a universal benchmark for evaluating algorithms that learn DAGs or fit SEMs to DAGs. Python and R implementations of the DaO method are available on GitHub: `https://github.com/bja43/DaO_simulation` and the Python implementation is also available on PyPI: `https://pypi.org/project/daosim`.

The DaO method solves multiple problems that have been recently highlighted by several papers criticizing common data simulation practices. These papers identified properties they claimed are unrealistic and could be unfairly exploited by some learning algorithms. Uniformly sampling matrices, as per the DaO method, ensures that any signatures in the data are a product only of sampling data from DAGs, and are not a product of some human bias for selecting simulation designs favorable to one type of learning algorithm or another.

DaO allows any distribution with finite variance to be used for the additive error terms. This is not the same as allowing for the generation of data following any chosen multivariate distribution. For example, in the case of a non-Gaussian distribution like Student's T, the multivariate distribution produced by DaO when using Student's T additive errors will not

be a multivariate Student's T distribution. This is because many non-Gaussian families are not closed under summation. However, in some cases it is possible to sample data from a multivariate non-Gaussian distribution with a specific correlation matrix. For example, it is possible to sample directly from a multivariate Student's T distribution with specified correlations. If one further wants this correlation matrix to be consistent with a specific DAG, then DaO could be used to generate that correlation matrix. Note that in this situation, the pairwise relationships may no longer be linear.

6.1 Limitations

The DaO simulation method has several limitations. First, the DaO method and DAG sampling methods in general are naturally limited to acyclic models, however, many causal processes in the natural world are cyclic. Second, we do not present any methodology for simulation or evaluation in the presence of unmeasured confounding. Third, our simulations only produce cross-sectional data and cannot be directly used for evaluating methods intended for time-series contexts. Fourth, while sampling uniformly from the space of all correlation matrices has many advantages, we may want to know how an algorithm is likely to perform on a specific dataset from a specific causal system. In such cases, a simulation that is more tailored to that specific dataset and causal system may be more informative than a DaO simulation. Fifth, while extension to nonlinear models would be valuable, it is not immediately clear how DaO can be extended to a nonlinear setting, since it is reliant on the representative power of a correlation matrix.

6.2 Future Directions

Future Directions for further developing DaO include the following:

1. Extend the DaO method to latent variables.
2. Extend the DaO method to time-series data.
3. Design and execute a large simulation study using the DaO method.
4. Investigate methods for restricting DaO to a subspace of correlation matrices.
5. Contribute the DaO method to other tools that have been made for the comparison and validation of CDAs such as Benchpress (Rios et al., 2021).

6.3 Insights into the Conflicting Results of Previous Simulations

Finally, in addition to developing and characterizing the DaO method, this paper also sheds some light on the ongoing debate surrounding continuous optimization-based methods.

First, it highlights that continuous optimization-based methods need to pick a lower threshold in order to recover edges from standardized models. This is illustrated by Figures 3, 4, and 5. In addition to lacking signals like varsortability, this may be part of the reason that continuous optimization-based methods did not perform well on the DaO simulations reported in Figures 9, 19, and 20.

Second, the need to bound beta coefficients away from zero is a misconception. Instead, this can result in near determinism which creates (near-)violations of faithfulness. Figures 3,

4, and 5 highlight that the ZARX simulations, used by continuous optimization publications, produce a substantial number of these nearly deterministic relationships. Moreover, non-continuous optimization-based CDA are known to perform poorly in the presence of deterministic relationships (Glymour, 2007)

These points help explain the substantial conflict between the simulation results reported in papers that use the ZARX method versus other methods. The performance of DAGMA and sDAGMA in Figure 10 illustrates these points.

All of the above issues can be resolved if future researchers replace the use of ad hoc simulation designs with simulation designs that are (1) well-characterized; (2) proven to not produce ad hoc simulation artifacts; (3) cover the complete space of models. At this time, the only method that satisfies all of these criteria is the DaO method.

Acknowledgments

We thank Peter Spirtes, Joe Ramsey, and Constantin Aliferis for insightful comments and discussion. BA was supported by the NIH under the Comorbidity: Substance Use Disorders and Other Psychiatric Conditions Training Program T32DA037183 and the Midwest Regional Postdoctoral Training in Eating Disorders Research T32MH082761. EK was supported by the NIH under awards P50MH119569 and UL1TR002494.

References

T.W. Anderson. *An Introduction to Multivariate Statistical Analysis.* An Introduction to Multivariate Statistical Analysis. Wiley, 1984.

Bryan Andrews, Joseph D. Ramsey, Rubén Sánchez-Romero, Jazmin Camchong, and Erich Kummerfeld. Fast scalable and accurate discovery of DAGs using the best order score search and grow shrink trees. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, volume 36, pages 63945–63956, 2023.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

Kevin Bello, Bryon Aragam, and Pradeep K. Ravikumar. DAGMA: Learning DAGs via M-matrices and a log-determinant acyclicity characterization. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, volume 35, pages 8226–8239, 2022.

Kenneth A. Bollen. *Structural equations with latent variables*, volume 210. John Wiley & Sons, 1989.

David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

Diego Colombo and Marloes H. Maathuis. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research*, 15:3741–3782, 2014.

Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 11 2005.

Brian A. Davey and Hilary A. Priestley. *Introduction to lattices and order*. Cambridge University Press, 2002.

A.P. Dawid. Conditional independence in statistical theory. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41:1–15, 1979.

Mathias Drton. Algebraic problems in structural equation modeling. In *The 50th anniversary of Gröbner bases*, volume 77 of *Advanced Studies in Pure Mathematics*, pages 35–87. Mathematical Society of Japan, 2018.

Frederick Eberhardt. Introduction to the foundations of causal discovery. *International Journal of Data Science and Analytics*, 3:81–91, 2017.

Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

Kai-Tang Fang, Samuel Kotz, and Kai W. Ng. *Symmetric multivariate and related distributions*. Chapman and Hall, 1990.

Dan Geiger and David Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *The Annals of Statistics*, 30: 1412–1440, 2002.

Soumyadip Ghosh and Shane G. Henderson. Behavior of the NORTA method for correlated random vector generation as the dimension increases. *ACM Transactions on Modeling and Computer Simulation*, 13:276–294, 2003.

Soumyadip Ghosh and Shane G. Henderson. Corrigendum: Behavior of the NORTA method for correlated random vector generation as the dimension increases. *ACM Transactions on Modeling and Computer Simulation*, 19:1–3, 2009.

Clark Glymour. Learning the Structure of Deterministic Systems. In *Causal Learning: Psychology, Philosophy, and Computation*, pages 231–240. Oxford University Press, 2007.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10:524, 2019.

Dominique M.A. Haughton. On the choice of a model to fit data from an exponential family. *The Annals of Statistics*, pages 342–355, 1988.

Takashi Ikeuchi, Mayumi Ide, Yan Zeng, Takashi Nicholas Maeda, and Shohei Shimizu. Python package for causal discovery based on LiNGAM. *Journal of Machine Learning Research*, 24:1–8, 2023.

Marcus Kaiser and Maksim Sipos. Unsuitability of NOTEARS for causal graph discovery when dealing with dimensional quantities. *Neural Processing Letters*, 54:1587–1595, 2022.

Markus Kalisch and Peter Bühlman. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8, 2007.

Mohammad Khalafi and Majid Azimmohseni. Multivariate Pearson type II distribution: Statistical and mathematical features. *Probability and Mathematical Statistics*, 34:119–126, 2014.

Jack Kuipers and Giusi Moffa. Uniform random generation of large acyclic digraphs. *Statistics and Computing*, 25:227–242, 2015.

Jack Kuipers, Giusi Moffa, and David Heckerman. Addendum on the scoring of Gaussian directed acyclic graphical models. *The Annals of Statistics*, pages 1689–1691, 2014.

Jack Kuipers, Polina Suter, and Giusi Moffa. Efficient sampling and structure learning of Bayesian networks. *Journal of Computational and Graphical Statistics*, 31:639–650, 2022.

Erich Kummerfeld, Leland Williams, and Sisi Ma. Power analysis for causal discovery. *International Journal of Data Science and Analytics*, pages 1–16, 2023.

Wai-Yin Lam, Bryan Andrews, and Joseph D. Ramsey. Greedy relaxations of the sparsest permutation algorithm. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 1052–1062. PMLR, 2022.

Steffen L. Lauritzen, A. Philip Dawid, Birgitte N. Larsen, and H.-G. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

Andrew R. Lawrence, Marcus Kaiser, Rui Sampaio, and Maksim Sipos. Data generating process to evaluate causal discovery techniques for time series data, 2021.

Daniel Lewandowski, Dorota Kurowicka, and Harry Joe. Generating random correlation matrices based on vines and extended Onion method. *Journal of Multivariate Analysis*, 100:1989–2001, 2009.

Daniel Malinsky and David Danks. Causal discovery algorithms: A practical guide. *Philosophy Compass*, 13:e12470, 2018.

Christopher Meek. Strong completeness and faithfulness in Bayesian networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 411–418. PMLR, 1995.

Guy Melançon, Isabelle Dutour, and Mireille Bousquet-Mélou. Random generation of directed acyclic graphs. *Electronic Notes in Discrete Mathematics*, 10:202–207, 2001.

Francesco Montagna, Atalanti Mastakouri, Elias Eulig, Nicoletta Noceti, Lorenzo Rosasco, Dominik Janzing, Bryon Aragam, and Francesco Locatello. Assumption violations in causal discovery and the robustness of score matching. *Advances in Neural Information Processing Systems*, 36:47339–47378, 2023.

Mervin E. Muller. Some continuous Monte Carlo methods for the Dirichlet problem. *The Annals of Mathematical Statistics*, 27:569–589, 1956.

Ignavier Ng, Biwei Huang, and Kun Zhang. Structure learning with continuous optimization: A sober look and beyond. In *Causal Learning and Reasoning*, pages 71–105. PMLR, 2024.

Diane Valérie Ouellette. Schur complements and statistics. *Linear Algebra and its Applications*, 36:187–295, 1981.

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

Judea Pearl. *Book of Why*. Basic Books, 2020.

Jonas Peters and Peter Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, 101:219–228, 2014.

Derek de Solla Price. A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27:292–306, 1976.

Joseph D. Ramsey and Bryan Andrews. A comparison of public causal search packages on linear, Gaussian data with no latent variables, 2017.

Joseph D. Ramsey, Madelyn Glymour, Rubén Sánchez-Romero, and Clark Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International Journal of Data Science and Analytics*, 3:121–129, 2017.

Joseph D. Ramsey, Kun Zhang, Madelyn Glymour, Ruben Sanchez Romero, Biwei Huang, Imme Ebert-Uphoff, Savini Samarasinghe, Elizabeth A. Barnes, and Clark Glymour. TETRAD—a toolbox for causal discovery. In *8th International Workshop on Climate Informatics*, 2018.

Joseph D. Ramsey, Daniel Malinsky, and Kevin V. Bui. algcomparison: Comparing the performance of graphical structure learning algorithms with TETRAD. *Journal of Machine Learning Research*, 21:1–6, 2020.

Alexander Reisach, Christof Seiler, and Sebastian Weichwald. Beware of the simulated DAG! causal discovery benchmarks may be easy to game. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, volume 34, pages 27772–27784, 2021.

Alexander Reisach, Myriam Tami, Christof Seiler, Antoine Chambaz, and Sebastian Weichwald. A scale-invariant sorting criterion to find a causal order in additive noise models. In *Advances in Neural Information Processing Systems*, volume 36, pages 785–807, 2023.

Felix L. Rios, Giusi Moffa, and Jack Kuipers. Benchpress: a scalable and versatile workflow for benchmarking structure learning algorithms for graphical models, 2021.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, pages 461–464, 1978.

Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7, 2006.

Shohei Shimizu, Takanori Inazumi, Yasuhiro Sogawa, Aapo Hyvarinen, Yoshinobu Kawahara, Takashi Washio, Patrik O. Hoyer, Kenneth Bollen, and Patrik Hoyer. DirectLiNGAM: A direct method for learning a linear non-Gaussian structural equation model. *Journal of Machine Learning Research*, 12:1225–1248, 2011.

Gyorgy J. Simon and Constantin Aliferis. *Artificial intelligence and machine learning in health care and medical sciences: best practices and pitfalls*. Springer Nature, 2024.

Peter Spirtes. Introduction to causal inference. *Journal of Machine Learning Research*, 11: 1643–1662, 2010.

Peter Spirtes and Jiji Zhang. A uniformly consistent estimator of causal effects under the k-triangle-faithfulness assumption. *Statistical Science*, pages 662–678, 2014.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Lecture Notes in Statistics. Springer, 1993.

Polina Suter, Jack Kuipers, Giusi Moffa, and Niko Beerenwinkel. Bayesian structure learning and sampling of Bayesian networks with the R package BiDAG. *Journal of Statistical Software*, 105:1–31, 2023.

Michail Tsagris. A new scalable Bayesian network learning algorithm with applications to economics. *Computational Economics*, 57:341–367, 2021.

Michail Tsagris. *PCHC: Bayesian Network Learning with the PCHC and Related Algorithms*, 2023. URL `https://CRAN.R-project.org/package=pchc`. R package version 1.2.

Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 2006.

Ruibo Tu, Kun Zhang, Bo Bertilson, Hedvig Kjellstrom, and Cheng Zhang. Neuropathic pain diagnosis simulator for causal discovery algorithm evaluation. *Advances in Neural Information Processing Systems*, 32, 2019.

Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, 41:436–463, 2013.

Thomas Verma and Judea Pearl. Causal networks: Semantics and expressiveness. In *Uncertainty in Artificial Intelligence*, volume 9 of *Machine Intelligence and Pattern Recognition*, pages 69–76. North-Holland, 1990.

Shuyan Wang and Peter Spirtes. A uniformly consistent estimator of non-Gaussian causal effects under the k-triangle-faithfulness assumption. In Bernhard Schölkopf, Caroline Uhler, and Kun Zhang, editors, *Proceedings of the Conference on Causal Learning and*

Reasoning, volume 177 of *Proceedings of Machine Learning Research*, pages 861–876. PMLR, 2022.

Xun Zheng, Bryon Aragam, Pradeep K. Ravikumar, and Eric P. Xing. DAGs with no tears: Continuous optimization for structure learning. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, volume 31, pages 9492–9503, 2018.

Appendix A. Appendix

A.1 Method Label for DaO

Following Simon and Aliferis (2024) we provide a DaO method label:

Method Label for DaO	
Main use	• Avoids common artifacts and biases in simulation studies with directed acyclic graphs (DAGs), by randomly sampling standardized structural equation model (SEM) parameters with implied correlation matrices uniformly distributed over all correlation matrices Markov to the DAG
Context of use	• Simulation studies for evaluating the performance of DAG-learning and SEM-learning algorithms, such as causal discovery algorithms
Secondary use	• Sampling constrained correlation matrices for any other purpose, such as Markov Chain Monte Carlo (MCMC) procedures or as a prior distribution for Bayesian inference
Pitfalls	• Many real-world correlation matrices are not uniformly distributed • DaO sampling produces many small coefficients, which in scientific settings may be considered less important cause-effect pairs
Principle of operation	• Parameterizes a correlation matrix (and correspondingly, the coefficient and error-variance parameters of a SEM for a given input DAG) layer by layer with respect to a topological order. Each layer contains the correlations of a newly added variable with the preceding variables, and those values are drawn based on the values of the preceding layers (i.e. preceding variables' correlations) • Based on theory for elliptical distributions and properties of the multivariate Pearson type II distribution • DaO is a modification of the Onion method by Ghosh and Henderson, by generalizing it to all DAGs rather than only fully-connected DAGs
Theoretical properties and empirical evidence	• Proven to draw uniformly from the space of correlation matrices that satisfy the Markov Condition for the input DAG • Models are guaranteed to be standardized, thereby preventing simulation artifacts such as "varsortability" • Better at avoiding near deterministic relationships • Has support over all valid correlation matrices • Can not be used to cherry-pick algorithm performance, because it has no simulation parameters or other points of control • Because it samples uniformly, the DaO method represents a generic context with a complete lack of background knowledge (i.e., an uninformative prior over the model space) • When comparing DAG-learning methods, the DaO method enables identification of the method which has best average performance over all possible inputs
Best practices	• All causal discovery and DAG-learning methods meant for linear models should have their performance benchmarked on DaO simulations

A.2 Sampling Random DAGs

We consider two approaches for generating graphs: Erdős Rényi (ER) where edges occur with equal probability (Erdős and Rényi, 1959) and Scale-Free (SF) where edges occur such that the distribution of vertex degrees follows a power-law (Barabási and Albert, 1999). These methods produce undirected graphs rather than DAGs, but are frequently adapted to generate DAGs by imposing a topological order on the vertices—we also take this approach. Algorithm 5 contains pseudocode for sampling ER-DAGs using the total order established by Convention 1.

Algorithm 5: `ER-DAG`(V, α)

Input: `vertices` : V `avg-deg` : α
Output: `DAG` : $\mathcal{G} = (V, E)$

$E \leftarrow \varnothing$
repeat
  $(i, j) \sim f : V \times V \to \mathbb{R}$ s.t. $f \propto 1_{j<i}$
  $E \leftarrow E \cup \{(i, j)\}$
until $|E| = \frac{\alpha}{2} |V|$

In Algorithm 5, $1_{j<i}$ denotes the indicator function for $j < i$. Algorithm 5 (`ER-DAG`) takes vertex set V and average degree α as input to construct a DAG. The total order over V established by Convention 1 is imposed on the vertices to direct the edges. Edges consistent with the imposed order are sampled uniformly at random until an average degree of α is achieved. Since the operation of some causal discovery algorithms (CDAs) can depend on the order of the variables, the vertex order of a DAG can (and perhaps should) be randomized after sampling.

Notably, the ER-DAG method is biased and does not produce DAGs uniformly at random. This is a limitation of the current work. For reader interested in unbiased DAG sampling procedures, consider the works of Melançon et al. (2001); Kuipers and Moffa (2015).

SF-DAGs are commonly generated following Price (1976). Many studies use the `igraph` C library (Csardi and Nepusz, 2005) which by default generates SF-DAGs with scale-free in-degree and constant vertex out-degree (equal to $\frac{\alpha}{2}$). This approach can generate SF-DAGs with scale-free out-degree and constant vertex in-degree by reversing the direction of the edges in the graph. However, it is not straightforward to add variation to the non-scale-free vertex degree or to generate DAGs that have scale-free in-degree and out-degree at the same time.

We present methods for modifying an existing DAG in a way that retains some properties, such as average degree and number of vertices, while changing other properties, such as having a power-law distribution for the vertex in-degree or out-degree. These algorithms were original introduced in Andrews et al. (2023).

Algorithm 6: `SFi-DAG`$(\mathcal{H})$

Input: `DAG` : $\mathcal{H} = (V, \cdot)$ s.t. $|V| = p$
Output: `DAG` : $\mathcal{G} = (V, E)$

$E \leftarrow \varnothing$
for $i \leftarrow p$ **to** 1 **do**
  repeat
    $j \sim f : V \to \mathbb{R}$ s.t. $f \propto 1_{i<j} + |\mathrm{pa}_{\mathcal{G}}(j)|$
    $E \leftarrow E \cup \{(j, i)\}$
  until $|\mathrm{ch}_{\mathcal{G}}(i)| = |\mathrm{ch}_{\mathcal{H}}(i)|$

Algorithm 7: `SFo-DAG`$(\mathcal{H})$

Input: `DAG` : $\mathcal{H} = (V, \cdot)$ s.t. $|V| = p$
Output: `DAG` : $\mathcal{G} = (V, E)$

$E \leftarrow \varnothing$
for $i \leftarrow 1$ **to** p **do**
  repeat
    $j \sim f : V \to \mathbb{R}$ s.t. $f \propto 1_{j<i} + |\mathrm{ch}_{\mathcal{G}}(j)|$
    $E \leftarrow E \cup \{(i, j)\}$
  until $|\mathrm{pa}_{\mathcal{G}}(i)| = |\mathrm{pa}_{\mathcal{H}}(i)|$

In Algorithms 6, and 7: $1_{i<j}$ and $1_{j<i}$ denote the indicator function for $i < j$ and $j < i$ respectively. Algorithm 6 (`SFi-DAG`) resamples the parents of each vertex via preferential attachment in reverse topological order resulting in a scale-free in-degree distribution while maintaining the prior out-degree distribution. Conversely, Algorithm 7 (`SFo-DAG`) resamples the children of each vertex via preferential attachment in topological order resulting in a scale-free out-degree distribution while maintaining the prior in-degree distribution.

These algorithms allow one to provide a DAG that has some desirable properties, such as number of edges or in/out-degree distribution, and modify it so that either the in-degree or the out-degree is scale-free. Additionally, both procedures may be applied to produce a graph with both scale-free in-degree and out-degree.[8] Figures 11 and 12 illustrate in/out-degree distributions for DAGs generated using the SFi-DAG method and the SFo-DAG method compared the ER-DAG method.

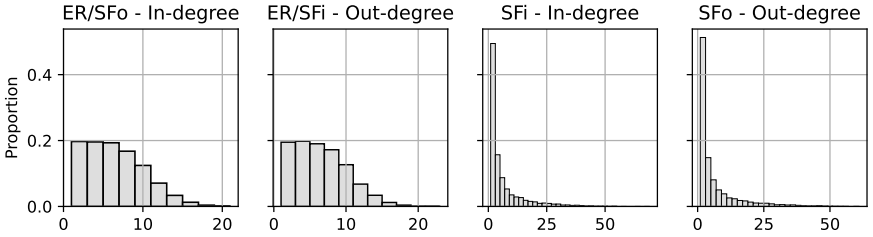

Figure 11: Vertex in/out-degree distributions for 100 DAGs with $|V| = 100$ and $\alpha = 10$.

Figure 12: Edge matrices for ER/SFi/SFo-DAGs with $|V| = 100$ and $\alpha = \frac{99}{2}$ (density $\frac{1}{2}$).

8. Applying `SFi-DAG` before `SFo-DAG` versus `SFo-DAG` before `SFi-DAG` results in different in/out-degree distributions. Their application may be iterated in an alternating manner until convergence if identical in/out-degree distributions are desired.

A.3 Additional Algorithms for Reference

We provide descriptions and pseudocode for algorithms used in Section 4.2 for readers who are unfamiliar with them. Algorithm 8 (`Cov-to-Corr`) converts a covariance matrix to a correlation matrix and Algorithm 9 (`Cov-to-DAG`) takes a DAG and covariance/correlation matrix as input and returns the corresponding B and Ω parameters for the model—the parameters are standardized if the input is a DAG and a correlation matrix.

Algorithm 8: `Cov-to-Corr`(Σ)

Input: `cov` : $\Sigma_{(p\times p)}$
Output: `corr` : R

$D \leftarrow 0_{p\times p}$
for $i \leftarrow 1$ **to** p **do**
  $D_{i,i} \leftarrow (\Sigma_{i,i})^{-\frac{1}{2}}$
$R \leftarrow D\,\Sigma\,D$

Algorithm 9: `Cov-to-DAG`$(\mathcal{G}, \Sigma)$

Input: `DAG` : $\mathcal{G}$ `cov/corr` : $\Sigma_{(p\times p)}$
Output: `coef` : B `err-var` : Ω

$B \leftarrow 0_{p\times p}$; $\Omega \leftarrow 0_{p\times p}$
for $i \leftarrow 1$ **to** p **do**
  $J \leftarrow \text{pa}_{\mathcal{G}}(i)$
  $B_{i,J} \leftarrow \Sigma_{i,J}\,(\Sigma_{J,J})^{-1}$
  $\Omega_{i,i} \leftarrow \Sigma_{i,i} - B_{i,J}\,\Sigma_{J,i}$

In Algorithms 8 and 9: $0_{p\times p}$ denotes the $p \times p$ zero matrix while the subscript of $\Sigma_{(p\times p)}$ denote its dimensions. We first evaluate the distributions of edge coefficients, implied absolute correlations, and the standard deviations of the independent error terms for each of these simulation methods. We then evaluate the R^2-sortability of these methods.

A.4 Additional Comparisons of Simulation Designs

We provide an extension to the analysis conducted in Section 4.2 by analyzing data generated from DAGs constructed as described in Appendix A.2 and Algorithm 4 using the alternative simulation method that standardizes the each variable immediately after sampling it (setting the flag parameter to *true*).

Figures 13, 14, and 15 depict empirical cumulative distribution functions (ECDFs) of the absolute beta coefficients, absolute correlations, and standard deviations of 100 models generated from DAGs with 100 variables and average degree $\alpha = 10$. The values where the corresponding implied correlation is zero or one are omitted and vertical dotted lines indicate where the maximum value of each ECDF is obtained. Lastly, Zheng et al. (2018) recommend using 0.3 as a threshold for the absolute beta coefficients to determine if a non-zero value corresponds to an edge in continuous optimization-based methods—we mark this threshold with a vertical dashed line.

A.5 Additional R^2-Sortability Analyses

We provide an extension to the analysis of R^2-sortability conducted in Section 4.3 by analyzing data generated from smaller DAGs. We evaluate the R^2-sortability of the parametric models produced by the three simulation methods. We generated 100 models with 20 variables for each model structure: ER, SFi, and SFo. All models were generated with average degree $\alpha = 10$.

We quantified the R^2-sortability of each simulation method on each model structure type by calculating the rank correlation coefficient between each variable's R^2-sortability rank and its index from the variable order used to generate the DAG; see Section A.2.

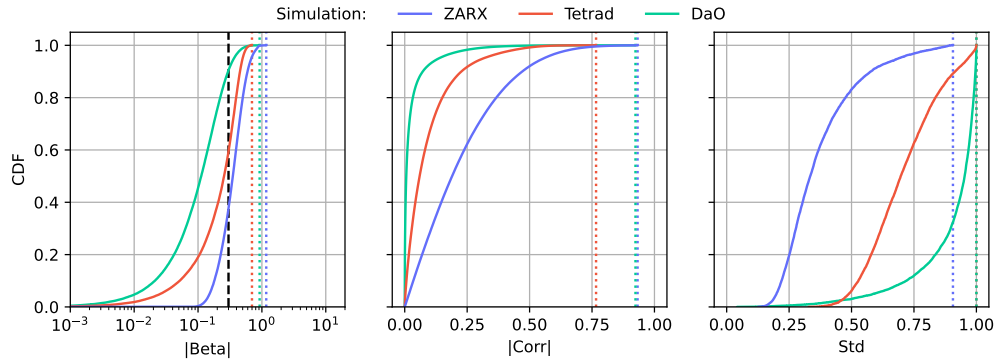

Figure 13: Properties of 100 models generated from ER-DAGs with $|V| = 100$ and $\alpha = 10$.

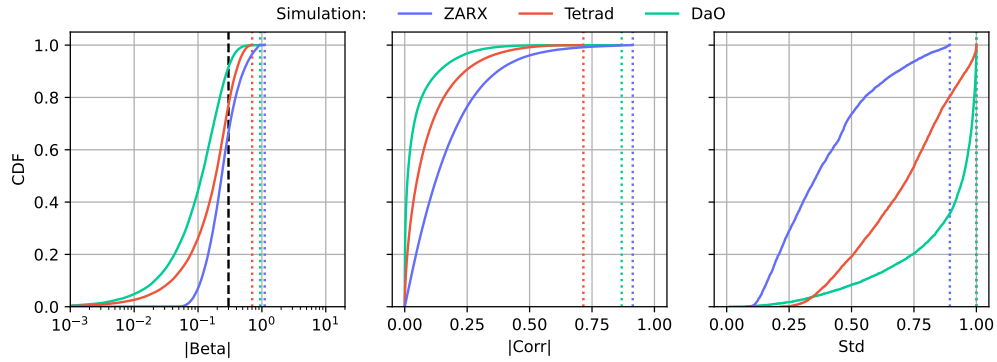

Figure 14: Properties of 100 models generated from SFi-DAGs with $|V| = 100$ and $\alpha = 10$.

Simulated R^2-sortable data will have correlations approaching plus or minus one, while, simulated non-R^2-sortable data will have correlations closer to zero.

	ZARX	Tetrad	DaO
ER	-0.859	-0.500	-0.264
SFi	-0.788	-0.532	-0.223
SFo	-0.624	-0.126	0.033

Table 3: R^2 rank correlation on DAGs with $|V| = 20$ and $\alpha = 10$ over 100 repetitions

Table 3 tabulates the levels of R^2-sortability for the three simulation methods while varying model structure. ZARX simulations have the highest R^2-sortability for all model structures, while R^2-sortability appears to be present in Tetrad simulations but at a weaker level than in ZARX simulations. Overall, the DaO method has lower R^2-sortability levels. Interestingly, we found that R^2-sortability varies substantially with model structure, as ER models typically had the highest R^2-sortability correlations, and SFo models typically had the lowest R^2-sortability correlations.

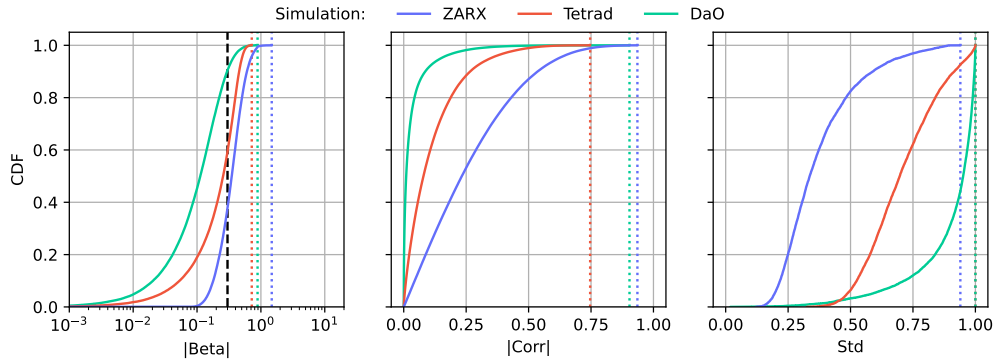

Figure 15: Properties of 100 models generated from SFo-DAGs with $|V| = 100$ and $\alpha = 10$.

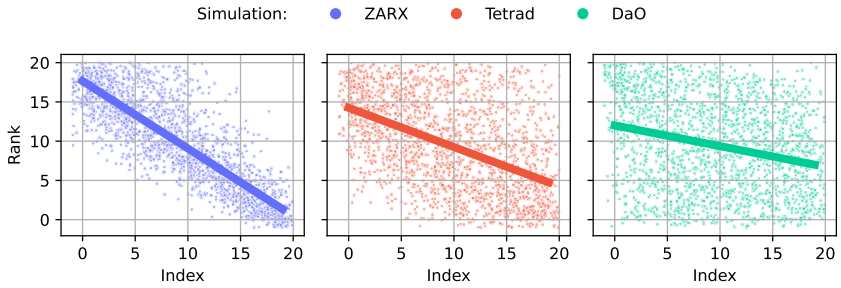

Figure 16: R^2-sortability of models generated from ER-DAGs with $|V| = 20$ and $\alpha = 10$ over 100 repetitions.

A.6 Scale-free Simulation Results

We include an extension to the simulations conducted in Section 5.3 by analyzing data generated from DAGs with scale-free structure. In the DaO simulations, graphs were generated using the SFi-DAG and SFo-DAG methods after the ER-DAG method; see Appendix A.2. The model parameters were passed to Algorithm 4 to generate two versions of the simulated data: Gaussian error and non-Gaussian exponential error. The Gaussian version of the simulated data was used to evaluate all CDAs with one exception. The non-Gaussian version of the simulated data was used to evaluate dLiNGAM due to its dependence on a non-Gaussian signal. Accordingly, dLiNGAM had access to additional information that the other algorithms did not, which is reflected in its superior performance. The σ-sortability algorithm was not evaluated on DaO simulations for a similar reason—there is no signal to sort the variables since all variances are one.

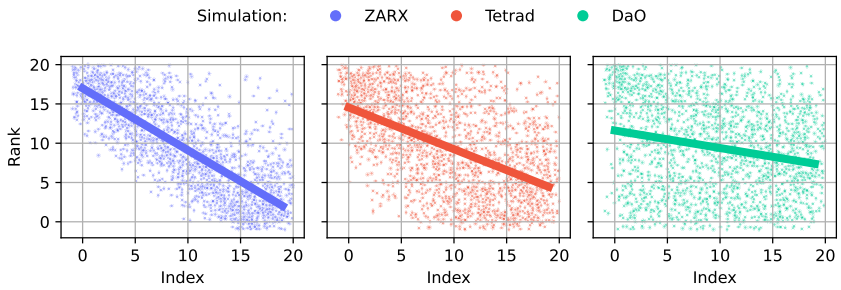

Figure 17: R^2-sortability of models generated from SFi-DAGs with $|V| = 20$ and $\alpha = 10$ over 100 repetitions.

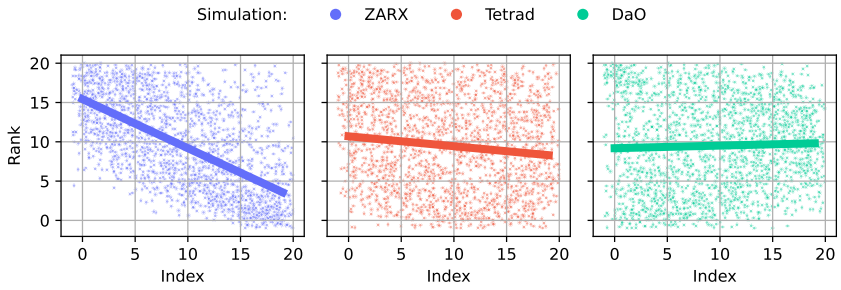

Figure 18: R^2-sortability of models generated from SFo-DAGs with $|V| = 20$ and $\alpha = 10$ over 100 repetitions.

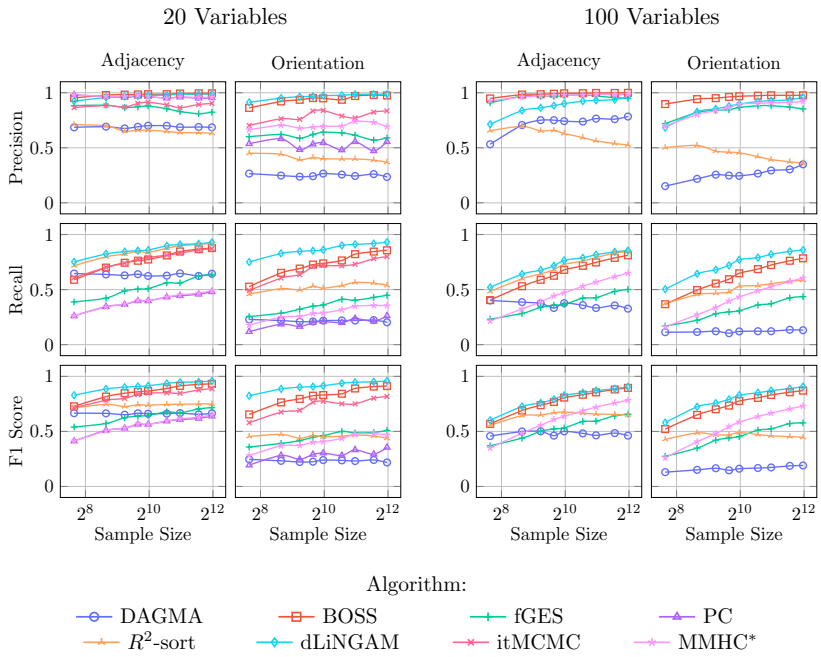

Figure 19: Mean performance of CDAs on data generated from SFi-DAGs with $\alpha = 10$ using the DaO method for DAG Models over 20 repetitions.

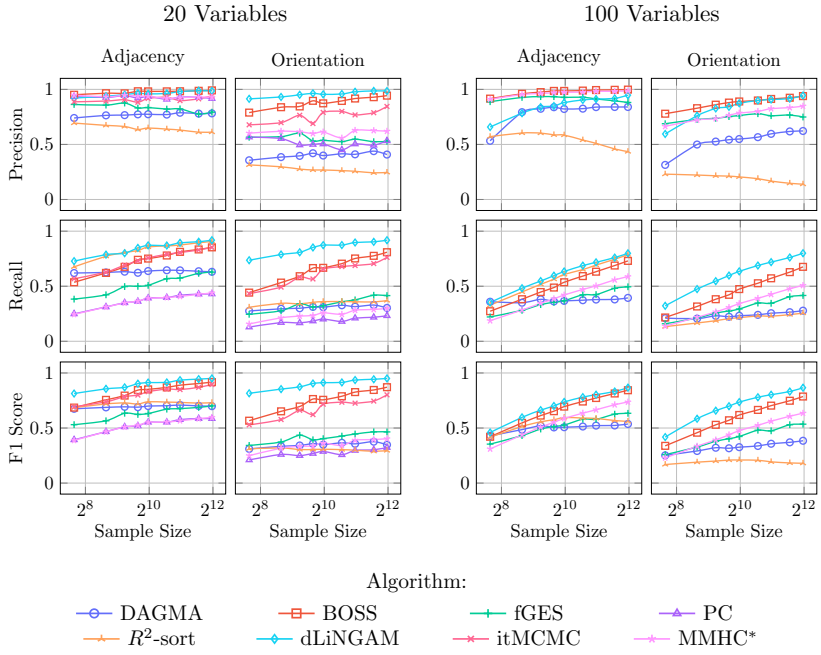

Figure 20: Mean performance of CDAs on data generated from SFo-DAGs with $\alpha = 10$ using the DaO method for DAG Models over 20 repetitions.