CAUSAL DISCOVERY VIA BAYESIAN OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing score-based methods for directed acyclic graph (DAG) learning from observational data struggle to recover the causal graph accurately and sample-efficiently. To overcome this, in this study, we propose **DrBO** (**D**AG **r**ecovery via **B**ayesian **O**ptimization)—a novel DAG learning framework leveraging Bayesian optimization (BO) to find high-scoring DAGs. We show that, by sophisticatedly choosing the promising DAGs to explore, we can find higher-scoring ones much more efficiently. To address the scalability issues of conventional BO in DAG learning, we replace Gaussian Processes commonly employed in BO with dropout neural networks, trained in a continual manner, which allows for (i) flexibly modeling the DAG scores without overfitting, (ii) incorporation of uncertainty into the estimated scores, and (iii) scaling with the number of evaluations. As a result, **DrBO** is computationally efficient and can find the accurate DAG in fewer trials and less time than existing state-of-the-art methods. This is demonstrated through an extensive set of empirical evaluations on many challenging settings with both synthetic and real data.

1 INTRODUCTION

Learning directed acyclic graphs (DAGs) encoding the underlying causal relationships, also known as causal discovery, provides invaluable insights about interventional outcomes and counterfactuals, and thus significant research effort has been dedicated on this frontier. Our study focuses on the score-based framework—a major class of causal discovery methods which casts the DAG learning problem as an optimization problem, maximizing over the space of DAGs a predefined score function measuring how a DAG $\mathcal{G}$ fits the observed data $\mathcal{D}$:

$$\mathcal{G}^* = \underset{\mathcal{G} \in \text{DAGs}}{\arg\max}\, S\left(\mathcal{D}, \mathcal{G}\right). \tag{1}$$

There are several challenges associated with this formulation. First, this optimization problem is NP-hard in general (Chickering, 1996), due to the combinatorial search domain that scales super-exponentially with the graph size (Robinson, 1977) and the acyclicity condition that is nontrivial to maintain. The second challenge is computational cost, as the score evaluation can be expensive for complex models (Zhu et al., 2020; Wang et al., 2021), and hence, methods requiring too many trials will incur a heavy computational expense.

Bayesian Optimization (BO) has emerged as an effective approach for expensive black-box optimization thanks to its sample efficiency.[1] Its applicability covers many domains due to the pervasiveness of optimization tasks in virtually every field. The central idea is that past evaluation data can reveal potential candidates to evaluate next, and effectively exploiting them allows us to arrive at better solutions using fewer evaluations. However, while BO has been employed for *active causal discovery* (Toth et al., 2022; Zhang et al., 2024) to suggest cost-effective intervention strategies to discover causal graphs from active experiments, its power is yet to be harnessed for the problem of *observational causal discovery*, where no active intervention is available.

We have identified two potential difficulties preventing BO to optimize Eq. equation 1. The first is the limited scalability of BO in general, which is usually restricted to only a few hundred dimensions and thousands of evaluations (Wang et al., 2023), while DAG learning often involves far more trials (Zhu et al., 2020; Wang et al., 2021; Duong et al., 2024). The second challenge is how to efficiently optimize the "acquisition function" measuring the potential of DAG solutions in BO. This itself is a score-based DAG learning problem and is expected to be cheaper than the original problem as it is repeated many times in the BO pipeline, thus requiring to be very efficient to be practical.

[1] "Sample" in "sample efficiency" refers to the number of trials.

Present study. In this study, we tackle these obstacles and bring forth the benefits of BO into causal discovery with the introduction of **DrBO** (**D**AG **r**ecovery via **B**ayesian **O**ptimization)[2]—a novel causal discovery algorithm that leverages BO to find the highest-scoring DAG. More particularly, we solve the aforementioned challenges by several key design choices. (i) Inspired by Yu et al. (2021); Massidda et al. (2024); Duong et al. (2024), we devise a low-rank DAG representation that relaxes the constrained optimization in Eq. equation 1 to an *unconstrained optimization problem* with a dimensionality growing linearly with the number of nodes (Sec. 4.1), enabling the use of BO with an amenable dimensionality. (ii) We replace the Gaussian processes (GPs) in conventional BO approaches that scale cubically w.r.t. the number of evaluations with *dropout neural networks* (Srivastava et al., 2014; Guo et al., 2021), which offer expressive uncertainty-aware modeling capabilities, as well as faster acquisition function calculation and optimization (Sec. 4.3). (iii) Our surrogate model *learns the DAG score indirectly* via node-wise local scores, allowing us to predict the DAG scores more accurately with enhanced training information, compared with learning a direct map from a DAG to its score and not exploiting the local scores (Sec. 4.4). (iv) Instead of being retrained every step with all data, our neural networks are *trained continually*, enabling our method to scale better with the number of trials (Sec. 4.5).

Contributions. The main contributions of our study are summarized as follows:

1. To facilitate sample-efficient causal discovery, we propose **DrBO**—a causal discovery method employing BO to optimize for the DAG score. **DrBO** is specifically designed for causal discovery, aiming to be not only accurate, but also computationally manageable. *To our knowledge, this is the first score-based causal discovery method based on BO for purely observational data.*

2. We demonstrate the effectiveness of our method on a comprehensive set of experiments, showing that **DrBO** can consistently surpass state-of-the-art baselines on various conditions, with fewer evaluations and less time, signifying the sample efficiency of BO in our design. In addition, extensive ablation studies verify the significance of our design choices.

2 Related Work

Causal discovery methods can be broadly categorized into two major classes, namely constraint-based and score-based methods. The former category (Spirtes et al., 2000; Colombo et al., 2012) involves performing a series of hypothesis tests to recover the undirected skeleton of the causal graph, before orienting the edges using graphical rules. On the other hand, score-based causal discovery recasts the problem as a combinatorial optimization task. Classical methods (Chickering, 2002; Ramsey et al., 2017; Ramsey, 2015) greedily traverse the DAG space by adding and removing edges one-at-a-time to maintain acyclicity. Recent advances include relaxing the combinatorial optimization to a continuous optimization problem (Zheng et al., 2018; Yu et al., 2019; Zheng et al., 2020; Yu et al., 2021; Zhang et al., 2022; Bello et al., 2022; Annadani et al., 2023). In addition, methods based on Reinforcement learning (RL) (Zhu et al., 2020; Wang et al., 2021; Yang et al., 2023a; Duong et al., 2024) have recently emerged as competitive search strategies. We also acknowledge *interventional causal discovery* studies (Hauser & Bühlmann, 2012; Brouillard et al., 2020; Lippe et al., 2022) where interventional data is exploited to help identify the causal DAG, however, here we focus on the more challenging setting where no intervention is available.

Furthermore, *Bayesian causal discovery* studies are an intriguing direction (Deleu et al., 2022; Tran et al., 2023; Annadani et al., 2023), where the aim is to infer the posterior distribution over causal graphs given observed data, in order to quantify uncertainty in *DAG estimates*. Meanwhile, our use of BO involves quantifying uncertainty in *DAG scores*, so despite sharing the term "Bayesian", these studies are distant from us. Moreover, while Bayesian optimization has been employed for causal discovery in the *active* setting (Toth et al., 2022; Zhang et al., 2024), these methods utilize BO to suggest optimal interventions to quickly recover the causal DAG, necessitating the ability to perform active experiments. Meanwhile, our study utilizes BO to optimize for the score function that can be calculated purely from observational data. In addition, *causal Bayesian optimization* (Aglietti et al., 2020; 2021), which deals with optimizing a variable of interest that is part of a causal system with *known DAG* via a series of interventions suggested by BO, is also a different problem from ours.

[2]The source code will be publicly released upon publication.

3 BACKGROUND

Notations. In this paper, unless specifically indicated, normal lowercase letters indicate scalars (e.g., x, y) or functions (e.g., f, g), while bold lowercase letters represent vectors (e.g., $\mathbf{x}$, $\mathbf{y}$), and bold uppercase letters denote matrices (e.g., $\mathbf{X}$). Meanwhile, subscripts and bracketed superscripts index dimensions and samples, respectively, e.g., $x_i^{(j)}$ denotes the i-th dimension of the j-th sample.

3.1 BAYESIAN OPTIMIZATION

We provide here only the details necessary to understand our contributions. For a more comprehensive review of BO, see Wang et al. (2023), for example. Consider a maximization of a function f that is expensive to evaluate: $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. BO is a class of sequential optimization methods, which iteratively (i) proposes potential candidate(s) to evaluate based on an "acquisition function", then (ii) evaluates said candidate(s), and (iii) updates its statistical model with the newly acquired observations. More specifically, BO defines a probabilistic "surrogate" model over the distribution f given observed data $\mathbf{X}, \mathbf{y}$, i.e., $P(f \mid \mathbf{X}, \mathbf{y})$, to define the acquisition function.

Surrogate model. To model $P(f \mid \mathbf{X}, \mathbf{y})$, Gaussian process (GP) is the standard in BO since it offers a closed-form solution (Rasmussen, 2003). Assuming we have observed a dataset of n evaluations $\mathbf{X} = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right]^\top$ and $\mathbf{y} = \left[y^{(1)}, \ldots, y^{(n)}\right]^\top$, where $\mathbf{x} \sim \mathcal{X} = \mathbb{R}^d$ and $y := f(\mathbf{x})$, GP assumes that $\mathbf{y}$ follows a multivariate Gaussian distribution governed by a mean function $\mu : \mathcal{X} \to \mathbb{R}$ and positive-definite covariance function κ: $\mathcal{X} \times \mathcal{X} \to \mathbb{R}$: $\mathbf{y} \sim \mathcal{N}(\mu(\mathbf{x}), \mathbf{K}_{\mathbf{X},\mathbf{X}})$ where $\mathbf{K}_{\mathbf{X},\mathbf{X}} := \left[\kappa\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right)\right]_{i,j=1,\ldots,n}$ is the $n \times n$ covariance matrix. Using Bayes' theorem, the posterior of the function value at a new location $\mathbf{x}$ is given analytically as: $P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{y}) = \mathcal{N}\left(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\right)$ where $\mu(\mathbf{x}) := \mathbf{k}_{\mathbf{x},\mathbf{X}}\mathbf{K}_{\mathbf{X},\mathbf{X}}^{-1}\mathbf{y}$ and $\sigma^2(\mathbf{x}) := \kappa(\mathbf{x}, \mathbf{x}) - \mathbf{k}_{\mathbf{X},\mathbf{x}}^\top\mathbf{K}_{\mathbf{X},\mathbf{X}}^{-1}\mathbf{k}_{\mathbf{X},\mathbf{x}}$. GPs scale poorly with n due to the need to invert $\mathbf{K}_{\mathbf{X},\mathbf{X}}$, so alternative statistical models like random forest (Hutter et al., 2011), Bayesian linear regression (Snoek et al., 2015), and Bayesian neural network (Springenberg et al., 2016) have been employed as more scalable surrogate models.

Acquisition functions (AFs) in BO judge how promising an arbitrary candidate $\mathbf{x}$ is based on the posterior inferred by the surrogate model, to make a more informed candidate suggestion. As an example, upper confidence bound (UCB) is a common choice, designed to minimize regret in the multi-armed bandit literature (Srinivas et al., 2010) and given by $\mathrm{AF}(\mathbf{x}) := \mu(\mathbf{x}) + \beta\sigma(\mathbf{x})$, where $\beta > 0$ is a hyperparameter controlling the exploitation-exploration trade-off. Another promising acquisition is Thompson sampling (TS, Thompson, 1933), which is a stochastic function that uses a random draw from the posterior as the potential indicator: $\mathrm{AF}(\mathbf{x}) \sim P(y \mid \mathbf{x}, \mathbf{X}, \mathbf{y})$.

3.2 STRUCTURAL CAUSAL MODEL

Let $\mathbf{x} \in \mathbb{R}^d$ be the random vector capturing the system of interest, and $\mathcal{D} = \left\{\mathbf{x}^{(k)}\right\}_{k=1}^n$ denote an i.i.d. dataset of n samples from $P(\mathbf{x})$. The structural causal model (SCM, Pearl, 2000; 2009) among said variables can be described by (i) a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $i \in \mathcal{V} = \{1, \ldots, d\}$ corresponds to a random variable x_i, and each edge $(j \to i) \in \mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ implies that x_j is a direct cause of x_i, (ii) a set of functions $\{f_i\}_{i=1}^n$ dictating the causal mechanisms, and (iii) a noise distribution $P(\boldsymbol{\varepsilon})$. Together, these components define a generative process $x_i := f_i\left(\mathbf{x}_{\mathrm{pa}_i^{\mathcal{G}}}, \varepsilon_i\right), \forall i = 1, \ldots, d$, where $\mathrm{pa}_i^{\mathcal{G}} = \{j \in \mathcal{V} \mid (j \to i) \in \mathcal{E}\}$ is the set of direct causes (a.k.a. structural parents) of node i in $\mathcal{G}$, and $\boldsymbol{\varepsilon} \sim P(\boldsymbol{\varepsilon})$ is the noise vector. Then, the observational causal discovery problem is concerned about recovering the DAG $\mathcal{G}$ from the observational dataset $\mathcal{D}$. In addition, following (Zhu et al., 2020; Wang et al., 2021; Yang et al., 2023a;b; Duong et al., 2024), we also assume: (i) *causal sufficiency*: there is no unobserved confounders among the variables; (ii) *causal minimality*: there is no function f_i that is constant to any of its argument (Peters et al., 2014); and (iii) *identifiable causal models*: this means $\mathcal{G}$ is the unique causal graph that can induce $P(\mathbf{x})$, and thus it is possible to be recovered. For instance, while general linear-Gaussian models are known to be unidentifiable (Spirtes et al., 2000), examples for identifiable causal models include linear-Gaussian models with the equal-variance assumption (Peters et al., 2014) and nonlinear additive noise models (ANMs) in general (Hoyer et al., 2008). Our experiments will adopt these identifiable models.

3.3 SCORE-BASED CAUSAL DISCOVERY

A critical component of score-based causal discovery is the proper specification of a scoring function. With the proper scoring function, the optimization of the score is equivalent to reaching to

ground truth DAG. Consistent scoring functions (Chickering, 2002) are known to satisfy this requirement. In the main text, we demonstrate our method with the Bayesian Information Criterion (BIC, Schwarz, 1978), which is a consistent score as shown by Haughton (1988) and is widely considered in numerous existing studies (Chickering, 2002; Zhu et al., 2020; Wang et al., 2021; Yang et al., 2023a;b; Duong et al., 2024).

More formally, let $\theta := \left\{ \{f_i\}_{i=1}^d, P(\boldsymbol{\varepsilon}) \right\}$ be the parameters of an SCM, then the BIC score is given by $S_{\text{BIC}}(\mathcal{D}, \mathcal{G}) := 2 \ln p\left(\mathcal{D} \mid \hat{\theta}, \mathcal{G}\right) - |\mathcal{G}| \ln n$, where $\hat{\theta} := \arg\max_\theta p(\mathcal{D} \mid \theta, \mathcal{G})$ is the maximum-likelihood estimator of the causal model parameters, n is the sample size of $\mathcal{D}$, and $|\mathcal{G}|$ denotes the number of edges in $\mathcal{G}$. The generality of BIC allows for its adaptation to numerous causal models. In the main paper, we showcase our method with the BIC defined for the popular additive noise model (ANM, Hoyer et al., 2008): $x_i := f_i\left(\mathbf{x}_{\text{pa}_i}\right) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}\left(0, \sigma_i^2\right)$. In the general form, where the noise variances can be non-equal, the BIC for ANM is specified as follows:

$$S_{\text{BIC-NV}}(\mathcal{D}, \mathcal{G}) := -n \sum_{i=1}^{d} \ln \text{MSE}_i\left(\text{pa}_i^{\mathcal{G}}\right) - |\mathcal{G}| \ln n, \tag{2}$$

where NV stands for "non-equal variance" and $\text{MSE}_i\left(\text{pa}_i^{\mathcal{G}}\right) := \frac{1}{n}\sum_{j=1}^n \left(x_i^{(j)} - \hat{f}_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right)\right)^2$ is the mean squared error after regressing x_i on $\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}$. In addition, if we further assume that the noise variables have equal variances (Bühlmann et al., 2014) then the BIC yields:

$$S_{\text{BIC-EV}}(\mathcal{D}, \mathcal{G}) := -nd \ln \frac{\sum_{i=1}^d \text{MSE}_i\left(\text{pa}_i^{\mathcal{G}}\right)}{d} - |\mathcal{G}| \ln n. \tag{3}$$

Eqs. equation 2 and equation 3 are common in prior studies (Zhu et al., 2020; Wang et al., 2021; Yang et al., 2023a;b; Duong et al., 2024), and we also provide their derivation in Appendix B.

3.4 Parametrized DAG generation

For effective acquisition function optimization, we find it crucial for our method to be able to quickly generate candidate DAGs within specific regions determined by a low-dimensional search space. This would help narrow down the regions of interest and improve the quality of suggested candidates. A potential approach towards this end is autoregressive DAG generation techniques (Wang et al., 2021; Deleu et al., 2022; Yang et al., 2023a; Deleu et al., 2024), which break down the DAG generation into multiple sequential steps. However, each step of the generation process is computationally involved due to their autoregressive nature (Duong et al., 2024), accumulating into a significant DAG generation cost. Recently, constraint-free DAG representations with a quadratic complexity are proposed in (Yu et al., 2021; Massidda et al., 2024; Duong et al., 2024), which introduce different maps from an unconstrained real-valued representation to the space of DAGs. For example, the `Vec2DAG` operator in (Duong et al., 2024) takes as input a continuous "node potential" vector $\mathbf{p} \in \mathbb{R}^d$ and strictly upper-triangular "edge potential" matrix $\mathbf{E} \in \mathbb{R}^{d \times d}$ to deterministically create a DAG: $\texttt{Vec2DAG}(\mathbf{p}, \mathbf{E}) := H(\text{grad}(\mathbf{p})) \odot H\left(\mathbf{E} + \mathbf{E}^\top\right)$, where $H(x) := \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise} \end{cases}$ is the entry-wise Heaviside step function, $\odot$ is the Hadamard product, and $\text{grad}(\mathbf{p})_{ij} := p_j - p_i$ is the gradient flow operator (Lim, 2020).

Example. Consider a system of 3 nodes with potentials $\mathbf{p} = [-1, 3, 2]$ and $\mathbf{E} = \begin{bmatrix} 0 & 2 & -4 \\ 0 & 0 & 7 \\ 0 & 0 & 0 \end{bmatrix}$. Then, $\texttt{Vec2DAG}(\mathbf{p}, \mathbf{E}) = H\left(\begin{bmatrix} 0 & 4 & 3 \\ -4 & 0 & -1 \\ -3 & 1 & 0 \end{bmatrix}\right) \odot H\left(\begin{bmatrix} 0 & 2 & -4 \\ 2 & 0 & 7 \\ -4 & 7 & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$. This is the adjacency of the DAG $1 \to 2 \leftarrow 3$, where the edge directions and connectivities are determined by the first and second terms of `Vec2DAG`, respectively.

Using this approach, it is simple to sample candidate DAGs whose representations are in the neighborhood around some values $\mathbf{p}$ and $\mathbf{E}$ of interest.

Algorithm 1 The **DrBO** method for causal discovery.

Require: Dataset $\mathcal{D} = \left\{\mathbf{x}^{(j)} \in \mathbb{R}^d\right\}_{j=1}^n$ of d nodes and n observations, score function $S(\mathcal{D}, \cdot)$, DAG rank k, batch size B, no. of preliminary candidates C, and total no. of evaluations T.

Ensure: A DAG $\hat{\mathcal{G}}$ that maximizes $S(\mathcal{D}, \mathcal{G})$.

1: Initialize empty experience $\mathcal{H} := \emptyset$ and node-wise dropout neural nets: $\{\text{DropoutNN}_i\}_{i=1}^d$.
2: **while** $|\mathcal{H}| < T$ **do**
3: Generate random DAGs: $\left\{\mathcal{G}^{(j)} := \tau\left(\mathbf{z}^{(j)}\right)\right\}_{j=1}^C$ where $\mathbf{z} \in [-1, 1]^{d(1+k)}$. ▷ Sec. 4.1.
4: Sample local scores: $\left\{\left\{l_i^{(j)} \sim \text{DropoutNN}_i\left(\text{pa}_i^{\mathcal{G}^{(j)}}\right)\right\}_{i=1}^d\right\}_{j=1}^C$. ▷ Sec. 4.3.
5: Combine local scores: $\left\{\text{AF}^{(j)} := \text{Combine}\left(l_1^{(j)}, \ldots, l_d^{(j)}\right)\right\}_{j=1}^C$. ▷ Sec. 4.4.
6: Select top B candidates with highest AF values: $j_1, \ldots, j_B := \underset{j=1,\ldots,C}{\text{argtop}_B}\, \text{AF}^{(j)}$. ▷ Sec. 4.2.
7: Evaluate these candidates and update experience: $\mathcal{H} := \mathcal{H} \cup \left\{\left(\mathcal{G}^{(j)}, S\left(\mathcal{D}, \mathcal{G}^{(j)}\right)\right)\right\}_{j=j_1,\ldots,j_B}$.
8: Update the neural nets on new $\mathcal{H}$. ▷ Sec. 4.5.
9: **end while**
10: Get highest-scoring DAG so far: $\hat{\mathcal{G}} := \arg\max_{\mathcal{G} \in \mathcal{H}} S(\mathcal{D}, \mathcal{G})$.
11: Prune $\hat{\mathcal{G}}$ if needed. ▷ Sec. 4.6.

4 DrBO: DAG Recovery via Bayesian Optimization

An overview of our framework is illustrated in Algorithm 1. In the following, we describe the proposed **DrBO** algorithm step-by-step.

4.1 Search Space

To effectively utilize BO, the search space should be unconstrained. Thus, following Yu et al. (2021); Massidda et al. (2024); Duong et al. (2024), we transform the constrained combinatorial optimization problem in Eq. equation 1 to an unconstrained optimization task. However, the dimensionality of $\mathcal{O}\left(d^2\right)$ of their search spaces can still be reduced to mitigate the effect of the curse of dimensionality, facilitating easier acquisition function optimization. For this purpose, we consider a low-rank adaptation (Fang et al., 2023) of `Vec2DAG` (Duong et al., 2024) that offers a search space that grows linearly with the number of nodes. Specifically, each node i is now associated with a low-dimensional embedding vector $r_i \in \mathbb{R}^k$ with $k \ll d$, and two nodes i and j are connected if and only if $\langle r_i, r_j \rangle > 0$. The total dimensionality of this search space is thus $d \cdot (1 + k)$. More formally, given a node potential $\mathbf{p} \in \mathbb{R}^d$ and an embedding matrix $\mathbf{R} \in \mathbb{R}^{d \times k}$, we define the following map

$$\tau(\mathbf{p}, \mathbf{R}) := H(\text{grad}(\mathbf{p})) \odot H\left(\mathbf{R} \cdot \mathbf{R}^\top\right). \tag{4}$$

The following Lemma ensures the acyclicity of the DAG corresponding to $\tau(\mathbf{p}, \mathbf{R})$.

Lemma 1. *For all $d, k \in \mathbb{N}^+$, $\mathbf{p} \in \mathbb{R}^d$ and $\mathbf{R} \in \mathbb{R}^{d \times k}$, let $\tau : \mathbb{R}^d \times \mathbb{R}^{d \times k} \to \{0, 1\}^{d \times d}$ be defined as in Eq. equation 4. Then, $\tau(\mathbf{p}, \mathbf{R})$ represents a binary adjacency matrix of a DAG.*

The proof can be found in Appendix A.1. In addition, like `Vec2DAG`, our variation also exhibits the following scale-invariance property.

Lemma 2. *For all $d, k \in \mathbb{N}^+$, $\mathbf{p} \in \mathbb{R}^d$ and $\mathbf{R} \in \mathbb{R}^{d \times k}$, let $\tau : \mathbb{R}^d \times \mathbb{R}^{d \times k} \to \{0, 1\}^{d \times d}$ be defined as in Eq. equation 4. Then, for all $\alpha > 0$, $\tau(\mathbf{p}, \mathbf{R}) = \tau(\alpha\mathbf{p}, \alpha\mathbf{R})$.*

The proof is provided in Appendix A.2. This insight allows us to restrict the search domain to a fixed range (e.g., $[-1, 1]$) for numerical stability. For brevity, the remaining parts of this manuscript will use vector $\mathbf{z}$ of $d \cdot (1 + k)$ dimensions as the concatenation of $\mathbf{p}$ and the flattened $\mathbf{R}$, and we adopt the notation $\tau(\mathbf{z}) \equiv \tau(\mathbf{p}, \mathbf{R})$. In short, we can now translate the original optimization problem in Eq. equation 1 to the following unconstrained optimization problem:

$$\mathbf{z}^* := \underset{\mathbf{z} \in \mathbb{R}^{d(1+k)}}{\arg\max}\, S(\mathcal{D}, \tau(\mathbf{z})). \tag{5}$$

Obviously, if $k \geq d$ then $\mathbf{R} \cdot \mathbf{R}^\top$ is full-rank, so τ can represent any DAG possible (Theorem 1, Duong et al., 2024), and thus $\mathcal{G}^* := \tau(\mathbf{z}^*)$ is a maximizer of Eq. equation 1 for any maximizer $\mathbf{z}^*$ of Eq. equation 5. While this may not hold for $k < d$, our empirical evaluations reveal that this representation suffices even for very complex graphs.

4.2 Acquisition Function Optimization

To perform each step of the BO pipeline (Sec. 3.1), we use the acquisition function obtained so far to select a batch of B most promising candidates to evaluate, also known as Batch BO (Joy et al., 2020). This is done by first generating $C \geq B$ preliminary candidates $\left\{\mathbf{z}^{(j)}\right\}_{j=1}^{C}$ from a hypercube centered at the current best solution $\mathbf{z}^*$ (see Appendix C.1 for more details). Subsequently, we evaluate the acquisition function of the DAGs induced by these candidates,[3] and choose the top B candidates based on the acquisition function values. The quality of the candidates batch strongly depends on C, i.e., if we can evaluate the acquisition function of a lot of candidates, then the top B candidates are likely to have higher values. However, this also increases computational cost, so our acquisition function must scale well with the number of candidates C to mitigate this overhead, which leads us to the next point.

4.3 Surrogate Modeling with Dropout Networks

To overcome the scalability issues of standard BO due to the use of GPs as discussed earlier, we instead pursue neural networks, which are well-known for their scalability and flexibility (Snoek et al., 2015). Our networks must be able to model uncertainty to help the optimizer prioritize evaluating uncertain but promising candidates. Towards this end, we employ dropout activations (Srivastava et al., 2014), whose original purpose was to reduce overfitting in training neural networks, and were later found to be also useful as an approximate Bayesian inference method (Gal & Ghahramani, 2016), and thus have been successfully applied to BO (Guo et al., 2021). We provide a detailed discussion on this choice compared with other models in Appendix C.2.

Specifically, we devise a single-layer neural network with dropout activation as follows. Let $p \in (0, 1)$ be the dropout rate, d denotes the dimensionality of the input, h is the number of hidden units, $\mathbf{W}_1 \in \mathbb{R}^{d \times h}$ and $\mathbf{W}_2 \in \mathbb{R}^{h \times 1}$ are weight matrices, $\mathbf{b}_1 \in \mathbb{R}^h$ and $b_2 \in \mathbb{R}$ are biases. Our dropout networks are then defined as:

$$\mathrm{DropoutNN}(\mathbf{x}) := \mathbf{W}_2^\top \left(\mathrm{BatchNorm}\left(\mathrm{ReLU}\left(\frac{1}{1-p}\left((1-\mathbf{m}) \circ \left(\mathbf{W}_1^\top \mathbf{x} + \mathbf{b}_1\right)\right)\right)\right)\right) + b_2,$$

where we also follow common practice to employ Batch Normalization (Ioffe & Szegedy, 2015) for improved training efficiency, and $\mathbf{m} \sim \mathrm{Bernoulli}(p)^h$ is drawn for every invocation of $\mathrm{DropoutNN}(\mathbf{x})$ in both train and test modes. By training this model on observed data $\mathbf{X}$ and $\mathbf{y}$ with the square loss, performing a stochastic forward pass $y \sim \mathrm{DropoutNN}(\mathbf{x})$ can be interpreted as drawing from an approximate posterior $y \sim q(y \mid \mathbf{x}, \mathbf{X}, \mathbf{y})$ (Gal & Ghahramani, 2016). Using Thompson sampling, we do not need to characterize the whole posterior and this mere sample suffices for acquisition function optimization (Russo & Van Roy, 2014; Eriksson et al., 2019).

4.4 Indirect Surrogate Modeling

A naïve surrogate modeling approach for DAG learning is modeling a direct map from a DAG to its score. However, DAG scores can typically be decomposed into independent node-wise components, which can be further exploited for better modeling. For example, the BIC scores in Eqs. equation 2 and equation 3 involve the local components $\left\{\mathrm{MSE}_i\left(\mathrm{pa}_i^{\mathcal{G}}\right)\right\}_{i=1}^{d}$ observable after each DAG score invocation. To exploit these information to the fullest, we propose to learn local surrogate models predicting the node-wise scores, then combine them using the rule in Eq. equation 2 or equation 3, depending on the situation.

Particularly, for each node i, we use the evaluation data $\left\{\left(\mathrm{pa}_i^{\mathcal{G}^{(j)}}, \mathrm{MSE}_i\left(\mathrm{pa}_i^{\mathcal{G}^{(j)}}\right)\right)\right\}_{j=1}$, where $\mathcal{G}^{(j)}$ is the j-th evaluated DAG, to train a dropout network $\mathrm{DropoutNN}_i$ predicting $\ln \mathrm{MSE}_i$ from $\mathrm{pa}_i^{\mathcal{G}}$, which is represented by a binary vector of d dimensions. The models are independent among all

[3] Our AFs do not take as input $\mathbf{z}$, but $\tau(\mathbf{z})$ instead, as many $\mathbf{z}$'s can produce the same DAG.

nodes instead of being shared to avoid spurious correlations. To summarize, for a dataset of d nodes, we jointly train d local surrogate models $\{\text{DropoutNN}_i\}_{i=1}^d$. To sample the DAG score of a graph $\mathcal{G}$, we first sample each local score $l_i \sim \text{DropoutNN}_i\left(\text{pa}_i^{\mathcal{G}}\right)$, then combine them with $\text{AF}\left(\mathcal{G}\right) := \text{Combine}_{\text{BIC}-\text{NV}}\left(l_1, \ldots, l_d\right) := -n\sum_{i=1}^d l_i - |\mathcal{G}| \ln n$, if the non-equal variance BIC score is being considered, or $\text{AF}\left(\mathcal{G}\right) := \text{Combine}_{\text{BIC}-\text{EV}}\left(l_1, \ldots, l_d\right) := -nd \ln \frac{\sum_{i=1}^d e^{l_i}}{d} - |\mathcal{G}| \ln n$ for the case of equal variance BIC score, which resemble Eqs. equation 2 and equation 3, respectively.

4.5 Continual Model Training

Upon acquiring a batch of B of new evaluations, the neural networks require retraining to update their weights. To avoid retraining with all data so far, which scales at least quadratically with the number of evaluations, we adopt a continual training approach (Wang et al., 2024) that only updates the models with the new data and a small portion of past data to mitigate forgetting. Specifically, we perform n_{grads} gradient steps within each BO iteration, each of which is calculated on the B new datapoints and a replay buffer of n_{replay} past observations using Reservoir Sampling (Vitter, 1985).

4.6 Finalizing the Result

Pruning the resultant DAG is common practice to suppress the redundant edges (Bühlmann et al., 2014; Zheng et al., 2018; Wang et al., 2021; Bello et al., 2022; Duong et al., 2024), and is also employed in our framework. This can be done by thresholding the weight matrix for linear data (Zheng et al., 2018), or employing significance testing for nonlinear data using generalized additive model regression (CAM pruning, Peters et al., 2014), or conditional independence testing under the faithfulness assumption (Duong et al., 2024). More details are provided in Appendix C.3.

5 Experiments

In this section, we verify our claim in the introduction: **DrBO** *is both more accurate and sample-efficient than existing approaches in score-based observational DAG learning*. We show this by comparing our **DrBO** method with a number of the most recent advances in causal discovery that are based on sequential optimization, including gradient-based methods DAGMA (Bello et al., 2022), COSMO (Massidda et al., 2024), GOLEM (Ng et al., 2020), NOTEARS (Zheng et al., 2018) with TMPI constraint (Zhang et al., 2022), as well as RL-based approaches CORL (Wang et al., 2021) and ALIAS (Duong et al., 2024). We note that CORL, ALIAS, and **DrBO** directly optimize the BIC score, COSMO, DAGMA, and NOTEARS optimize the penalized least-square loss, while GOLEM optimizes a penalized log-likelihood. For gradient-based methods, we consider a gradient update equivalent to one DAG evaluation. Additional information, including implementation details and metrics, are provided in Appendix D.

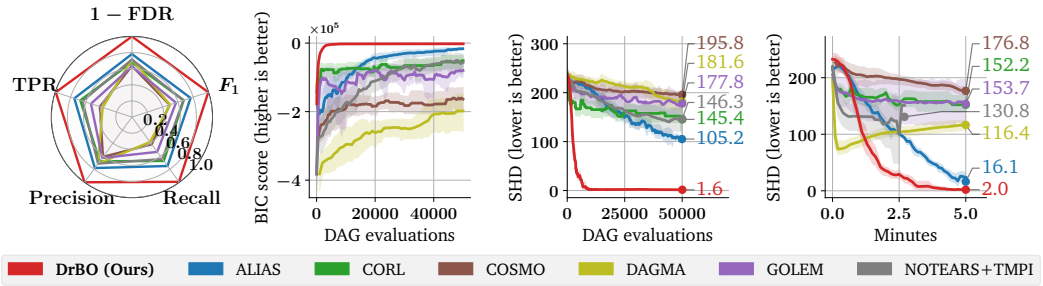
5.1 Results on Synthetic Data

We consider the standard Erdős-Rényi (ER) graph model (Erdős & Rényi, 1960) to generate data, where graphs with d nodes and de edges on average are referred to as dERe graphs (e.g, 10ER4).

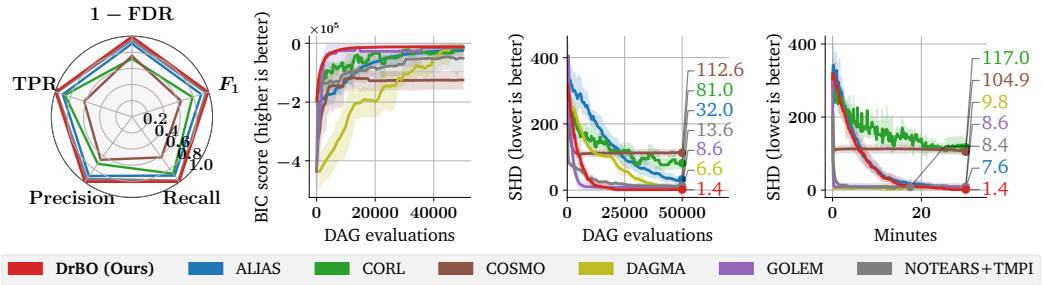
5.1.1 Linear-Gaussian Data

After simulating a DAG $\mathcal{G}$, we sample edge weights with $w_{ji} \sim \mathcal{U}\left([-2, -0.5] \cup [0.5, 2]\right)$ like prior studies (Zheng et al., 2018; Zhu et al., 2020; Wang et al., 2021). Then, we generate a dataset of $n = 1{,}000$ i.i.d. samples according to a linear-Gaussian SCM $x_i := \sum_{j \in \text{pa}_i} w_{ji} x_j + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}\left(0, 1\right)$. For fairness, we prune the DAGs returned by **DrBO**, along with ALIAS and CORL as prescribed in their papers, by thresholding the weight matrix obtained via linear regression at 0.3. This is not done for DAGMA and COSMO because their implementations already incorporated the same pruning scheme.

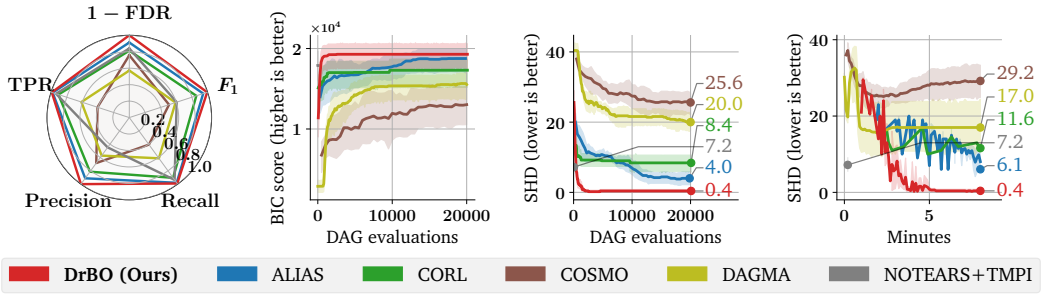
Dense graphs. We stress-test our method with complex structures in Figure 1(a). As depicted in the first column, our method is the only approach that can achieve absolute overall performance in all five metrics, surpassing the second- and third-best methods ALIAS and CORL by large margins of more than 20% in each metric, while gradient-based methods DAGMA and COSMO struggle with much worse performance. From the second and third columns, our **DrBO** approach can reach higher BIC scores, and thus lower SHDs, very sharply with the number of evaluations, highlighting



(a) **Dense graphs** with Linear-Gaussian data (DAGs with **30 nodes** and ≈**240 edges**). For fairness, summary metrics (first column) are calculated at 50,000 evaluations for all methods.



(b) **Large graphs** with Linear-Gaussian data (DAGs with **100 nodes** and ≈**200 edges**). For fairness, summary metrics (first column) are calculated at 50,000 evaluations for all methods.



(c) **Nonlinear data** with Gaussian processes (DAGs with **10 nodes** and ≈**40 edges**). For fairness, summary metrics (first column) are calculated at 20,000 evaluations for all methods.

Figure 1: **DAG learning results on Synthetic data.** *First column:* overall performance in terms of True Positive Rate (TPR, higher is better), Precision, Recall, and F_1 score (higher is better), as well as False Discovery Rate (FDR, lower is better). *Second column:* we track the best Bayesian Information Criterion (BIC, higher is better) so far at every optimization step. *Third and Fourth columns*: we monitor the Structural Hamming Distance (SHD, lower is better) of the DAG whose best BIC so far at every optimization step. Shaded areas in the line plots indicate 95% confidence intervals over 5 random datasets. NOTEARS+TMPI usually stops early before the time limit.

its sample-efficiency. Lastly, the fourth column shows that our method's SHD improvement over time is continuous and faster in terms of runtime than all other baselines.

Large-scale graphs. Next, we demonstrate the scalability of our method on high-dimensional graphs of 100 nodes, having up to 10,000 edges. Our results in Figure 1(b) shows that **DrBO** is still the leading method for high-dim data, where it obtains absolute overall performance along with the lowest SHD among all methods in limited runtime.

5.1.2 Nonlinear data

Following prior works (Zhu et al., 2020; Wang et al., 2021; Yang et al., 2023a; Duong et al., 2024), we demonstrate our method on nonlinear datasets generated using Gaussian processes from Lachapelle et al. (2020). Specifically, we employ the datasets generated according to an ANM $x_i := f_i\left(\mathbf{x}_{\mathrm{pa}_i}\right) + \varepsilon_i$ where f_i is drawn from a Gaussian process prior with a unit bandwidth RBF

kernel, and $\varepsilon_i \sim \mathcal{N}\left(0, \sigma_i^2\right)$ with non-equal noise variances σ_i^2 sampled uniformly in $[0.4, 0.8]$. ALIAS, CORL, and **DrBO** optimize $S_{\text{BIC}-\text{NV}}$, while COSMO, DAGMA, and NOTEARS+TMPI optimize the least-square objective. The empirical results reported in Figure 1(c) confirms that our **DrBO** method also excels on complex nonlinear data in both accuracy and computational cost. This is evidenced by the leading overall performance (first column), along with a vanishing SHD of only 0.4 at the end of the learning curve, surpassing other methods by a visible gap both in SHD and convergence speed.

5.2 Results on Real Data and Structures

Table 1: **Causal Discovery Performance on Real-world Structures (Scutari, 2010)**. The performance is measured in Structural Hamming Distance (SHD, lower is better). The numbers are $\text{mean} \pm \text{std}$ over 5 independent datasets with 1,000 observational samples. For fairness, all methods are limited to 20,000 evaluations.

Dataset Method	Alarm (37 nodes, 46 edges)	Asia (8 nodes, 8 edges)	Cancer (5 nodes, 4 edges)	Child (20 nodes, 25 edges)	Earthquake (5 nodes, 4 edges)
ALIAS (Duong et al., 2024)	26.8 ± 7.8	0.2 ± 0.5	0.0 ± 0.0	2.8 ± 1.6	$\mathbf{0.0 \pm 0.0}$
CORL (Wang et al., 2021)	19.8 ± 8.6	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	1.6 ± 2.3	$\mathbf{0.0 \pm 0.0}$
COSMO (Massidda et al., 2024)	26.8 ± 4.0	4.0 ± 1.6	2.2 ± 0.8	11.6 ± 2.3	2.2 ± 0.8
DAGMA (Bello et al., 2022)	25.0 ± 5.3	2.6 ± 1.3	1.0 ± 1.2	8.0 ± 5.2	1.0 ± 1.2
GOLEM (Ng et al., 2020)	4.8 ± 6.2	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
NOTEARS+TMPI (Zheng et al., 2018; Zhang et al., 2022)	7.0 ± 7.9	0.4 ± 0.9	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
DrBO (Ours)	$\mathbf{1.0 \pm 2.2}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$

Benchmark data. We verify the performance of our method on real data using the popular benchmark flow cytometry dataset (Sachs et al., 2005), concerning a protein signaling network based on expression levels of proteins and phospholipids. We employ the observational portion of the dataset containing 853 observations and a known causal network with 11 nodes and 17 edges. The evaluations shown in Figure 2 verifies the effectiveness of our method on real data, where it effortlessly achieves a *state-of-the-art SHD of 9* with fewer evaluations compared with the competitors.

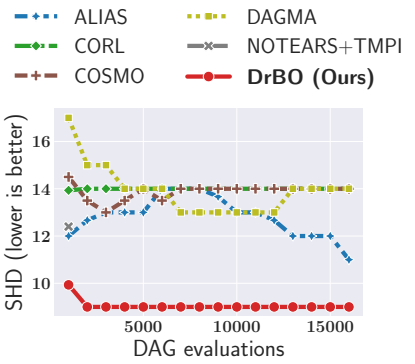


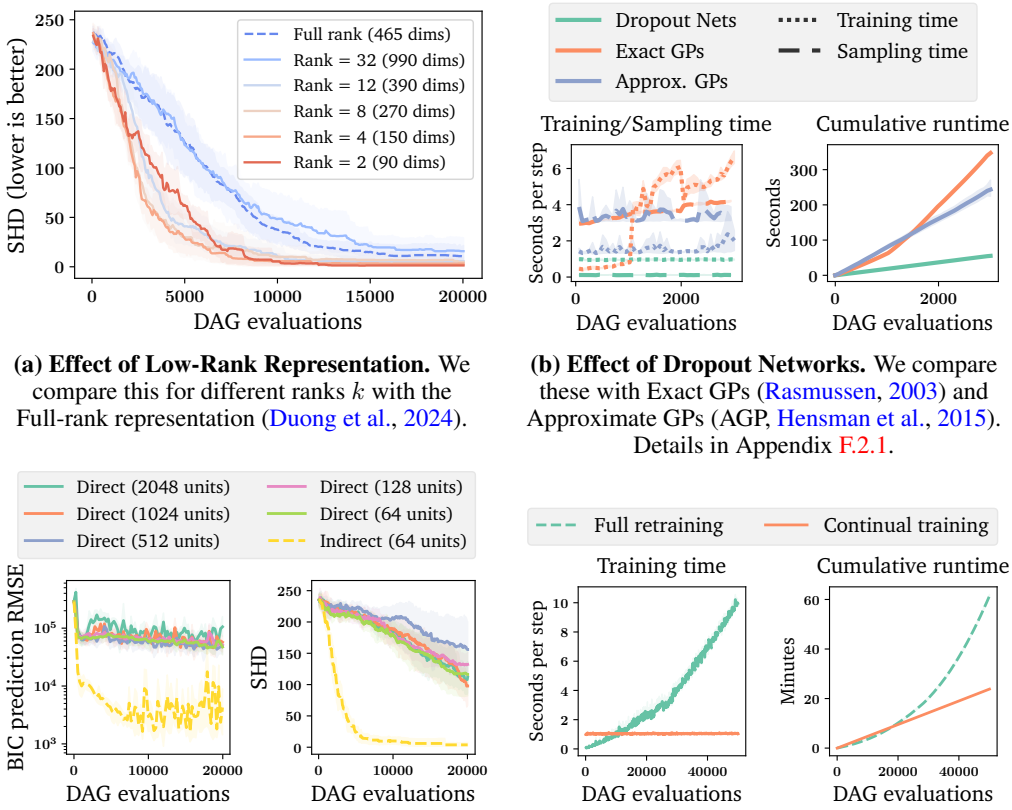
Figure 2: **Causal Discovery Performance on the Benchmark Sachs Dataset (Sachs et al., 2005)**. NOTEARS+TMPI stops early before the max no. of evaluations is reached.

Real-world structures. To further illustrate the capabilities of our approach on real-world scenarios, we conduct experiments on real structures provided by the BnLearn repository (Scutari, 2010). Each dataset contains 1,000 observational samples and a ground truth causal network belonging to real-world applications with varying size. Additional details regarding these datasets are given in Appendix D.3. The results are presented in Table 1, highlighting that our method is the only approach that can consistently achieve zero SHD on four out of five real-world structures. On the Alarm dataset, which appears to be most challenging, our **DrBO** method still leads with much lower SHD compared with all other baselines.

5.3 Supplementary Results

Extended Causal Discovery Settings. We investigate the performance of **DrBO** in extended scenarios in Appendix F as follows. **Varying sample sizes:** we show in Figure 4 that our method can achieve low SHDs even with limited data. **Different graph models:** in Table 3, **DrBO** achieves low SHDs and surpasses competitors for both ER and SF graphs, even on the dense graphs. **Different noise distributions:** Table 4 shows that **DrBO** also outperforms the baselines under five different noise types. **BGe score:** Figure 5 confirms that our method can work with a different score and match the score of ground truth graphs with low structural errors. **Discrete Data:** in Figure 6, we show that **DrBO** also obtains the highest scores and lowest SHDs compared with the baselines for non-continuous data.

Comparison with ordering-based methods. Ordering-based methods are also prominent in causal discovery, but they are not compared with yet as they are not sequential optimizers. Notwithstanding, we show in Table 5 (Appendix) that **DrBO** also significantly outperforms popular ordering-based methods on linear, nonlinear, as well as real data.



(a) Effect of Low-Rank Representation. We compare this for different ranks k with the Full-rank representation (Duong et al., 2024).

(b) Effect of Dropout Networks. We compare these with Exact GPs (Rasmussen, 2003) and Approximate GPs (AGP, Hensman et al., 2015). Details in Appendix F.2.1.

(c) Effect of Indirect DAG Modeling. We compare this with learning a Direct map from a DAG to its score, with varying no. hidden units.

(d) Effect of Continual Training. We compare this approach with Full retraining using all experienced data at every BO iteration. Details in Appendix F.2.2.

Figure 3: **Ablating our design choices.** All configurations are evaluated on 5 linear-Gaussian datasets of 1,000 samples on 30ER8 graphs. Shaded areas indicate 95% confidence intervals.

Runtime Comparison. In Appendix H, we also detail the numerical runtime among all methods.

5.4 Ablations

In Figure 3, through ablation studies, we justify the design choices outlined in the Introduction, showing that every component contributes considerately to the accuracy and/or scalability of our method. The remaining hyperparameters of our algorithm are also studied in Appendix F.2.

6 Concluding Remarks

This study presents **DrBO**, a novel BO method to search for high-scoring DAGs. We have shown that, by meticulously choosing promising DAGs to evaluate, we can find the optimal one more efficiently and cost-effectively. Our comprehensive experiments demonstrate that **DrBO** performs well even in many intricate settings like dense graphs, high-dimensional, and nonlinear data.

Regarding limitations, in our method, the surrogate model architecture is manually chosen and remains fixed through the course of optimization, and thus is prone underfitting at the end. To mitigate this, incremental neural architecture search techniques (Liu et al., 2018; Geifman & El-Yaniv, 2019) can be employed to facilitate autonomous architecture selection and scaling. In addition, our continual learning component is simple and can benefit from more advanced techniques (Wang et al., 2024) to further improve the performance of our surrogate models.

For future developments, it would be an interesting direction to combine our approach with active causal discovery methods to recover the causal structure even more efficiently. Moreover, our method can be extended to solve causal discovery problems with hidden confounders, where the outputs are no longer DAGs.

REFERENCES

Virginia Aglietti, Xiaoyu Lu, Andrei Paleyes, and Javier González. Causal Bayesian optimization. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 3155–3164. PMLR, 2020.

Virginia Aglietti, Neil Dhir, Javier González, and Theodoros Damoulas. Dynamic causal bayesian optimization. In *Advances in Neural Information Processing Systems*, pp. 10549–10560, 2021.

Yashas Annadani, Nick Pawlowski, Joel Jennings, Stefan Bauer, Cheng Zhang, and Wenbo Gong. BayesDAG: Gradient-based posterior sampling for causal discovery. In *Advances in Neural Information Processing Systems*, 2023.

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

Ingo A Beinlich, Henri Jacques Suermondt, R Martin Chavez, and Gregory F Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the European Conference on Artificial Intelligence in Medicine*, pp. 247–256, 1989.

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. DAGMA: learning DAGs via M-matrices and a log-determinant acyclicity characterization. In *Advances in Neural Information Processing Systems*, pp. 8226–8239, 2022.

Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and Alexandre Drouin. Differentiable causal discovery from interventional data. *Advances in Neural Information Processing Systems*, 33:21865–21877, 2020.

Peter Bühlmann, Jonas Peters, and Jan Ernest. CAM: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42:2526–2556, 2014.

David Maxwell Chickering. Learning Bayesian networks is NP-complete. *Learning from data: Artificial intelligence and statistics V*, pp. 121–130, 1996.

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

Diego Colombo, Marloes H Maathuis, Markus Kalisch, and Thomas S Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, pp. 294–321, 2012.

Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Multi-objective bayesian optimization over high-dimensional search spaces. In *Uncertainty in Artificial Intelligence*, pp. 507–517. PMLR, 2022.

Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. In *Proceedings of the Uncertainty in Artificial Intelligence*, pp. 518–528, 2022.

Tristan Deleu, Mizu Nishikawa-Toomey, Jithendaraa Subramanian, Nikolay Malkin, Laurent Charlin, and Yoshua Bengio. Joint Bayesian inference of graphical structure and parameters with a single generative flow network. In *Advances in Neural Information Processing Systems*, 2024.

Bao Duong, Hung Le, and Thin Nguyen. ALIAS: DAG learning with efficient unconstrained policies. *arXiv preprint arXiv:2408.13448*, 2024.

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 1960.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Zhuangyan Fang, Shengyu Zhu, Jiji Zhang, Yue Liu, Zhitang Chen, and Yangbo He. On low-rank directed acyclic graphs and causal structure learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4):4924–4937, 2023.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning*, pp. 1050–1059, 2016.

Yonatan Geifman and Ran El-Yaniv. Deep active learning with a neural architecture search. In *Advances in Neural Information Processing Systems*, 2019.

Dan Geiger and David Heckerman. Learning Gaussian networks. In *Uncertainty Proceedings*, pp. 235–243. 1994.

Dan Guo, Yaochu Jin, Jinliang Ding, and Tianyou Chai. Heterogeneous ensemble-based infill criterion for evolutionary multiobjective optimization of expensive problems. *IEEE Transactions on Cybernetics*, 49(3):1012–1025, 2018.

Dan Guo, Xilu Wang, Kailai Gao, Yaochu Jin, Jinliang Ding, and Tianyou Chai. Evolutionary optimization of high-dimensional multiobjective and many-objective expensive problems assisted by a dropout neural network. *IEEE transactions on systems, man, and cybernetics: systems*, 52: 2084–2097, 2021.

Dominique Marie-Annick Haughton. On the choice of a model to fit data from an exponential family. *The Annals of Statistics*, pp. 342–355, 1988.

Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13(1): 2409–2464, 2012.

David Heckerman, Dan Geiger, and David M Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, pp. 197–243, 1995.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 351–360. PMLR, 2015.

Patrik Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems*, 2008.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5*, pp. 507–523. Springer, 2011.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning*, 2015.

Tinu Theckel Joy, Santu Rana, Sunil Gupta, and Svetha Venkatesh. Batch Bayesian optimization using multi-scale search. *Knowledge-Based Systems*, 187:104818, 2020.

Daphne Koller and Mehran Sahami. Toward optimal feature selection. Technical report, 1996.

Kevin B Korb and Ann E Nicholson. *Bayesian artificial intelligence*. CRC press, 2010.

Jack Kuipers, Giusi Moffa, and David Heckerman. Addendum on the scoring of Gaussian directed acyclic graphical models. *The Annals of Statistics*, 42(4):1689 – 1691, 2014.

Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural DAG learning. In *Proceedings of the International Conference on Learning Representations*, 2020.

Steffen L Lauritzen and David J Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194, 1988.

Lek-Heng Lim. Hodge Laplacians on graphs. *Siam Review*, 62:685–715, 2020.

Phillip Lippe, Taco Cohen, and Efstratios Gavves. Efficient neural causal discovery without acyclicity constraints. In *Proceedings of the International Conference on Learning Representations*, 2022.

Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European Conference on Computer Vision*, pp. 19–34, 2018.

Riccardo Massidda, Francesco Landolfi, Martina Cinquini, and Davide Bacciu. Constraint-free structure learning with smooth acyclic orientations. In *Proceedings of the International Conference on Learning Representations*, 2024.

Michael D McKay, Richard J Beckman, and William J Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

Francesco Montagna, Nicoletta Noceti, Lorenzo Rosasco, Kun Zhang, and Francesco Locatello. Scalable causal discovery with score matching. In *Conference on Causal Learning and Reasoning*, pp. 752–771. PMLR, 2023.

Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and DAG constraints for learning linear DAGs. In *Advances in Neural Information Processing Systems*, pp. 17943–17954, 2020.

Art B Owen. Scrambling Sobol' and Niederreiter–Xing points. *Journal of complexity*, 14(4):466–489, 1998.

Judea Pearl. *Models, reasoning and inference*. Cambridge University Press, 2000.

Judea Pearl. *Causality*. Cambridge University Press, 2009.

Jonas Peters, Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *Journal of Machine Learning Research*, 2014.

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

Joseph Ramsey, Madelyn Glymour, Ruben Sanchez-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. Scaling up greedy causal search for continuous variables. *arXiv preprint arXiv:1507.07749*, 2015.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pp. 63–71. Springer, 2003.

Rommel G Regis and Christine A Shoemaker. Combining radial basis function surrogates and dynamic coordinate search in high-dimensional expensive black-box optimization. *Engineering Optimization*, 45(5):529–555, 2013.

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

Robert W Robinson. Counting unlabeled acyclic digraphs. In *Combinatorial Mathematics V*, pp. 28–43. Springer, 1977.

Paul Rolland, Volkan Cevher, Matthäus Kleindessner, Chris Russell, Dominik Janzing, Bernhard Schölkopf, and Francesco Locatello. Score matching enables causal discovery of nonlinear additive noise models. In *Proceedings of the International Conference on Machine Learning*, pp. 18741–18753, 2022.

Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308:523–529, 2005.

Pedro Sanchez, Xiao Liu, Alison Q O'Neil, and Sotirios A. Tsaftaris. Diffusion models for causal discovery via topological ordering. In *Proceedings of the International Conference on Learning Representations*, 2023.

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

Marco Scutari. Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35:1–22, 2010.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18, 2005.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *Proceedings of the International Conference on Machine Learning*, pp. 2171–2180. PMLR, 2015.

David J Spiegelhalter. Learning in probabilistic expert systems. *Bayesian statistics*, 4:447–465, 1992.

Peter Spirtes, Clark N Glymour, Richard Scheines, and David Heckerman. *Causation, prediction, and search*. MIT Press, 2000.

Jost Tobias Springenberg, Aaron Klein, Stefan Falkner, and Frank Hutter. Bayesian optimization with robust Bayesian neural networks. *Advances in Neural Information Processing Systems*, 29, 2016.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the International Conference on Machine Learning*, 2010.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the Artificial Intelligence and Statistics*, pp. 567–574. PMLR, 2009.

Christian Toth, Lars Lorch, Christian Knoll, Andreas Krause, Franz Pernkopf, Robert Peharz, and Julius Von Kügelgen. Active bayesian causal inference. In *Advances in Neural Information Processing Systems*, pp. 16261–16275, 2022.

Quang-Duy Tran, Phuoc Nguyen, Bao Duong, and Thin Nguyen. Differentiable bayesian structure learning with acyclicity assurance. In *Proceedings of the IEEE International Conference on Data Mining*, pp. 598–607, 2023.

Paul E Utgoff. Incremental induction of decision trees. *Machine learning*, 4:161–186, 1989.

Paul E Utgoff, Neil C Berkman, and Jeffery A Clouse. Decision tree induction based on efficient tree restructuring. *Machine Learning*, 29:5–44, 1997.

Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Xiaoqiang Wang, Yali Du, Shengyu Zhu, Liangjun Ke, Zhitang Chen, Jianye Hao, and Jun Wang. Ordering-based causal discovery with reinforcement learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 3566–3573, 2021.

Xilu Wang, Yaochu Jin, Sebastian Schmitt, and Markus Olhofer. Recent advances in Bayesian optimization. *ACM Computing Surveys*, 55:1–36, 2023.

Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale Bayesian optimization in high-dimensional spaces. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 745–754. PMLR, 2018.

Eric P Xing, Michael I Jordan, Richard M Karp, et al. Feature selection for high-dimensional genomic microarray data. In *Proceedings of the International Conference on Machine Learning*, volume 1, pp. 601–608. Citeseer, 2001.

Dezhi Yang, Guoxian Yu, Jun Wang, Zhengtian Wu, and Maozu Guo. Reinforcement causal structure learning on order graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 10737–10744, 2023a.

Dezhi Yang, Guoxian Yu, Jun Wang, Zhongmin Yan, and Maozu Guo. Causal discovery by graph attention reinforcement learning. In *Proceedings of the SIAM International Conference on Data Mining*, pp. 28–36, 2023b.

Yue Yu, Jie Chen, Tian Gao, and Mo Yu. DAG-GNN: DAG structure learning with graph neural networks. In *Proceedings of the International Conference on Machine Learning*, pp. 7154–7163, 2019.

Yue Yu, Tian Gao, Naiyu Yin, and Qiang Ji. DAGs with No Curl: An efficient DAG structure learning approach. In *Proceedings of the International Conference on Machine Learning*, pp. 12156–12166, 2021.

Keli Zhang, Shengyu Zhu, Marcus Kalander, Ignavier Ng, Junjian Ye, Zhitang Chen, and Lujia Pan. gCastle: A python toolbox for causal discovery. *arXiv preprint arXiv:2111.15155*, 2021.

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 804–813, 2011.

Zeyu Zhang, Chaozhuo Li, Xu Chen, and Xing Xie. Bayesian active causal discovery with multi-fidelity experiments. In *Advances in Neural Information Processing Systems*, 2024.

Zhen Zhang, Ignavier Ng, Dong Gong, Yuhang Liu, Ehsan Abbasnejad, Mingming Gong, Kun Zhang, and Javen Qinfeng Shi. Truncated matrix power iteration for differentiable DAG learning. In *Advances in Neural Information Processing Systems*, pp. 18390–18402, 2022.

Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, pp. 9472–9483, 2018.

Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. Learning sparse nonparametric DAGs. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2020.

Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning. In *Proceedings of the International Conference on Learning Representations*, 2020.

A PROOFS

A.1 PROOF OF LEMMA 1

Proof. The acyclicity of Eq. equation 4 is ensured by the first term: $H\left(\operatorname{grad}(\mathbf{p})\right)$. This is a binary matrix representing the adjacency matrix of a directed graph. In this graph, the presence of an edge $i \to j$ is equivalent to $p_i < p_j$. As such, a cycle $i_1 \to \ldots \to i_1$, if exists, would lead to $p_i < p_i$, which is contradictory, meaning this graph must be a DAG. Multiplying this adjacency matrix with the second term $H\left(\mathbf{R} \cdot \mathbf{R}^\top\right)$, which is also a binary matrix, has the effect of removing already existing edges in this graph, and thus cannot introduce any cycle. This concludes our proof. □

A.2 PROOF OF LEMMA 2

Proof. For any $\alpha > 0$, we have $H\left(p_i - p_j\right) = H\left(\alpha\left(p_i - p_j\right)\right) = H\left(\alpha p_i - \alpha p_j\right)$ and $H\left(\mathbf{R} \cdot \mathbf{R}^\top\right) = H\left(\alpha^2 \mathbf{R} \cdot \mathbf{R}^\top\right) = H\left((\alpha\mathbf{R}) \cdot (\alpha\mathbf{R})^\top\right)$. Thus, $H\left(\operatorname{grad}(\mathbf{p})\right) \odot H\left(\mathbf{R} \cdot \mathbf{R}^\top\right) = H\left(\operatorname{grad}(\alpha\mathbf{p})\right) \odot H\left((\alpha\mathbf{R}) \cdot (\alpha\mathbf{R})^\top\right)$, concluding our proof. □

B DERIVING BIC SCORES

Recall that for a causal model with parameters $\theta := \left\{\{f_i\}_{i=1}^d, P(\boldsymbol{\varepsilon})\right\}$, the general BIC is given by

$$S_{\text{BIC}}(\mathcal{D}, \mathcal{G}) := 2 \ln p\left(\mathcal{D} \mid \hat{\theta}, \mathcal{G}\right) - |\mathcal{G}| \ln n, \tag{6}$$

where $\hat{\theta} := \arg\max_\theta p(\mathcal{D} \mid \theta, \mathcal{G})$ is the maximum-likelihood estimator of the causal model parameters, n is the sample size of $\mathcal{D}$, and $|\mathcal{G}|$ denotes the number of edges in $\mathcal{G}$. The first term in Eq. equation 6 is a log-likelihood objective similar to GOLEM (Ng et al., 2020), GraN-DAG (Lachapelle et al., 2020), etc., while the second term penalizes extra edges.

The flexibility of Eq. equation 6 allows for the adoption of BIC in various causal models by simply specifying the likelihood model. In the following, we derive the BIC scores for additive noise models with non-equal and equal variances, as well as logistic models.

B.1 BIC FOR ANM WITH NON-QUAL VARIANCES

Let us consider an ANM defined by

$$x_i := f_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}\right) + \varepsilon_i, \ \forall i = 1, \ldots, d, \tag{7}$$

where the noise is assumed to be Gaussian with fixed variance: $\varepsilon_i \sim \mathcal{N}\left(0, \sigma_i^2\right) \ \forall i = 1, \ldots d$. The likelihood of a dataset $\mathcal{D} = \left\{\mathbf{x}^{(j)}\right\}_{j=1}^n$ under this model is given by

$$\mathcal{L} = \ln p\left(\mathcal{D} \mid \mathcal{G}, \{f_i\}_{i=1}^d, \{\sigma_i\}_{i=1}^d\right) = -\frac{1}{2} \sum_{i=1}^d \frac{\sum_{j=1}^n \left(x_i^{(j)} - f_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right)\right)^2}{\sigma_i^2} - \frac{n}{2} \sum_{i=1}^d \ln \sigma_i^2 + \text{constant}. \tag{8}$$

Taking its derivative and setting it to zero, we can solve for the parameters as follows:

$$\hat{\sigma}_i^2 = \underbrace{\frac{1}{n} \sum_{j=1}^n \left(x_i^{(j)} - \hat{f}_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right)\right)^2}_{\text{MSE}_i}, \tag{9}$$

where $\left\{\hat{f}_i\right\}_{i=1}^d$ are least-square estimators obtained by minimizing $\left(x_i^{(j)} - f_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right)\right)^2$ over a restricted hypothesis class $f \sim \mathcal{F}$. Following past studies (Zhu et al., 2020; Wang et al., 2021; Yang

et al., 2023a;b; Duong et al., 2024), we use linear regression for linear data and Gaussian process regression for nonlinear data. However, any other regression method can be used as desired.

Subsequently, the first term in Eq. equation 8 cancels out and the maximum log-likelihood is reduced to

$$\hat{\mathcal{L}} = -\frac{n}{2} \sum_{i=1}^{d} \ln \text{MSE}_i + \text{constant}. \tag{10}$$

Finally, the BIC score for ANMs with non-equal variances is obtained by:

$$S_{\text{BIC-NV}}(\mathcal{D}, \mathcal{G}) = -n \sum_{i=1}^{d} \ln \text{MSE}_i - |\mathcal{G}| \ln n. \tag{11}$$

B.2 BIC for ANM with Equal Variances

By setting $\sigma_i = \sigma \; \forall i = 1, \ldots, d$, we repeat the previous steps and obtain the following solution for the maximum likelihood estimators:

$$\hat{\sigma}^2 = \frac{1}{d} \sum_{i=1}^{d} \underbrace{\frac{1}{n} \sum_{j=1}^{n} \left(x_i^{(j)} - \hat{f}_i \left(\mathbf{x}_{\text{pa}_i}^{(j)} \right) \right)^2}_{\text{MSE}_i}. \tag{12}$$

The maximum likelihood now is given by

$$\hat{\mathcal{L}} = -\frac{nd}{2} \ln \frac{\sum_{i=1}^{d} \text{MSE}_i}{d} + \text{constant}. \tag{13}$$

Finally, we obtain the the BIC score for ANMs with equal variances as follows:

$$S_{\text{BIC-EV}}(\mathcal{D}, \mathcal{G}) = -nd \ln \frac{\sum_{i=1}^{d} \text{MSE}_i}{d} - |\mathcal{G}| \ln n. \tag{14}$$

B.3 BIC for Binary Data with Logistic Regression

From the formulation in Eq. equation 6, we can also adapt it to non-continuous data. For example, let us consider the logistic causal model governed by

$$x_i \sim \text{Bernoulli} \left(f_i \left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}} \right) \right). \tag{15}$$

where f_i models the conditional probability of x_i given $\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}$.

The log-likelihood of data under this model is determined by

$$\mathcal{L} = \ln p \left(\mathcal{D} \mid \mathcal{G}, \{f_i\}_{i=1}^{d} \right) = \sum_{i=1}^{d} \sum_{j=1}^{n} \left(x_i^{(j)} \ln f_i \left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)} \right) + \left(1 - x_i^{(j)} \right) \ln \left(1 - f_i \left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)} \right) \right) \right).$$

Maximizing this objective and plugging it into Eq. equation 6 gives us with the BIC for logistic model:

$$S_{\text{BIC-Logistic}}(\mathcal{D}, \mathcal{G}) = 2\hat{\mathcal{L}} - |\mathcal{G}| \ln n. \tag{16}$$

The first term is equivalent to the logistic loss used in, e.g., DAGMA (Bello et al., 2022), while the second term punishes redundant edges as always.

C Additional Discussions and Details

C.1 Preliminary Candidate Generation

In Sec. 4.2, we generate a set of C preliminary candidates in a hyperrectangle centered at the best solution so far $\mathbf{z}^*$. This hyperrectangle can be seen as a single "trust region" in BO (Eriksson et al., 2019; Daulton et al., 2022). More specifically, our trust region is the intersection of the search space $[-1, 1]^{d(1+k)}$ and the hypercube of length L centered at $\mathbf{z}^*$. Following Eriksson et al. (2019), we adaptively update L according to the learning progress. This is done by maintaining a success (and failure) counter that keeps track the number of consecutive BO iterations that improves (or fails to improve, resp.) the DAG score. After n_{succ} consecutive successes, we enlarge L by two times in order to shift the focus to other regions, and after n_{fail} consecutive failures, we shrink it by two times to zoom more into the current region. In all experiments, we use the fixed values of $n_{\text{succ}} = 3$, $n_{\text{fail}} = 5$, and L is initialized with the value of 1 and is clipped to be within $[0.01, 2]$ after each update.

To produce the preliminary candidates on which we generate Thompson samples, for the very first suggestions, following Eriksson et al. (2019), we employ the Latin hypercube design (LHD, McKay et al., 2000), which is a space-filling method used to generate near-random samples from a multi-dimensional space, which ensures that each dimension of the hypercube is evenly covered, while random sampling can lead to an unevenly covered space. For subsequent suggestions, we follow the established procedure in (Eriksson et al., 2019) to first generate a scrambled Sobol sequence (Owen, 1998) within the current trust region, then we use the perturbation value in the Sobol sequence with probability $\min\left\{1, \frac{20}{d(1+k)}\right\}$ for each given candidate and dimension, and the value of the center $\mathbf{z}^*$ otherwise. As noted in (Regis & Shoemaker, 2013; Eriksson et al., 2019), perturbing only a few dimensions can lead to a significant performance improvement for high-dim scenarios.

C.2 Scalable Surrogate Modeling

The literature of BO is vast and here we only discuss a few promising alternative approaches to scale up surrogate models in BO, and justify of our dropout neural network choice.

Bayesian neural networks (BNN) are a also natural replacement thanks to the flexibility of neural networks combined with the inherent ability to model uncertainty of the Bayesian ideology (Springenberg et al., 2016). However, to stay as close as possible to a truly Bayesian treatment, i.e., characterizing the exact posterior distributions, they require stochastic gradient Markov Chain Monte Carlo (MCMC) to sample from the posterior, which necessitates several sampling steps to reach the desirable posterior (Springenberg et al., 2016). Meanwhile, in our method, we sacrifice the accuracy of the posterior inference, so that we can sample from the posterior faster with a single forward pass through the dropout neural networks. That being said, our DAG learning accuracy is still strong, which justifies this sacrifice.

Sparse GPs (or Approximate GPs) have also been studied to enhance scalability of GPs (Snelson & Ghahramani, 2005; Hensman et al., 2015; Titsias, 2009). However, they do not scale well with dimensionality (Wang et al., 2018).

Random forest (RF) has also been considered as an alternative surrogate model in BO (Hutter et al., 2011). An RF is composed of multiple decision trees, each of which is constructed from a portion of the training set and a few random dimensions. Therefore, uncertainty modeling in RF is achieved via the variation of the individual trees' predictions. While RF is known to be efficient, especially in tabular data, we find it more straightforward to train neural networks continually (Wang et al., 2024) compared with tree-based models (Utgoff, 1989; Utgoff et al., 1997). This is crucial for our BO framework to scale with the number of iterations as aforementioned.

Ensembling methods (Wang et al., 2018; Guo et al., 2018) sidestep the scalability challenge of BO with the use of an ensemble of models trained on different partitions of the samples and dimensions. A similar idea is also proposed in (Eriksson et al., 2019), where local GPs are trained on multiple local trust regions of the search space. However, these methods require training multiple models, and managing them is less straightforward than maintaining a single neural network like our method, which also scales very well with a strong causal discovery performance.

C.3 Pruning Techniques

Pruning is typically employed in causal discovery to reduce false positive estimates, and there are several approaches depending on the causal model as follows.

Linear data. For linear data, given the resultant DAG $\mathcal{G}$ that needs to be pruned, linear regression is used to find the linear coefficients $\{\hat{w}_{ij}\}_{(i\to j)\in\mathcal{G}}$ associating with the edges in $\mathcal{G}$. Then, only the weights satisfying a certain absolute strength α is kept, while the remaining are removed. In other words, the edge $(i \to j) \in \mathcal{G}$ is kept iff $|\hat{w}_{ij}| > \alpha$. Typically, the weights in the true generative process for linear models are sampled from $\mathcal{U}([-2,-0.5]\cup[0.5,2])$, and the common value for α is 0.3 (Zheng et al., 2018; Zhu et al., 2020; Wang et al., 2021; Bello et al., 2022; Massidda et al., 2024; Duong et al., 2024).

Nonlinear additive model. For nonlinear data under additive models, a common approach is based on feature selection using generalized additive model (GAM) regression (Bühlmann et al., 2014), also known as CAM pruning. Particularly, each node i is regressed on its parents $\mathbf{x}_{\mathrm{pa}_i^{\mathcal{G}}}$ using GAM, then the significance test of covariates is conducted, and a parent is kept if its p-value is lower than the significance level of 0.001. This is also done in (Zhu et al., 2020; Wang et al., 2021; Massidda et al., 2024; Rolland et al., 2022; Sanchez et al., 2023; Duong et al., 2024).

Non-additive models. For general non-additive models, conditional independence (CI) testing can also be employed to prune edges. To be more specific, the Faithfulness assumption (Peters et al., 2017) implies that any conditional independence observed in data reflects the corresponding d-separation in the causal graph, so if $x_i \perp\!\!\!\perp x_j \mid \mathbf{x}_{\mathrm{pa}_i\setminus\{j\}}$ for some $j \in \mathrm{pa}_i$, then j is an extra edge and needs to be removed. This follows the same idea of constraint-based causal discovery (Spirtes et al., 2000; Colombo et al., 2012) and feature selection via Markov Blankets (Koller & Sahami, 1996; Xing et al., 2001). As observed in (Duong et al., 2024), CI-based pruning leads to better performance on the Sachs dataset compared with CAM pruning, and hence we also employ it for all methods on the Sachs dataset. The specific CI test is the popular kernel-based method KCIT (Zhang et al., 2011) with a significance level of 0.001.

D Experiment Details

D.1 DAG Learning Metrics

We evaluate the performance of each method using the following standard measures, each of which calculates the disparity between an estimated DAG and the ground truth DAG:

- **Structural Hamming Distance (SHD, lower is better):** this is the most common metric in causal discovery, which counts the minimum number of edge additions, removals, and reversals to turn the estimated DAG into the true graph.
- **BIC score (higher is better):** we also monitor the BIC score of the estimated DAG for all methods in the main text (despite the fact that some baselines do not optimize this score). For linear data, we use the BIC with equal variances and linear regression, while for nonlinear data, BIC with non-equal variance with Gaussian process regression is used, and BIC with logistic regression is employed for binary data.
- **True Positive Rate (TPR, higher is better):** this measures the ratio of correctly recovered edges over the true edges in the ground truth DAG.
- **False Discovery Rate (FDR, lower is better):** this measures the proportion of incorrectly estimated edges over all estimated edges.
- **Precision, Recall, and F_1:** this measures the binary classification performance by treating the binary adjacency matrix as a set of individual binary classification tasks.

D.2 Synthetic Causal Graphs

Our synthetic causal graphs involve one of the two well-known graph models:

- **Erdős-Rényi** (ER, Erdős & Rényi, 1960): the edges in this type of graph are added independently with fixed probability. To generate a DAG of d nodes with an expected in-degree

of e, we first generate an undirected graph where edges are added with probability $\frac{4de}{d(d-1)}$, then orient the edges using a random permutation over the list of nodes.

- **Scale-Free** (SF, Barabási & Albert, 1999): these are graphs where the degree distribution follows a power law, where a few nodes have many connections while others have only a few connections. To generate SF graphs with $\approx de$ edges, we start with an empty graph then repeatedly grow it by attaching new nodes, each with k edges, that are preferentially attached to existing nodes.

Remark 1. It is noteworthy that the majority of methods perform well only for graphs with up to $e = 4$ (Zheng et al., 2018; Zhu et al., 2020; Ng et al., 2020; Yu et al., 2021; Wang et al., 2021; Bello et al., 2022; Yang et al., 2023b). Indeed, it was noted in (Bello et al., 2022) that ER4 and SF4 are the hardest settings. However, in this study, we have shown that our **DrBO** method is still very accurate on much denser ER8 and SF8 graphs.

D.3 BnLearn Datasets

The BnLearn repository[4] (Scutari, 2010) contains a set of Bayesian networks of varying sizes and complexities from different real-world domains. The chosen networks in our study include Alarm (Beinlich et al., 1989), Asia (Lauritzen & Spiegelhalter, 1988), Cancer (Korb & Nicholson, 2010), Child (Spiegelhalter, 1992), and Earthquake (Korb & Nicholson, 2010). However, these datasets only describe the conditional probability distributions of discrete variables, while the baselines considered in our method are mostly implemented for continuous data. Since the purpose of this experiment is to show that *our method can correctly recover real structures*, we use the networks from the BnLearn to generate continuous data. Specifically, we employ linear-Gaussian SCMs as in Sec. 5.1.1 to generate synthetic data adhering to real causal networks, and each dataset contains only 1,000 observational samples.

D.4 Implementations and Platform

Implementations. We employ the following implementations for the considered baselines as follows:

- DAGMA (Bello et al., 2022): we use the official implementation provided by the authors at `https://github.com/kevinsbello/dagma`.
- COSMO (Massidda et al., 2024): we use the original implementation attached as supplementary material at `https://openreview.net/forum?id=KWO8LSUC5W`.
- CORL (Wang et al., 2021): we employ the official implementation provided in the gCastle library (Zhang et al., 2021) at `https://github.com/huawei-noah/trustworthyAI`.
- ALIAS (Duong et al., 2024): we reimplement their method by following the exact instructions and libraries described.
- GOLEM (Ng et al., 2020): we adopt the implementation provided by gCastle (Zhang et al., 2021) at `https://github.com/huawei-noah/trustworthyAI`.
- NOTEARS+TMPI (Zheng et al., 2018; Zhang et al., 2022): we replace the DAG constraint in NOTEARS' implementation at `https://github.com/xunzheng/notears` with the TMPI constraint at `https://github.com/zzhang1987/Truncated-Matrix-Power-Iteration-for-Differentiable-DAG-Learning`.

Platform. The majority of our experiments are conducted on a 24-core Intel processor with 4.9 GHz frequency, and an NVIDIA GPU with 16G of CUDA memory and compute capability of 8.9. The only exception is the case of CORL on large graphs (Figure 1(b)) which requires more than 16Gb of CUDA memory, and therefore these experiments are conducted on an NVIDIA A100 GPU with 32G of CUDA memory instead.

[4]Data is publicly downloadable at `https://www.bnlearn.com/bnrepository`

E HYPERPARAMETERS

We provide the specific set of hyperparameters for our **DrBO** method in Table 2.

Table 2: Hyperparameters for **DrBO**. Unless specifically indicated, the default hyperparameters here are used for all experiments.

Hyperparameter	Experiment		
	Linear data	**Nonlinear data with GPs**	**Sachs data**
Normalize data	No	No	Yes
Scoring function	$S_{\text{BIC-EV}}$ with linear regression	$S_{\text{BIC-NV}}$ with GP regression	$S_{\text{BIC-NV}}$ with GP regression
Pruning method	Linear pruning	No pruning	CIT pruning
Batch size B		64	
DAG rank k		8	
No. training steps n_{grads}		10	
No. preliminary candidates C		100,000	
Optimizer		Adam	
Learning rate		0.1	
Replay buffer size n_{replay}		1,024	
No. hidden units		64	
Dropout rate		0.1	

For ALIAS and CORL, apart from the recommended default hyperparameters, for fairness, we also use the same batch size of 64 as ours (except for CORL which requires the batch size to be at least the number of nodes, so for 100-node graphs we have to set the batch size to 100), as well as the same scoring function and pruning method as ours in each experiment. Regarding COSMO and DAGMA, we use the linear and nonlinear versions specific to each experiment and the hyperparameters are set as recommended. We also use the same BIC score and pruner specific to each experiment to track the progress of all methods in Figure 1.

Regarding the number of evaluations, for all methods, we run more than needed then cut off at common thresholds.

Remark 2. Following the same line as in (Bello et al., 2022; Zheng et al., 2018) and many studies in this field, we intentionally avoid hyperparameter tuning. This is to prevent injecting spurious information of the dataset characteristics into the causal discovery results, which is against the idea of causality. Therefore, we use a fixed set of hyperparameters for each method in each setting. The hyperparameters of baseline methods are chosen as recommended in the original manuscripts, while ours are chosen based on common practice and prior experience. Indeed, our ablation studies reveal that there are better hyperparameter choices that can further improve the performance of our method.

F ADDITIONAL CAUSAL DISCOVERY SETTINGS

F.1 DIFFERENT SAMPLE SIZES

In Figure 4, we test our method with varying sample sizes on linear-Gaussian data. The results indicate that our method is already very accurate with SHD ≈ 0 at merely 500 samples.

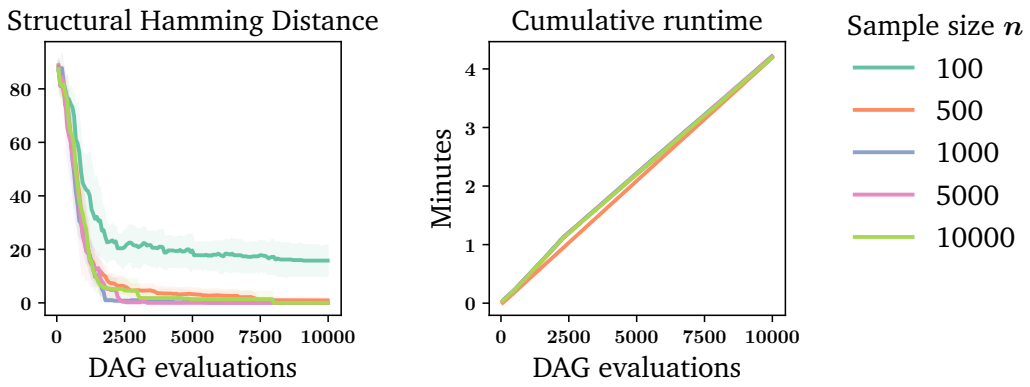


Figure 4: **Causal Discovery Performance with Varying Sample Sizes.** We apply our **DrBO** method on linear-Gaussian data with 20ER4 graphs. Shaded areas represent 95% confidence interval over 5 runs.

F.1.1 Different Graph Types

We evaluate our method on different graph types, namely ER and SF, with varying densities, as shown in Table 3. Our method perfectly identifies the correct DAG in all cases for both graph models, even in the dense graphs ER8 and SF8.

Table 3: **Causal Discovery Performance with Different Graph Types.** The considered graph models include Erdős-Rényi (Erdős & Rényi, 1960) and Scale-Free (SF, Barabási & Albert, 1999), with expected in-degrees of 2 and 8 corresponding to the case of sparse and dense structures, respectively. We compare our **DrBO** method with ALIAS (Duong et al., 2024) and DAGMA (Bello et al., 2022) on linear-Gaussian datasets with 20 variables and 10,000 samples. The numbers are mean $\pm$ std over 5 random datasets. For fairness, all methods are limited to 10,000 score evaluations.

Graph	Expected In-degree	Method	SHD ↓	FDR ↓	TPR ↑
ER	2	ALIAS	21.6 ± 7.2	0.32 ± 0.12	0.68 ± 0.08
		DAGMA	25.8 ± 7.3	0.28 ± 0.13	0.55 ± 0.07
		DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
	8	ALIAS	84.2 ± 7.5	0.15 ± 0.03	0.53 ± 0.04
		DAGMA	122.4 ± 4.7	0.19 ± 0.03	0.28 ± 0.02
		DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
SF	2	ALIAS	13.6 ± 2.7	0.18 ± 0.09	0.71 ± 0.02
		DAGMA	23.4 ± 9.8	0.25 ± 0.19	0.53 ± 0.13
		DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
	8	ALIAS	68.8 ± 9.0	0.34 ± 0.08	0.52 ± 0.06
		DAGMA	94.4 ± 8.1	0.49 ± 0.11	0.2 ± 0.06
		DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$

F.1.2 Different Noise Distributions

To examine the performance of our method compared with others under noise misspecification, in Table 4, we evaluate causal discovery performance various noise distributions. Specifically, we consider linear SCM $x_i := \sum_{j \in \mathrm{pa}_i} w_{ji} x_j + \varepsilon_i$, where ε_i is drawn from one of the following distributions

- **Exponential noise:** $\varepsilon_i \sim \mathrm{Exp}(1), \forall i = 1, \ldots, d$, where 1 is the scale parameter.
- **Gaussian noise:** $\varepsilon_i \sim \mathcal{N}(0, 1), \forall i = 1, \ldots, d$, where 0 is the mean and 1 is the variance.

- **Gumbel noise:** $\varepsilon_i \sim \text{Gumbel}(0,1), \forall i = 1, \ldots, d$, where 0 is the location parameter and 1 is the scale parameter.
- **Laplace noise:** $\varepsilon_i \sim \text{Laplace}(0,1), \forall i = 1, \ldots, d$, where 0 is the location parameter and 1 is the scale parameter.
- **Uniform noise:** $\varepsilon_i \sim \mathcal{U}(-1,1), \forall i = 1, \ldots, d$, where the minimum value is -1 and maximum value is 1.

Remark 3. We note that our BIC score is kept unchanged for different noises to show that our method can work well beyond the assumed Gaussian noise.

Table 4: **Causal Discovery Performance with Different Noise Distributions.** We consider 5 noise distributions: Exponential, Gaussian, Gumbel, Laplace, and Uniform. Our **DrBO** method is compared with ALIAS (Duong et al., 2024) and DAGMA (Bello et al., 2022) on linear-Gaussian datasets with 20 variables and 10,000 samples. The numbers are mean ± std over 5 random datasets. For fairness, all methods are limited to 20,000 score evaluations.

Noise Type	Method	SHD ↓	FDR ↓	TPR ↑
Exponential	ALIAS	36.2 ± 13.3	0.24 ± 0.07	0.75 ± 0.09
	DAGMA	56.6 ± 6.8	0.22 ± 0.10	0.37 ± 0.05
	DrBO (ours)	$\mathbf{0.4 \pm 0.6}$	$\mathbf{0.00 \pm 0.01}$	$\mathbf{0.99 \pm 0.01}$
Gaussian	ALIAS	34.2 ± 11.4	0.23 ± 0.08	0.77 ± 0.05
	DAGMA	57.2 ± 7.6	0.25 ± 0.09	0.39 ± 0.09
	DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
Gumbel	ALIAS	36.6 ± 12.9	0.26 ± 0.08	0.77 ± 0.06
	DAGMA	54.8 ± 7.4	0.25 ± 0.04	0.44 ± 0.08
	DrBO (ours)	$\mathbf{0.2 \pm 0.5}$	$\mathbf{0.00 \pm 0.01}$	$\mathbf{1.00 \pm 0.01}$
Laplace	ALIAS	36.4 ± 14.2	0.26 ± 0.07	0.76 ± 0.09
	DAGMA	56.0 ± 8.8	0.25 ± 0.08	0.40 ± 0.09
	DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$
Uniform	ALIAS	34.4 ± 13.0	0.24 ± 0.08	0.77 ± 0.05
	DAGMA	59.0 ± 7.3	0.26 ± 0.08	0.39 ± 0.09
	DrBO (ours)	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$

F.1.3 BGe Score for Markov Equivalence Class Discovery

Here, we show that our method is not restricted to the BIC score and can be extended to other scores as well. We consider the Bayesian Gaussian equivalent (BGe, Geiger & Heckerman, 1994; Heckerman et al., 1995) score for learning the Markov Equivalence Class (MEC) of DAGs in linear-Gaussian settings. The BGe score assigns equal scores to DAGs belonging to the same MEC, and can be decomposed as the sum of local scores as follows:

$$S_{\text{BGe}}(\mathcal{D}, \mathcal{G}) = \sum_{i}^{d} \text{LocalBGe}_i\left(\text{pa}_i^{\mathcal{G}}\right). \tag{17}$$

We refer readers to, for example, Kuipers et al. (2014), for the specific formula of the BGe score. To adapt our method with this score, we simply train each dropout network DropoutNN_i to predict LocalBGe_i from $\text{pa}_i^{\mathcal{G}}$ and combine them using Eq. equation 17. It can be seen that this adaptation does not involve changing any other component of our method.

We report the results in Figure 5, where we compare **DrBO** with two popular baselines that are well-known to recover the MEC, namely PC (Spirtes et al., 2000) and GES (Chickering, 2002).[5] The results illustrate that our method can find the DAGs with the highest BGe scores, while the scores from PC' and GES's estimations are well below those of the ground truths. As a result, our recovered structures are more accurate than the baselines.

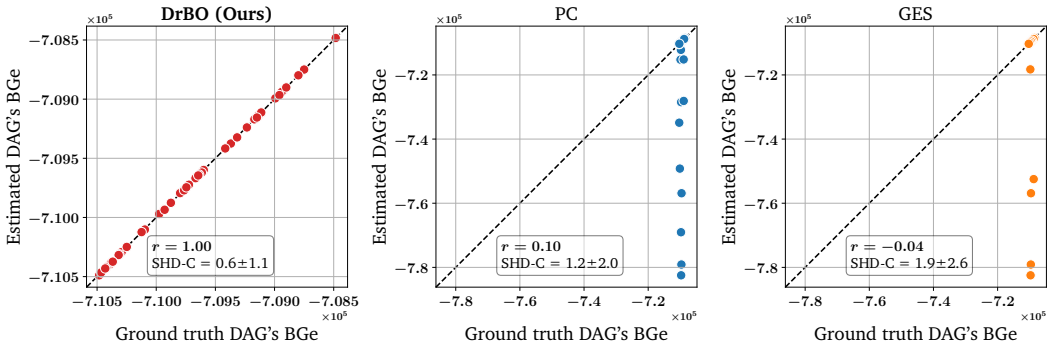


Figure 5: **BGe for Markov Equivalence Class Discovery.** We compare the BGe score of ground truth DAGs and the estimations from **DrBO** with two popular baselines PC (Spirtes et al., 2000) and GES (Chickering, 2002). Each point corresponds to one of 50 random datasets with linear Gaussian data on ER graphs of 5 nodes and 5 edges on average. The Pearson correlation coefficient r between the scores of the estimated and ground truth DAGs are included. In addition, we also report the SHD-C metric, which measures the structural distance between MECs.

F.1.4 Discrete Data

In this section, we show that our method is not limited to continuous data either. To demonstrate, we consider binary data with logistic causal models:

$$x_i \sim \text{Bernoulli}\left(f_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}\right)\right).$$

We adapt our method to this situation by simply changing the BIC score to take into account logistic models. More particularly, we use the BIC score for logistic data as in Eq. equation 16, where the maximum log-likelihood can be decomposed into

$$\hat{\mathcal{L}} = \sum_{i=1}^{d} \text{LocalMLL}_i,$$

where $\text{LocalMLL}_i = \sum_{j=1}^{n}\left(x_i^{(j)} \ln \hat{f}_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right) + \left(1 - x_i^{(j)}\right) \ln\left(1 - \hat{f}_i\left(\mathbf{x}_{\text{pa}_i^{\mathcal{G}}}^{(j)}\right)\right)\right)$, with $\hat{f}_i$ being the maximum-likelihood estimator found via logistic regression.

Then, we employ our local surrogate models DropoutNN_i to model LocalMLL_i, and the acquisition function values are obtained by summing the local score samples. Again, this adaptation only involves changing the scoring function and does not modify any other component of our BO framework.

We report the causal discovery performance under binary data in Figure 6, where we also compare **DrBO** with the state-of-the-art DAGMA, which is now set to use the logistic loss supported. It can be seen that our method also performs well for binary data, where it consistently obtains the highest scoring graphs, resulting in a low structural error. Meanwhile, DAGMA usually finds only sub-optimal solutions which lead to high structural errors.

[5] We use the gCastle library (Zhang et al., 2021) for their implementations, where hyperparameters are left as default.

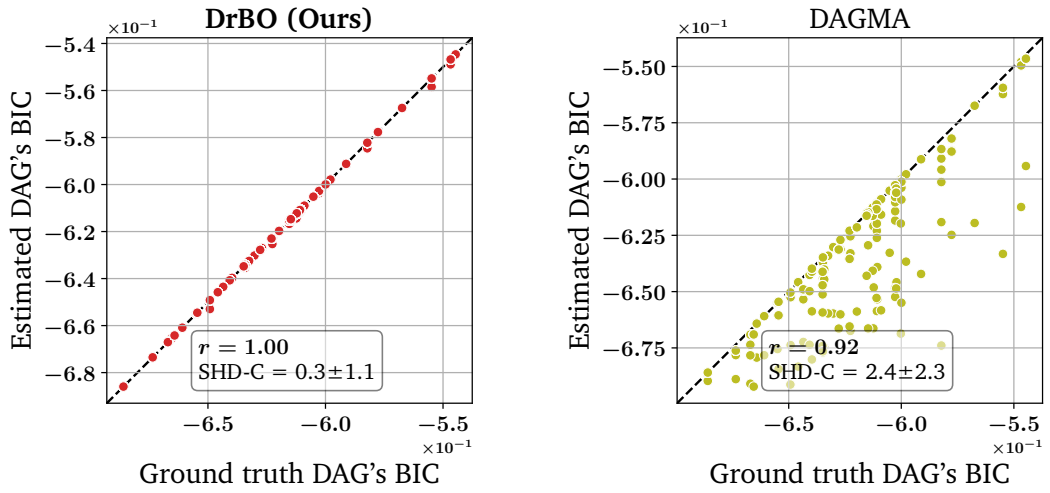


Figure 6: **Causal Discovery performance on Binary Data.** We compare our **DrBO** method using the BIC score for the logistic model with DAGMA (Bello et al., 2022) using the logistic loss. Each point corresponds to one of 50 random datasets with logistic data on ER graphs of 5 nodes and 5 edges on average. The Pearson correlation coefficient r between the scores of the estimated and ground truth DAGs are included. In addition, we also report the SHD-C metric, which measures the structural distance between MECs.

F.2 Ablation Experiments

F.2.1 Effect of Dropout Networks

In Figure 3(b), we compare our dropout networks as the surrogate model with exact GPs and approximate GPs. Approximate GPs learn a set of pseudo data points called inducing points and conduct inference via these points instead of the real data (Hensman et al., 2015). Here we use a small number of 100 inducing points for Approximate GPs, to see they can scale well with few inducing points. Due to the limited scalability and intensive memory requirement for GPs, we can only use $C = 10{,}000$ preliminary candidates on which we sample from the posteriors, while our default hyperparameter is $C = 100{,}000$ using dropout networks.

F.2.2 Effect of Continual Training

In Figure 3(d), we compare the continual training approach with fully retraining using all data. For continual learning, we use default hyperparameters $n_{\text{replay}} = 1{,}024$, $B = 64$, and $n_{\text{grads}} = 10$, meaning for each BO iteration, we perform 10 gradients update, each update is calculated from $1{,}024 + 64 = 1{,}088$ datapoints. To ensure fairness, we also use $n_{\text{grads}} = 10$ epochs and a mini-batch size of $1{,}088$ for the full retraining approach.

F.2.3 Effect of Evaluation Batch Size (B)

In Figure 7, we show the influence of the evaluation batch size B onto the performance and scalability of our method. Overall, it is clear that smaller batch sizes lead to better SHD but much worse runtime since the surrogate model is updated more frequently, and vice versa. However, $B = 64$ seems to achieve balance, where it enables SHD ≈ 0 and lower runtime than smaller batch sizes.

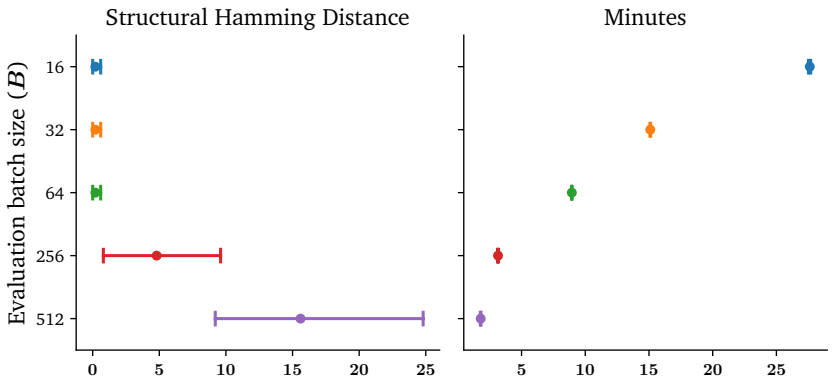


Figure 7: **Effect of Evaluation Batch Size B.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.4 Effect of Number of Preliminary Candidates (C)

We show the variation our **DrBO**'s performance w.r.t. different numbers of preliminary candidates C in Figure 8, showing that the best performance and runtime can be achieved at 10,000 candidates, and even with 10x more candidates, the runtime of our method increases only slightly.

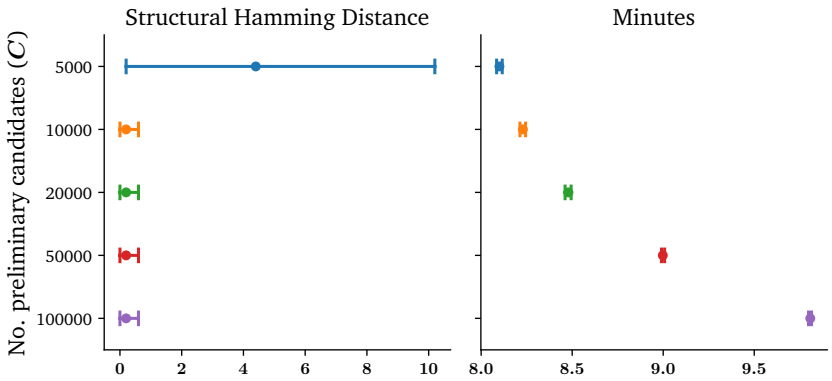


Figure 8: **Effect of Number of Preliminary Candidates C.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.5 Effect of Number of Training Steps per BO Iteration (n_{GRADS})

We study the effect of the number of gradient steps in each BO iteration (n_{grads}) in Figure 9. In general, small values may lead to underfitting and large values may be prone to overfitting, so values in the middle are better for this hyperparameter.

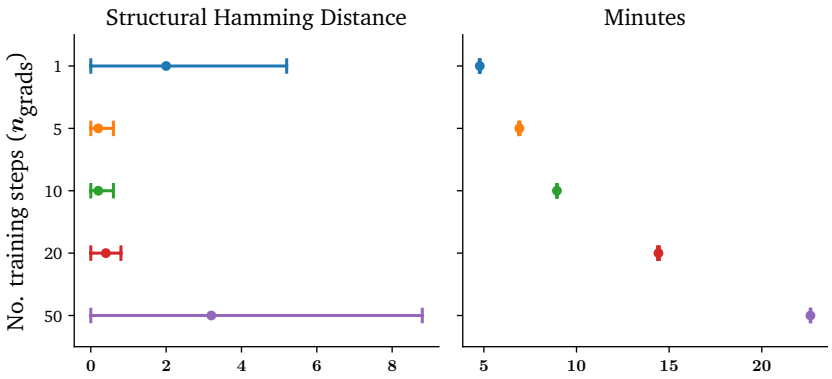


Figure 9: **Effect of Number of Training Steps per BO Iteration n_{grads}.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.6 Effect of Replay Buffer Size (n_{REPLAY})

In Figure 10, we show that higher values for the replay buffer size significantly reduces SHD but does not considerately influence the runtime.

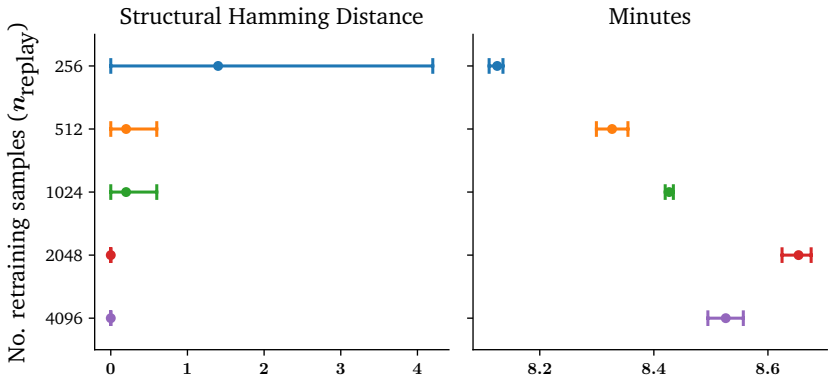


Figure 10: **Effect of Replay Buffer Size n_{replay}.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.7 Effect of Learning Rate

Figure 11 depicts that the learning rate has a weak effect on the performance and scalability of our method, where any value below 1 can achieve the same level of SHD and runtime. The SHD only becomes large for a high learning rate of 1.

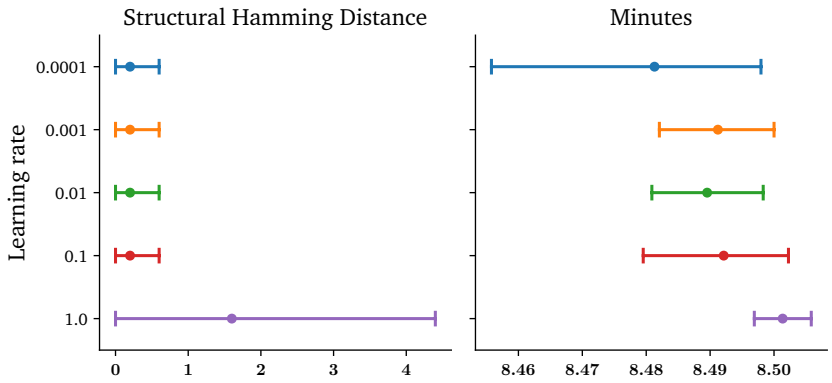


Figure 11: **Effect of Learning Rate.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.8 Effect of Number of Hidden Units

We present in Figure 12 that the number of hidden units in our dropout networks also has a visible effect on our method's performance, but not much on the runtime. Specifically, a moderate value of 32 achieves a vanishing SHD with the equivalent runtime as others. Meanwhile, to many hidden units may challenge the training process so performance may drop.

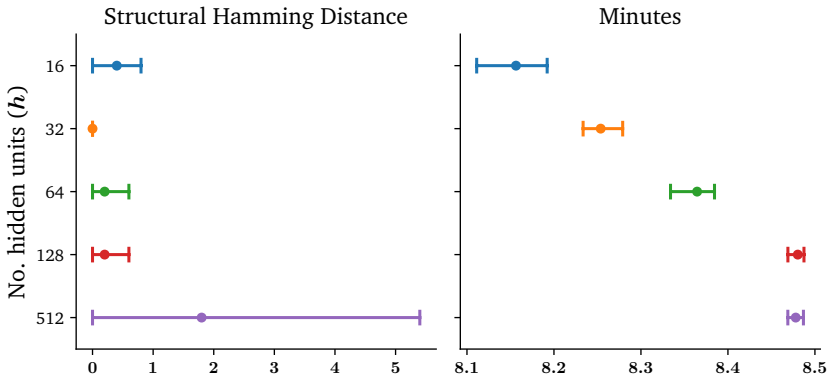


Figure 12: **Effect of Number of Hidden Units h.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

F.2.9 Effect of Number of Dropout Rate

Figure 13 suggests that the performance of our method improves with higher dropout rates, while the runtime does not vary significantly.

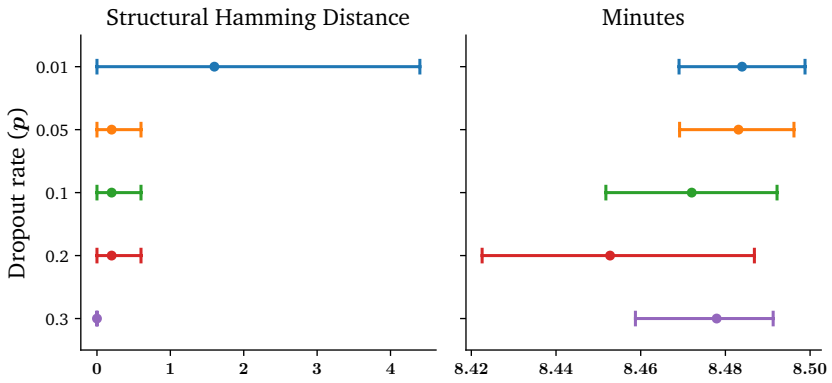


Figure 13: **Effect of Dropout Rate p.** We evaluate our method on linear-Gaussian data with 20ER4 graphs and 1,000 observations. Error bars indicate 95% confidence intervals over 5 runs. The number of evaluations is limited to 20,000.

G Comparison with Ordering-based Methods

Apart from the score-based competitors compared so far, ordering-based methods have also been gaining popularity (Reisach et al., 2021; Rolland et al., 2022; Sanchez et al., 2023; Montagna et al., 2023). These methods sidestep the acyclicity issue by first learning a topological ordering of the causal DAG, then pruning the fully-connected DAG induced by the ordering to obtain the causal structure.

In Table 5, we provide a comprehensive comparison between our method and popular ordering-based, including sortnregress (Reisach et al., 2021), SCORE (Rolland et al., 2022), and DiffAN (Sanchez et al., 2023), on linear, nonlinear, as well as real data. The empirical evaluations indicate that our method is also able to surpass these methods in all metrics and settings.

Table 5: **Comparison with Ordering-based Methods.** We compare our method with popular ordering-based algorithms sortnregress (Reisach et al., 2021), SCORE (Rolland et al., 2022), and DiffAN (Sanchez et al., 2023). The orderings produced by those methods are first transformed into respective fully connected DAGs. Then, for fairness, all DAGs, including ours, are pruned using the weight matrix thresholded at 0.3 for linear data, CAM pruning (Bühlmann et al., 2014) for nonlinear data, and KCIT (Zhang et al., 2011) for the Sachs dataset. The figures are mean$\pm$std over 5 datasets, except for the Sachs dataset. Note that **DrBO**'s result in Figure 1(c) with SHD $\approx$ 0.4 is obtained without pruning, while using CAM pruning leads to a few missing edges as shown below.

Dataset	**Method**	**SHD** $\downarrow$	**FDR** $\downarrow$	**TPR** $\uparrow$	F_1 $\uparrow$
Linear Data (30ER8 graphs, 1,000 samples)	sortnregress	101.4 ± 21.8	0.24 ± 0.0	0.80 ± 0.1	0.78 ± 0.0
	SCORE	247.0 ± 11.8	0.76 ± 0.1	0.05 ± 0.0	0.08 ± 0.1
	DiffAN	236.0 ± 3.7	0.58 ± 0.1	0.16 ± 0.1	0.23 ± 0.1
	DrBO (ours)	$\mathbf{1.6 \pm 1.5}$	$\mathbf{0.00 \pm 0.0}$	$\mathbf{0.99 \pm 0.0}$	$\mathbf{1.0 \pm 0.0}$
Nonlinear Data (10ER4 graphs, 1,000 samples)	sortnregress	27.4 ± 3.3	0.42 ± 0.1	0.37 ± 0.1	0.45 ± 0.1
	SCORE	9.4 ± 4.6	0.11 ± 0.1	0.83 ± 0.1	0.86 ± 0.1
	DiffAN	18.6 ± 3.8	0.34 ± 0.1	0.61 ± 0.1	0.63 ± 0.1
	DrBO (ours)	$\mathbf{4.2 \pm 1.3}$	$\mathbf{0.00 \pm 0.0}$	$\mathbf{0.90 \pm 0.0}$	$\mathbf{0.95 \pm 0.0}$
Sachs et al. (2005), (11 nodes, 17 edges, 853 samples)	sortnregress	13	0.44	0.29	0.38
	SCORE	12	0.33	0.35	0.46
	DiffAN	16	0.75	0.12	0.16
	DrBO (ours)	**9**	**0.11**	**0.47**	**0.62**

H TIME COMPARISONS

Table 6: Runtime comparison on 30ER8 linear-Gaussian data (experiment in Figure 1(a)).

Method	Max number of evaluations					
	10,000		20,000		50,000	
	SHD	**Runtime (mins)**	**SHD**	**Runtime (mins)**	**SHD**	**Runtime (mins)**
ALIAS	208.8 ± 13.1	0.5 ± 0.0	165.0 ± 18.0	1.1 ± 0.0	105.2 ± 2.9	2.4 ± 0.0
CORL	169.0 ± 17.1	1.1 ± 0.0	162.0 ± 13.4	2.1 ± 0.0	145.4 ± 29.3	5.2 ± 0.1
COSMO	213.6 ± 17.4	0.6 ± 0.0	203.4 ± 21.0	1.1 ± 0.0	195.8 ± 10.8	2.8 ± 0.0
DAGMA	222.2 ± 11.2	$\mathbf{0.0 \pm 0.0}$	218.8 ± 10.6	$\mathbf{0.0 \pm 0.0}$	181.6 ± 22.4	$\mathbf{0.0 \pm 0.0}$
DrBO (ours)	$\mathbf{2.0 \pm 1.2}$	4.6 ± 0.0	$\mathbf{2.0 \pm 1.4}$	9.2 ± 0.0	$\mathbf{1.6 \pm 1.5}$	22.9 ± 0.1

Table 7: Runtime comparison on 100ER2 linear-Gaussian data (experiment in Figure 1(b)).

Method	Max number of evaluations					
	10,000		20,000		50,000	
	SHD	**Runtime (mins)**	**SHD**	**Runtime (mins)**	**SHD**	**Runtime (mins)**
ALIAS	230.4 ± 32.7	2.7 ± 0.3	136.8 ± 30.2	5.5 ± 0.3	32.0 ± 16.7	13.5 ± 0.6
CORL	148.0 ± 42.9	11.1 ± 0.1	120.2 ± 18.1	22.0 ± 0.2	81.0 ± 19.9	54.2 ± 0.5
COSMO	111.0 ± 10.9	0.8 ± 0.0	111.8 ± 12.2	1.5 ± 0.0	112.6 ± 13.4	3.7 ± 0.0
DAGMA	124.8 ± 17.4	0.0 ± 0.0	93.6 ± 16.1	0.1 ± 0.0	6.6 ± 4.0	0.3 ± 0.0
DrBO (ours)	$\mathbf{29.2 \pm 16.7}$	12.7 ± 0.1	$\mathbf{3.4 \pm 4.3}$	25.4 ± 0.3	$\mathbf{1.4 \pm 1.1}$	62.7 ± 0.8

Table 8: Runtime comparison on 10ER4 nonlinear data with Gaussian processes (experiment in Figure 1(c)).

Method	**Max number of evaluations**					
	1,000		2,000		20,000	
	SHD	**Runtime (mins)**	**SHD**	**Runtime (mins)**	**SHD**	**Runtime (mins)**
ALIAS	15.8 ± 4.4	4.0 ± 0.2	12.6 ± 4.2	5.8 ± 0.2	4.0 ± 2.3	8.0 ± 0.3
CORL	10.4 ± 3.4	20.9 ± 3.0	9.2 ± 2.3	26.5 ± 3.1	8.4 ± 3.0	29.9 ± 2.4
COSMO	34.8 ± 2.6	0.3 ± 0.1	33.0 ± 3.3	0.4 ± 0.1	25.6 ± 4.1	1.8 ± 0.1
DAGMA	40.4 ± 2.3	0.0 ± 0.0	38.4 ± 2.7	0.1 ± 0.0	34.2 ± 3.5	0.8 ± 0.1
DrBO (ours)	$\mathbf{2.2 \pm 1.6}$	3.7 ± 0.8	$\mathbf{0.4 \pm 0.5}$	3.9 ± 0.9	$\mathbf{0.4 \pm 0.5}$	6.1 ± 1.0