# DiBS: Differentiable Bayesian Structure Learning

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

Bayesian structure learning allows inferring Bayesian network structure from data while reasoning about the epistemic uncertainty—a key element towards enabling active causal discovery and designing interventions in real world systems. In this work, we propose a general, fully *differentiable* framework for *Bayesian structure learning (DiBS)* that operates in the continuous space of a latent probabilistic graph representation. Building on recent advances in variational inference, we use DiBS to devise an efficient method for approximating posteriors over structural models. Contrary to existing work, DiBS is agnostic to the form of the local conditional distributions and allows for joint posterior inference of both the graph structure and the conditional distribution parameters. This makes our method directly applicable to posterior inference of nonstandard Bayesian network models, e.g., with nonlinear dependencies encoded by neural networks. In evaluations on simulated and real-world data, DiBS significantly outperforms related approaches to joint posterior inference.

## 1 Introduction

Discovering the statistical and causal dependencies that underlie the variables of a data-generating system is of central scientific interest. Bayesian networks (BNs) [1] and structural equation models are commonly used for this purpose [2–7]. Structure learning, the task of learning a BN from observations of its variables, is well-studied, but computationally very challenging due to the combinatorially large number of candidate graphs and the constraint of graph acyclicity.

While structure learning methods arrive at a single plausible graph or its Markov equivalence class (MEC), e.g., [8–11], *Bayesian structure learning* aims to infer a full posterior distribution over plausible BNs given the observations. A distribution over structures allows quantifying the epistemic uncertainty and the degree of confidence in any given solution, e.g., when the amount of data is small. Most importantly, downstream tasks such as experimental design and active learning rely on a posterior distribution over BNs to quantify the information gain from specific interventions and to uncover the causal structure using a small number of experiments [12–16].

A key challenge in Bayesian structure learning is working with a posterior over BNs—a distribution over the joint space of (discrete) directed acyclic graphs and (continuous) conditional distribution parameters. Most of the practically viable approaches to Bayesian structure learning revolve around Markov chain Monte Carlo (MCMC) sampling in discrete spaces and bootstrapping of classical score and constraint-based structure learning methods, e.g., in causal discovery [12–16]. However, to remain tractable while marginalizing out the parameters, these methods typically require a closed form for the marginal likelihood of the observations given the graph. This limits inference to simple and by now well-studied linear Gaussian or Categorical BN models [17, 18] and makes it difficult to infer more expressive BNs that, e.g., model nonlinear relationships among the variables. Due to the discrete nature of these approaches, recent advances in approximate inference and gradient-based optimization could not yet be translated into similar performance improvements in Bayesian structure learning.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

In this work, we propose a novel, *fully differentiable framework for Bayesian structure learning* that operates in the continuous space of a latent probabilistic graph representation. Contrary to existing work, our formulation is agnostic to the distributional form of the BN and allows for inference of the joint posterior of both the conditional distribution parameters and the graph structure. This makes our approach directly applicable to nonstandard BN models where neither the marginal likelihood nor the maximum likelihood parameter estimate have a closed form. We instantiate our framework with the particle variational inference method of Liu and Wang [19] and present a general purpose method for approximate Bayesian structure learning. In our experiments on synthetic and real-world data, our method outperforms all alternative approaches to joint posterior inference of graphs and parameters and when modeling nonlinear interactions among the variables, often by a significant margin. This allows us to narrow down plausible causal graphs with greater precision and to make better predictions under interventions—an important stepping stone towards active causal discovery.

## 2 Background

**Bayesian Networks** A Bayesian network $(\mathbf{G}, \mathbf{\Theta})$ models the joint density $p(\mathbf{x})$ of a set of $d$ variables $\mathbf{x} = x_{1:d}$ using 1) a directed acyclic graph (DAG) $\mathbf{G}$ encoding the conditional independencies of $\mathbf{x}$ and 2) parameters $\mathbf{\Theta}$ defining the local conditional distributions of each variable given its parents in the DAG. When modeling $p(\mathbf{x})$ using a BN, each variable is assumed to be independent of its non-descendants given its parents, thus allowing for a compact factorization of the joint $p(\mathbf{x} \mid \mathbf{G}, \mathbf{\Theta})$ into a product of local conditional distributions for each variable and its parents in $\mathbf{G}$.

**Bayesian Inference of BNs** Given independent observations $\mathcal{D} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$, we consider the task of inferring a *full posterior* density over Bayesian networks that model the observations. Following [20], given a prior distribution over DAGs $p(\mathbf{G})$ and a prior over BN parameters $p(\mathbf{\Theta} \mid \mathbf{G})$, Bayes' Theorem yields the marginal and joint posterior distributions

$$p(\mathbf{G} \mid \mathcal{D}) \propto p(\mathbf{G}) p(\mathcal{D} \mid \mathbf{G}) \ , \tag{1}$$

$$p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D}) \propto p(\mathbf{G}) p(\mathbf{\Theta} \mid \mathbf{G}) p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta}) \tag{2}$$

where $p(\mathcal{D} \mid \mathbf{G}) = \int p(\mathbf{\Theta} \mid \mathbf{G}) p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta}) d\mathbf{\Theta}$ is the marginal likelihood. Thus, $p(\mathbf{G} \mid \mathcal{D})$ in (1) is only tractable in special conjugate cases where the integral over $\mathbf{\Theta}$ can be computed in closed form. The Bayesian formalism allows us to compute expectations of the form

$$\mathbb{E}_{\mathbf{G} \mid \mathcal{D}} \Big[ f(\mathbf{G}) \Big] \qquad \text{or} \qquad \mathbb{E}_{\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D}} \Big[ f(\mathbf{G}, \mathbf{\Theta}) \Big] \tag{3}$$

for any function $f$ of interest. For instance, to perform Bayesian model averaging, we would use $f(\mathbf{G}) = p(\mathbf{x} \mid \mathbf{G})$ or $f(\mathbf{G}, \mathbf{\Theta}) = p(\mathbf{x} \mid \mathbf{G}, \mathbf{\Theta})$, respectively [21, 22]. In active learning of causal BN structures, a commonly used $f$ is the expected decrease in entropy of $\mathbf{G}$ after an intervention [12–14, 16]. Inferring either posterior is computationally challenging because there are $\mathcal{O}(d! 2^{\binom{d}{2}})$ possible DAGs with $d$ nodes [23]. Thus, computing the normalization constant $p(\mathcal{D})$ is generally intractable.

**Continuous Characterization of Acyclic Graphs** Orthogonal to the work on Bayesian inference, Zheng et al. [11] have recently proposed a differentiable characterization of acyclic graphs for structure learning. In this work, we adopt the formulation of Yu et al. [24], who show that a graph with adjacency matrix $\mathbf{G} \in \{0, 1\}^{d \times d}$ does not have any cycles if and only if $h(\mathbf{G}) = 0$, where

$$h(\mathbf{G}) := \operatorname{tr} \Big[ (\mathbf{I} + \tfrac{1}{d} \mathbf{G})^d \Big] - d \ . \tag{4}$$

If $h(\mathbf{G}) > 0$, the function can be interpreted as quantifying the *cyclicity* or *non-DAG-ness* of $\mathbf{G}$. Follow-up work has leveraged this insight to model nonlinear relationships [24–27], time-series data [28], in the context of generative modeling [29, 30], for causal inference [31], and contributed to its theoretical understanding [32, 33]. So far, a connection to Bayesian structure learning has been missing.

## 3 Related Work

The existing literature on Bayesian Structure Learning predominantly focuses on inferring the marginal graph posterior $p(\mathbf{G} \mid \mathcal{D})$. Since this requires $p(\mathcal{D} \mid \mathbf{G})$ to be tractable, inference is mostly limited to BNs with linear Gaussian or Categorical conditional distributions [17, 18]. By contrast, the

formulation we introduce overcomes this fundamental restriction by allowing for joint inference of the graph and the parameters, thereby facilitating the active (causal) discovery of more expressive BNs.

**MCMC** Sampling from the posterior over graphs is the most general approach to approximate Bayesian structure learning. Structure MCMC [34, 35] performs Metropolis-Hastings in the space of DAGs by changing one edge at a time. Several works try to remedy its poor mixing behavior [36–38]. Alternatively, order MCMC draws samples in the smaller but still exponential space of node orders, which typically requires a hard limit on the maximum parent set size [20]. Attempts to correct for its unintended structural bias are themselves NP-hard to compute and/or limit the parent size [36, 39]. By performing variational inference in a continuous latent space, our method circumvents such mixing issues and parent size limitations.

**Bootstrapping** The nonparametric DAG bootstrap [40] performs model averaging by bootstrapping $\mathcal{D}$, where each resampled data set is used to learn a single graph, e.g., using the GES or PC algorithms [8, 9]. The collection of graphs approximates the posterior by weighting each unique graph by its unnormalized posterior probability. In simple cases, a closed-form maximum likelihood parameter estimate may be used to approximate the joint posterior [16], but only if $p(\mathcal{D} \mid \mathbf{G})$ is tractable in the first place.

**Exact Methods** A few notable exceptions use dynamic programming to achieve exact marginal inference in time $O(d2^d)$, which is only feasible for $d \leq 20$ nodes [41, 42]. In special cases, e.g., for tree structures or known node orderings, exact inference can be performed more efficiently [43, 44].

# 4 A Fully Differentiable Framework for Bayesian Structure Learning

## 4.1 General Approach

With the goal of moving beyond the restrictive conjugate setups required by most discrete sampling methods for Bayesian structure learning, we propose to transfer the posterior inference task into the latent space of a probabilistic graph representation. Our resulting framework is *consistent* with the original Bayesian structure learning task in (3), enforces the *acyclicity* of $\mathbf{G}$ via the latent space prior, and provides the *score* of the continuous latent posterior, thus making general purpose inference methods applicable off-the-shelf.

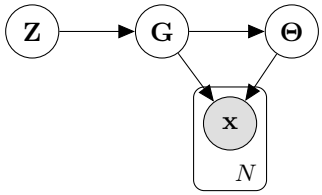



Figure 1: The generative model of BNs with latent variable $\mathbf{Z}$. This graphical model generalizes the standard Bayesian setup in (2) where only $\mathbf{G}$, $\mathbf{\Theta}$, and $\mathbf{x}$ are modeled explicitly.

Without loss of generality, we assume that there exists a latent variable $\mathbf{Z}$ that models the generative process of $\mathbf{G}$. Specifically, the default generative model in Section 2 is generalized into the following factorization:

$$p(\mathbf{Z}, \mathbf{G}, \mathbf{\Theta}, \mathcal{D}) = p(\mathbf{Z})p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta}) \tag{5}$$

Figure 1 displays the corresponding graphical model. The following insight provides us with an equivalence between the expectation we ultimately want to approximate and an expectation over the posterior of the continuous variable $\mathbf{Z}$:

**Proposition 1** (Latent posterior expectation)**.** *Under the generative model in (5), it holds that*

$$\textit{(a)} \quad \mathbb{E}_{\mathbf{G} \mid \mathcal{D}}\Big[f(\mathbf{G})\Big] \;=\; \mathbb{E}_{\mathbf{Z} \mid \mathcal{D}}\left[\frac{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\left[f(\mathbf{G})p(\mathcal{D} \mid \mathbf{G})\right]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\left[p(\mathcal{D} \mid \mathbf{G})\right]}\right], \quad \textit{and}$$

$$\textit{(b)} \quad \mathbb{E}_{\mathbf{G},\mathbf{\Theta} \mid \mathcal{D}}\Big[f(\mathbf{G}, \mathbf{\Theta})\Big] \;=\; \mathbb{E}_{\mathbf{Z},\mathbf{\Theta} \mid \mathcal{D}}\left[\frac{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\left[f(\mathbf{G}, \mathbf{\Theta})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})\right]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\left[p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})\right]}\right].$$

A proof is provided in Appendix A.1. Rather than approximating $p(\mathbf{G} \mid \mathcal{D})$ or $p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D})$, our goal will be to infer $p(\mathbf{Z} \mid \mathcal{D})$ or $p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D})$ instead, which by Proposition 1 allows us to compute expectations of the form in (3). In the following, we first discuss how to define the two factors $p(\mathbf{G} \mid \mathbf{Z})$ and $p(\mathbf{Z})$ in a way that models only acyclic graphs. Then, we provide the details necessary to perform black box inference of the posteriors of the continuous latent variable $\mathbf{Z}$.

### 4.2 Representing DAGs in a Continuous Latent Space

**Generative Graph Model** We consider the latent variable $\mathbf{Z}$ to be the collection of two embedding matrices $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{k \times d}$, i.e., $\mathbf{Z} = [\mathbf{U}, \mathbf{V}]$. Inspired by Kipf and Welling [45], the generative model of the graph adjacency matrix $\mathbf{G} \in \{0, 1\}^{d \times d}$ is given by an inner product between the latent variables $\mathbf{Z}$:

$$p_\alpha(\mathbf{G} \mid \mathbf{Z}) = \prod_{i=1}^{d} \prod_{j \neq i}^{d} p_\alpha(g_{ij} \mid \mathbf{u}_i, \mathbf{v}_j) \quad \text{with } p_\alpha(g_{ij} = 1 \mid \mathbf{u}_i, \mathbf{v}_j) = \sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j) \tag{6}$$

where $\sigma_\alpha(x) = 1/(1 + \exp(-\alpha x))$ denotes the sigmoid function with temperature parameter $\alpha$. We denote the matrix of edge probabilities in $\mathbf{G}$ given $\mathbf{Z}$ by $\mathbf{G}_\alpha(\mathbf{Z}) \in [0, 1]^{d \times d}$ with

$$\mathbf{G}_\alpha(\mathbf{Z})_{ij} := p_\alpha(g_{ij} = 1 \mid \mathbf{U}, \mathbf{V}) \ . \tag{7}$$

Since we model acyclic graphs, which do not contain self-loops, we define the diagonal elements of $\mathbf{G}_\alpha(\mathbf{Z})$ to be zero. The latent dimensionality $k$ trades off the complexity of the variable interactions with tractability during inference. For $k \geq d$, the matrix of edge probabilities $\mathbf{G}_\alpha(\mathbf{Z})$ is not constrained in rank, and the generative model in (6) can represent any adjacency matrix without self-loops. That said, the size of the latent representation only grows $\mathcal{O}(d \cdot k)$ and, in principle, $k$ can be chosen independently of $d$. Note that the formulation in (6) models *directed* graphs since $\sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j) \neq \sigma_\alpha(\mathbf{u}_j^\top \mathbf{v}_i)$. We can even interpret $\mathbf{u}_i$ and $\mathbf{v}_i$ as node embeddings that may encode more information than mere edge probabilities, e.g., for graph neural networks. The fact that $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ is invariant to orthogonal transformations of $\mathbf{U}$ and $\mathbf{V}$ is not an issue in practice.

**Acyclicity via the Latent Prior Distribution** A major constraint in learning BNs is the acyclicity of $\mathbf{G}$. With the latent graph model $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ in place, we design the prior $p_\beta(\mathbf{Z})$ to act as a soft constraint enforcing that only DAGs are modeled. Specifically, we define the prior of $\mathbf{Z}$ as the product of independent Gaussians with a Gibbs distribution that penalizes the *expected cyclicity* of $\mathbf{G}$ given $\mathbf{Z}$:

$$p_\beta(\mathbf{Z}) \propto \exp\left(-\beta \, \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\Big[h(\mathbf{G})\Big]\right) \prod_{ijk} \mathcal{N}(z_{ijk}; 0, \sigma_z^2) \tag{8}$$

Here, $h(\cdot)$ is the DAG constraint function given in (4) and $\beta$ is a temperature parameter that controls how strongly the acyclicity is enforced. As $\beta \to \infty$, the support of $p_\beta(\mathbf{Z})$ reduces to all $\mathbf{Z}$ that only model valid DAGs. The Gaussian component ensures that the norm of $\mathbf{Z}$ is well-behaved. Traditional graph priors of the form $p(\mathbf{G}) \propto f(\mathbf{G})$ can be flexibly incorporated into $p_\beta(\mathbf{Z})$ by means of an additional factor involving, e.g., $f(\mathbf{G}_\alpha(\mathbf{Z}))$ or $\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}[f(\mathbf{G})]$.

### 4.3 Black Box Bayesian Inference

The extended generative model in (5) not only allows us to incorporate the notoriously difficult acyclicity constraint into the Bayesian framework. By rephrasing the posteriors $p(\mathbf{G} \mid \mathcal{D})$ and $p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D})$ in terms of $p(\mathbf{Z} \mid D)$ and $p(\mathbf{Z}, \mathbf{\Theta} \mid D)$, respectively, it also makes Bayesian structure learning amenable to variational inference techniques that operate in continuous space and rely on the gradient of the unnormalized log posterior, also known as the *score*. The following result provides us with the score of both latent posteriors. Detailed derivations can be found in Appendix A.2.

**Proposition 2** (Latent posterior score)**.** *Under the generative graph model defined in (5), the gradient of the log posterior density of* $\mathbf{Z}$ *is given by*

$$(a) \qquad \nabla_{\mathbf{Z}} \log p(\mathbf{Z} \mid \mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}} \, \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathcal{D} \mid \mathbf{G})\big]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathcal{D} \mid \mathbf{G})\big]} \tag{9}$$

*which is relevant when the marginal likelihood* $p(\mathcal{D} \mid \mathbf{G})$ *can be computed efficiently. In the general case, when inferring the joint posterior of* $\mathbf{Z}$ *and* $\mathbf{\Theta}$*, the gradients of the log posterior are given by*

$$(b) \qquad \nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}} \, \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})\big]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})\big]} \tag{10}$$

$$(c) \qquad \nabla_{\mathbf{\Theta}} \log p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) = \frac{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[\nabla_{\mathbf{\Theta}} p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})\big]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})\big]} \tag{11}$$

*where* $p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G}) = p(\mathbf{\Theta} \mid \mathbf{G}) p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})$*, which is efficient to compute by construction.*

In both cases, the expectations have tractable Monte Carlo approximations because sampling from $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ in (6) is simple and parallelizable. The gradient terms of the form $\nabla_{\mathbf{Z}} \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}[\,\cdot\,]$, which also appear in $\nabla_{\mathbf{Z}} \log p_\beta(\mathbf{Z})$, can be estimated using two different techniques, depending on the setting.

**Differentiable (Marginal) Likelihood** Using the Gumbel-softmax trick [46, 47], we can separate the randomness from $\mathbf{Z}$ when sampling from $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ and obtain the following estimator:

$$\nabla_{\mathbf{Z}} \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathcal{D} \mid \mathbf{G})\big] \approx \mathbb{E}_{\mathbf{L}}\Big[\nabla_{\mathbf{G}}\, p(\mathcal{D} \mid \mathbf{G})\big|_{\mathbf{G}=\mathbf{G}_\tau(\mathbf{L},\mathbf{Z})} \cdot \nabla_{\mathbf{Z}}\, \mathbf{G}_\tau(\mathbf{L}, \mathbf{Z})\Big] \tag{12}$$

where $\mathbf{L} \sim \text{Logistic}(0,1)^{d \times d}$. The matrix-valued function $\mathbf{G}_\tau(\cdot)$ is defined elementwise as

$$\mathbf{G}_\tau(\mathbf{L}, \mathbf{Z})_{ij} := \begin{cases} \sigma_\tau\left(l_{ij} + \alpha \mathbf{u}_i^\top \mathbf{v}_j\right) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{13}$$

The estimator also applies to $p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})$. However, the gradients $\nabla_{\mathbf{G}} p(\mathcal{D} \mid \mathbf{G})$ or $\nabla_{\mathbf{G}} p(\mathcal{D}, \mathbf{\Theta} \mid \mathbf{G})$, respectively, need to be well-defined and computable, which depends on the parameterization of the BN model we aim to infer. In case $p(\mathcal{D} \mid \mathbf{G})$ or $p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})$ is only defined for discrete $\mathbf{G}$, it is possible to evaluate $\nabla_{\mathbf{G}} p(\mathcal{D} \mid \mathbf{G})$ or $\nabla_{\mathbf{G}} p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})$ using hard Gumbel-max samples of $\mathbf{G}$ (i.e., with $\tau = \infty$) and a straight-through gradient estimator. Since the DAG constraint $h(\cdot)$ is differentiable, the Gumbel-softmax trick can always be applied inside $\nabla_{\mathbf{Z}} \log p_\beta(\mathbf{Z})$. In practice, we always use $\tau = 1$.

**Non-Differentiable (Marginal) Likelihood** In general, $\nabla_{\mathbf{G}} p(\mathcal{D} \mid \mathbf{G})$ or $\nabla_{\mathbf{G}} p(\mathcal{D}, \mathbf{\Theta} \mid \mathbf{G})$, depending on the task, might be not available or ill-defined. In this setting, the score function estimator provides us with a way to estimate the gradient we need [48]:

$$\nabla_{\mathbf{Z}} \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathcal{D} \mid \mathbf{G})\big] = \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\Big[\big(p(\mathcal{D} \mid \mathbf{G}) - b\big)\, \nabla_{\mathbf{Z}} \log p_\alpha(\mathbf{G} \mid \mathbf{Z})\Big] \tag{14}$$

and likewise for $p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})$ in place of $p(\mathcal{D} \mid \mathbf{G})$. Here, $b$ is a constant w. r. t. $\mathbf{G}$ that can be used for variance reduction [49], and $\nabla_{\mathbf{Z}} \log p_\alpha(\mathbf{G} \mid \mathbf{Z})$ is trivial to compute. The derivations of both (12) and (14), alongside a more detailed discussion, can be found in Appendix B.

# 5 Particle Variational Inference for Structure Learning

In the previous section, we have proposed a *differentiable* formulation for *Bayesian structure learning (DiBS)* that is agnostic to the form of the local BN conditionals and, more importantly, translates learning discrete graph structures into an inference problem over the continuous variable $\mathbf{Z}$. In the following, we overcome the remaining challenge of inferring the intractable DiBS posteriors $p(\mathbf{Z} \mid \mathcal{D})$ and $p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D})$ by employing Stein variational gradient descent (SVGD) [19], a gradient-based and general purpose variational inference method. The resulting algorithm infers a particle approximation of the marginal or joint posterior density over BNs given observational data.

**SVGD for Posterior Inference** Since Proposition 2 provides us with the gradient of the latent posterior score functions, we can apply SVGD off-the-shelf. SVGD approximates a target distribution by iteratively *transporting* a set of particles using a sequence of kernel-based transformation steps. Following this paradigm for DiBS, we iteratively update a fixed set $\{\mathbf{Z}^{(m)}\}_{m=1}^M$ or $\{\mathbf{Z}^{(m)}, \mathbf{\Theta}^{(m)}\}_{m=1}^M$ to approximate $p(\mathbf{Z} \mid \mathcal{D})$ or $p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D})$, respectively. If the BN model we aim to infer has a properly-defined likelihood gradient with respect to $\mathbf{G}$, we use the Gumbel-softmax estimator in (12) to compute the score function. Otherwise, we resort to the score function estimator in (14). When inferring $p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D})$, we use a simple RBF kernel in the underlying SVGD updates:

$$k\big((\mathbf{Z}, \mathbf{\Theta}), (\mathbf{Z}', \mathbf{\Theta}')\big) := \exp\left(-\frac{1}{\gamma_z}\|\mathbf{Z} - \mathbf{Z}'\|_2^2\right) + \exp\left(-\frac{1}{\gamma_\theta}\|\mathbf{\Theta} - \mathbf{\Theta}'\|_2^2\right) \tag{15}$$

with bandwidths $\gamma_z, \gamma_\theta$. For inference of $p(\mathbf{Z} \mid \mathcal{D})$, we leave out the second term involving $\mathbf{\Theta}, \mathbf{\Theta}'$. While $\mathbf{Z}$ is invariant to orthogonal transformations, more elaborate kernels that are, e.g., invariant to such transforms empirically perform worse in our studies.

**Annealing $\alpha$ and $\beta$** The latent variable $\mathbf{Z} = [\mathbf{U}, \mathbf{V}]$ not only probabilistically models the graph $\mathbf{G}$, but can also be viewed as a continuous relaxation of $\mathbf{G}$, with $\alpha$ trading off smoothness with accuracy. As $\alpha \to \infty$, the sigmoid $\sigma_\alpha(\cdot)$ converges to the unit step function, and thus, under the graph model $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ in (5), the expectations in Proposition 1 simplify to:

$$\begin{aligned} \mathbb{E}_{\mathbf{G} \mid \mathcal{D}}\Big[f(\mathbf{G})\Big] &\to \mathbb{E}_{\mathbf{Z} \mid \mathcal{D}}\Big[f\big(\mathbf{G}_\infty(\mathbf{Z})\big)\Big] \\ \mathbb{E}_{\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D}}\Big[f(\mathbf{G}, \mathbf{\Theta})\Big] &\to \mathbb{E}_{\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}}\Big[f\big(\mathbf{G}_\infty(\mathbf{Z}), \mathbf{\Theta}\big)\Big] \end{aligned} \tag{16}$$

**Algorithm 1** DiBS for $p(\mathbf{G} \mid \mathcal{D})$ using Stein variational gradient descent [19]

**Input:** Initial set of latent particles $\{\mathbf{Z}_0^{(m)}\}_{m=1}^M$, kernel $k(\cdot,\cdot)$, schedules $\eta(t)$, $\alpha(t)$, $\beta(t)$

**Output:** Set of discrete graph particles $\{\mathbf{G}^{(m)}\}_{m=1}^M$ approximating $p(\mathbf{G} \mid \mathcal{D})$

Incorporate prior belief of $p(\mathbf{G})$ into $p_\beta(\mathbf{Z})$ ▷ See Section 4.2

**for** iteration $t = 0$ to $T-1$ **do**

  Estimate score $\nabla_{\mathbf{Z}} \log p(\mathbf{Z} \mid \mathcal{D})$ given in (9) for each $\mathbf{Z}_t^{(m)}$ ▷ See (12) and (14)

  **for** particle $m = 1$ to $M$ **do**

    $\mathbf{Z}_{t+1}^{(m)} \leftarrow \mathbf{Z}_t^{(m)} + \eta(t)\, \boldsymbol{\phi}_t(\mathbf{Z}_t^{(m)})$ ▷ SVGD transport step

$$\text{where } \boldsymbol{\phi}_t(\cdot) := \frac{1}{M} \sum_{k=1}^{M} \Big[ k(\mathbf{Z}_t^{(k)}, \cdot)\, \nabla_{\mathbf{Z}_t^{(k)}} \log p(\mathbf{Z}_t^{(k)} \mid \mathcal{D}) + \nabla_{\mathbf{Z}_t^{(k)}} k(\mathbf{Z}_t^{(k)}, \cdot) \Big]$$

**return** $\{\mathbf{G}_\infty(\mathbf{Z}_T^{(m)})\}_{m=1}^M$ ▷ See (16) and (17)

where $\mathbf{G}_\infty(\mathbf{Z})$ denotes the single limiting graph implied by $\mathbf{Z}$ and is defined as

$$\mathbf{G}_\infty(\mathbf{Z})_{ij} := \begin{cases} 1 & \text{if } \mathbf{u}_i^\top \mathbf{v}_j > 0 \text{ and } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

In this case, $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ converges to representing only a single graph. To be able to invoke this simplification, we anneal $\alpha \to \infty$ over the iterations of SVGD and, upon termination, convert the latent variables $\mathbf{Z}$ to a single discrete $\mathbf{G}_\infty(\mathbf{Z})$. Furthermore, we similarly let $\beta \to \infty$ in the latent prior $p_\beta(\mathbf{Z})$ over the iterations to enforce that the latent representation of $\mathbf{G}$ only models DAGs. By Equation (16), the resulting DAGs form a consistent particle approximation of $p(\mathbf{G} \mid \mathcal{D})$ or $p(\mathbf{G}, \boldsymbol{\Theta} \mid \mathcal{D})$, respectively. Algorithm 1 summarizes DiBS instantiated with SVGD for inference of $p(\mathbf{G} \mid \mathcal{D})$. The general case of inferring $p(\mathbf{G}, \boldsymbol{\Theta} \mid \mathcal{D})$ is given in Algorithm 2 of Appendix C.

**Single-Particle Approximation** SVGD reduces to regular gradient ascent for the maximum a posteriori estimate when using only a *single* particle [19]. In this case, DiBS with SVGD recovers some of the existing continuous structure learning methods: gradient ascent on a linear Gaussian likelihood solves an optimization problem similar to NOTEARS [11]. The cyclicity penalizer acts analogously. However, not only does DiBS automatically turn into a full Bayesian approach when using more particles, it is also not limited to settings such as linear Gaussian conditionals, where the adjacency matrix $\mathbf{G}$ and the parameters $\boldsymbol{\Theta}$ can be modeled together by a weighted adjacency matrix.

**Weighted Particle Mixture** In high dimensional settings, it may be beneficial to move beyond a uniform weighting of the inferred particles of BN models to approximate $p(\mathbf{G} \mid \mathcal{D})$ or $p(\mathbf{G}, \boldsymbol{\Theta} \mid \mathcal{D})$. We consider a particle mixture that weights each particle by its unnormalized posterior probability $p(\mathbf{G}, \mathcal{D})$ or $p(\mathbf{G}, \boldsymbol{\Theta}, \mathcal{D})$, respectively, under the BN model. In our experiments, DiBS and its instantiation with SVGD are used interchangeably, and DiBS+ denotes the weighted particle mixture.

# 6 Evaluation on Synthetic Data

## 6.1 Experimental Setup

**Synthetic Data** We compare DiBS to a set of related methods in marginal and joint posterior inference of synthetic linear and nonlinear Gaussian networks. Our setup follows [11, 24, 27, 50], who consider inferring BNs with Erdős-Rényi and scale-free random structures [51, 52]. For each graph, here with $d \in \{20, 50\}$ nodes and $2d$ edges in expectation, we sample a set of ground truth parameters and then generate training, held-out, and interventional data sets. In all settings, we use $N = 100$ observations for inference, emulating the use case of Bayesian structure learning where the uncertainty about the graph structure is significant. Gaussian BNs are defined in Appendix D.1.

**Graph Priors** For Erdős-Rényi graphs, all methods use the prior $p(\mathbf{G}) \propto q^{\|\mathbf{G}\|_1} (1-q)^{\binom{d}{2} - \|\mathbf{G}\|_1}$, capturing that each edge exists independently w.p. $q$ [51]. For scale-free graphs, we define the prior $p(\mathbf{G}) \propto \prod_{i=1}^d (1 + \|\mathbf{G}_i^\top\|_1)^{-3}$, analogous to their power law degree distribution $p(\text{deg}) \sim \text{deg}^{-3}$ [52]. Here, $\mathbf{G}_i^\top$ is the $i$-th column of the adjacency matrix. DiBS implements either prior by using the corresponding term above as an additional factor in $p(\mathbf{Z})$ with $\mathbf{G} := \mathbf{G}_\alpha(\mathbf{Z})$ (see Section 4.2).

**Metrics** Since neither density nor samples of the ground truth posteriors are available for BNs of $d \in \{20, 50\}$ variables, we follow the evaluation metrics used by previous work. We define the

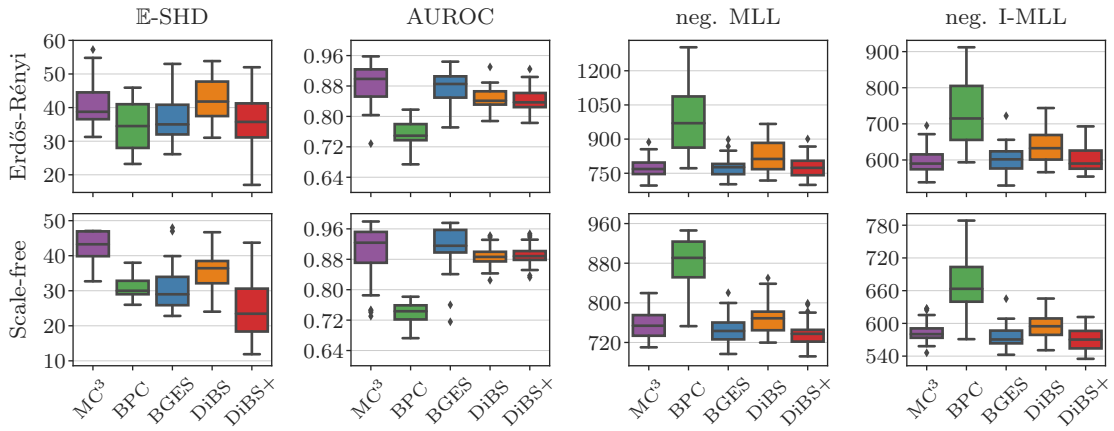



Figure 2: Marginal posterior inference of 20-node linear Gaussian BNs using the BGe marginal likelihood. Higher scores on AUROC and lower scores on $\mathbb{E}$-SHD, neg. MLL, neg. I-MLL are preferred. DiBS+ performs competitively across all metrics, in particular $\mathbb{E}$-SHD and neg. (I-)MLL.

*expected structural Hamming distance* to the ground truth structure $\mathbf{G}^*$ as

$$\mathbb{E}\text{-SHD}(p, \mathbf{G}^*) := \sum_{\mathbf{G}} p(\mathbf{G} \mid \mathcal{D}) \cdot \text{SHD}(\mathbf{G}, \mathbf{G}^*) \tag{18}$$

where $\text{SHD}(\mathbf{G}, \mathbf{G}^*)$ counts the edge changes that separate the essential graphs representing the MECs of $\mathbf{G}$ and $\mathbf{G}^*$ [10, 53]. In addition, we follow [14, 20, 39] and compute the *area under the ROC curve (AUROC)* [54] for pairwise edge predictions when varying the confidence threshold under the inferred marginal $p(g_{ij} = 1 \mid \mathcal{D})$. Following [12], we evaluate the ability to predict future observations by computing the average *negative (marginal) log likelihood* on 100 held-out observations $\mathcal{D}^{\text{test}}$:

$$\text{neg. LL}(p, \mathcal{D}^{\text{test}}) := -\sum_{\mathbf{G}, \mathbf{\Theta}} p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D}) \cdot \log p(\mathcal{D}^{\text{test}} \mid \mathbf{G}, \mathbf{\Theta}) \tag{19}$$

When inferring $p(\mathbf{G} \mid \mathcal{D})$, the corresponding neg. MLL metric instead uses $p(\mathcal{D}^{\text{test}} \mid \mathbf{G})$. Analogously, we also compute the *interventional* log likelihoods I-LL and I-MLL, a relevant metric in causal inference [12, 14]. Here, an interventional data set $(\mathcal{D}^{\text{int}}, \mathcal{I})$ is instead used to compute $p(\mathcal{D}^{\text{int}} \mid \mathbf{G}, \mathbf{\Theta}, \mathcal{I})$ and $p(\mathcal{D}^{\text{int}} \mid \mathbf{G}, \mathcal{I})$ in (19), respectively. For this, we randomly select 10% of the variables and clamp them to zero in the data-generating process. Scores are the average of 10 interventional data sets. All reported metrics are aggregates for inference of 30 random BNs.

In the remainder, DiBS is always run for 3,000 iterations and with $k = d$ for inference of $d$-variable BNs, which leaves the matrix of edge probabilities unconstrained in rank. We discard a DiBS particle in the rare case that a graph is cyclic. Specific hyperparameters are provided in Appendix D.2.

### 6.2 Linear Gaussian Bayesian Networks

**Marginal Posterior Inference** For linear Gaussian BNs, we first evaluate the classical setting of inferring the marginal posterior $p(\mathbf{G} \mid \mathcal{D})$ since $p(\mathcal{D} \mid \mathbf{G})$ can be computed in closed form. We use the Bayesian Gaussian Equivalent (BGe) marginal likelihood, which scores Markov equivalent structures equally [17]. The form of the BGe score requires DiBS to employ the score function estimator in (14).

We compare DiBS with the nonparametric DAG bootstrap [40] using the constraint-based PC [9] and the score-based GES [8] algorithms (BPC and BGES). For MCMC, we only consider structure MCMC (MC$^3$) [35] as a comparison. Order MCMC or hybrid DP approaches bound the number of parents and thus often exclude the ground truth graph a priori, especially for scale-free BN structures. Burn-in and thinning for MC$^3$ are chosen to make the runtime comparable with DiBS. The DAG bootstrap methods typically run significantly faster. Further details are provided in Appendix D.3. In the remainder of the paper, each method uses 30 samples to approximate the posterior.

Figure 2 summarizes the results for 30 randomly generated BNs with $d = 20$ nodes. We find that DiBS+ appears to be preferable to DiBS, and that DiBS+ performs well compared to the other methods, all of which were specifically developed for the marginal inference scenario evaluated here.

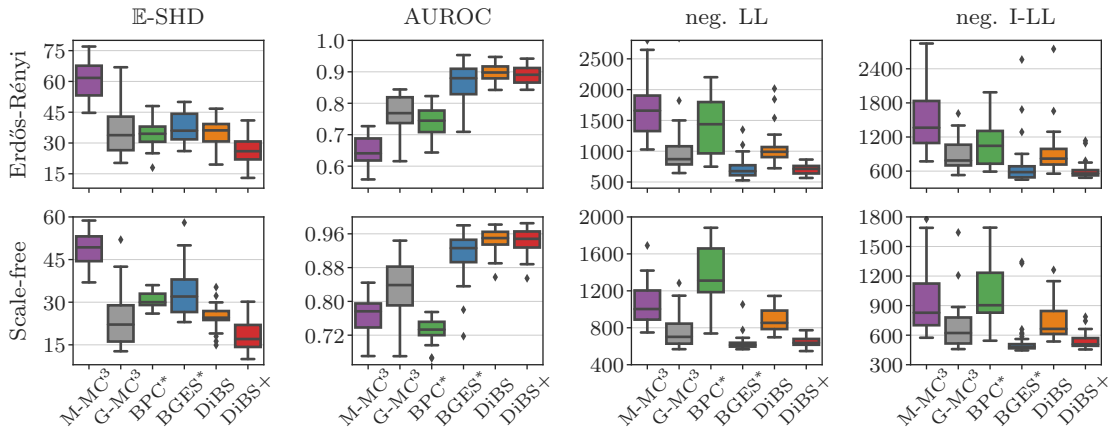



Figure 3: Joint posterior inference of graphs and parameters of linear Gaussian networks with $d = 20$ nodes. DiBS+ performs best across all of the metrics. BGES$^*$, the next-best alternative, yields substantially worse performance in $\mathbb{E}$-SHD, i.e., in recovering the overall graph structure and MEC. Recall that higher AUROC and lower $\mathbb{E}$-SHD, neg. LL, and neg. I-LL scores are preferable.

**Joint posterior inference** When inferring the joint posterior $p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D})$, we can employ a more explicit representation of linear Gaussian BNs, where the conditional distribution parameters are standard Gaussian, and DiBS can employ the Gumbel-softmax estimator in (12) (see Appendix D.1). To provide a comparison with DiBS in the absence of a comparable MCMC method, we propose two variants of MC$^3$ as baselines. Metropolis-Hastings MC$^3$ (M-MC$^3$) jointly samples parameters and structures, and Metropolis-within-Gibbs MC$^3$ (G-MC$^3$) alternates in proposing structure and parameters [55]. Moreoever, we extend the bootstrap methods by taking the maximum likelihood estimate [56] as the posterior parameter sample for a given graph inferred using the BGe score (BPC$^*$ and BGES$^*$), an approach previously used in, e.g., causal BN learning [16].

Figure 3 shows the results for $d = 20$ nodes, where $\mathbb{E}$-SHD and AUROC are computed by empirically marginalizing out the parameters. We find that DiBS+ is the only considered method that performs well across all of the metrics, often outperforming the baselines by a significant margin. As for marginal posterior inference of linear Gaussian BNs, our results suggest that DiBS+ is generally preferable to DiBS.

### 6.3 Nonlinear Gaussian Bayesian Networks

We also consider joint inference of nonlinear Gaussian BNs where the mean of each local conditional Gaussian is parameterized by a 2-layer dense neural network with five hidden nodes and ReLU activation functions (see Appendix D.1). Since the marginal likelihood does not have a closed form, we are unable to use BPC$^*$ and BGES$^*$ as a means of comparison. Figure 4 displays the results for $d = 20$ variables, where a given BN model has $|\mathbf{\Theta}| = 2{,}220$ weights and biases. Analogous to joint inference of linear Gaussian BNs, DiBS and DiBS+ significantly outperform the MCMC baselines across the considered metrics. To the best of our knowledge, this is the first time that such nonlinear Gaussian BN models have been inferred under the Bayesian paradigm, which opens up exciting avenues in the active learning of more sophisticated causal structures. Appendix D.5 extends Sections 6.2 and 6.3 with results for $d = 50$ variables, where DiBS+ likewise performs particularly favorably when jointly inferring $p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D})$.

### 6.4 DiBS: Additional Analyses and Ablation Studies

Having compared DiBS with SVGD to several alternative approaches, we devote Appendix E to empirically analyzing some of the algorithm's properties. One key results is that, all other things held equal, substituting our inner product model of Section 4.2 with $p_\sigma(g_{ij} = 1 \mid \mathbf{Z}) = \sigma_\alpha(\mathbf{z}_{ij})$, where single *scalars* encode the edge probabilities, causes the performance of DiBS to significantly worsen across metrics. In Appendix E, we additionally show the effects of reducing the latent size $k$ and the number of iterations $T$, and study the uncertainty quantification in (non)identifiable edge structures.

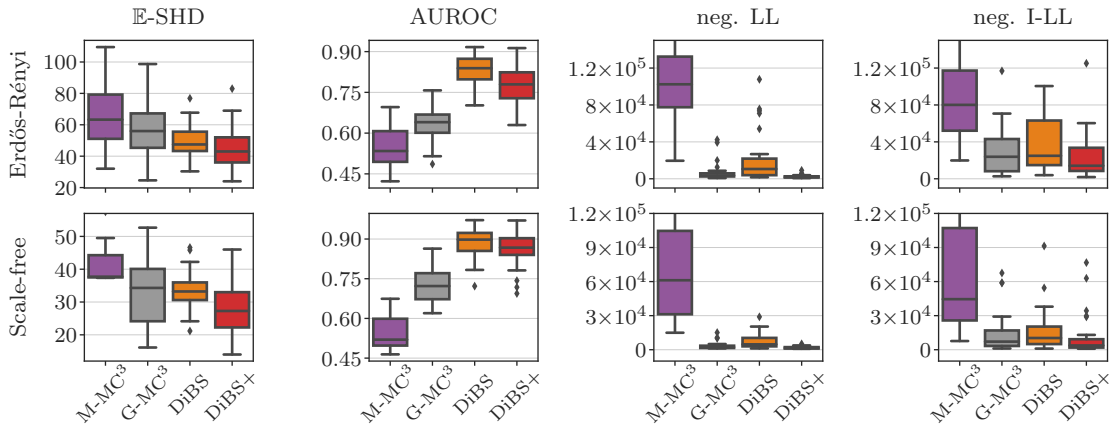



Figure 4: Joint posterior inference of nonlinear Gaussian BNs with $d = 20$ nodes. Here, the mean of each nodes' local conditional distribution is parameterized by a 2-layer neural network with five hidden nodes. DiBS and DiBS+ perform favorably across the board, particularly in the graph metrics.

## 7 Application: Inferring Protein Signaling Networks From Cell Data

A widely used benchmark in structure learning is the proteomics data set by Sachs et al. [3]. The data contain $N = 7,466$ continuous measurements of $d = 11$ proteins involved in human immune system cells as well as an established causal network of their signaling interactions.

We infer both linear and nonlinear Gaussian BNs with Erdős-Rényi graph priors exactly as in Section 6 using the full available data. The 𝔼-SHD scores in Table 1 show that the nonlinear BNs inferred by DiBS are closest in structure to the consensus network, which provides evidence of the value added by modeling nonlinear variable interactions. The AUROC results indicate that the posterior by DiBS under the BGe model provides the most calibrated edge confidence scores. Not penalizing model complexity as much, the BGe posterior by DiBS (DiBS+) averages a higher expected number of edges of 39.0 (35.0), compared to 8.8 (9.9) for linear and 5.2 (6.8) for nonlinear BNs, where graphs and parameters are both inferred. Further details are given in Appendix F.

Table 1: Inference of protein signaling pathways with Gaussian BNs. Metrics are the mean ± SD of 30 random restarts.

| | | 𝔼-SHD | AUROC |
|---|---|---|---|
| † | MC$^3$ | 34.0 ± 0.7 | 0.616 ± 0.027 |
| | BPC | 25.5 ± 2.3 | 0.566 ± 0.020 |
| | BGES | 33.7 ± 1.7 | 0.641 ± 0.030 |
| | DiBS | 37.4 ± 0.5 | **0.647 ± 0.047** |
| | DiBS+ | 34.7 ± 1.5 | 0.629 ± 0.045 |
| § | MH-MC$^3$ | 33.1 ± 3.4 | 0.543 ± 0.11 |
| | G-MC$^3$ | 29.8 ± 3.7 | 0.531 ± 0.08 |
| | DiBS | 20.3 ± 0.4 | 0.601 ± 0.04 |
| | DiBS+ | 20.0 ± 1.4 | 0.569 ± 0.04 |
| ¶ | MH-MC$^3$ | 18.4 ± 0.1 | 0.501 ± 0.043 |
| | G-MC$^3$ | 30.6 ± 2.5 | 0.538 ± 0.060 |
| | DiBS | **17.7 ± 0.1** | 0.550 ± 0.020 |
| | DiBS+ | 18.5 ± 0.5 | 0.530 ± 0.029 |

Applies to all methods in subsection:
† Linear Gaussian BN; graph only via BGe marginal lik.
§ Linear Gaussian BN; graph and parameters jointly
¶ Nonlinear Gaussian BN; graph and parameters jointly

## 8 Conclusion

We have presented a general, fully differentiable approach to inferring posterior distributions over BNs. Our framework is based on a continuous latent representation of DAGs, whose posterior can be equivalently inferred—without loss of generality and using existing black box inference methods. While we have used SVGD [19] for this purpose, our general approach could also be instantiated with, e.g., gradient-based sampling methods [57, 58]. This may improve upon the asymptotic runtime of DiBS, which, by relying on SVGD, scales quadratically in the number of sampled particles. We expect that our end-to-end approach can be extended to handle missing and interventional data as well as to amortized contexts, where rich unstructured data is available.

**Broader Impact** Our work is relevant to any scientific discipline that aims at inferring the (causal) structure of a system or reasoning about the effects of interventions. If algorithms and decisions are grounded in the structural understanding of a data-generating system and take into account the epistemic uncertainty, we expect them to be more robust and have fewer unforeseen side-effects. However, the assumptions allowing for a causal interpretation of DAGs, e.g., the absence of unmeasured confounders, are often untestable and to be taken with care [59], particularly in safety-critical and societally-sensitive applications. Hence, while potential misuse can never be ruled out, our presented method predominantly promises positive societal and scientific impact.

# References


[1] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. 1988.

[2] Dana Pe'er, Aviv Regev, Gal Elidan, and Nir Friedman. Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 17:S215–S224, 2001.

[3] 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(5721):523–529, 2005.

[4] Chikako Van Koten and AR Gray. An application of bayesian network for predicting object-oriented software maintainability. *Information and Software Technology*, 48(1):59–67, 2006.

[5] Philippe Weber, Gabriela Medina-Oliva, Christophe Simon, and Benoît Iung. Overview on bayesian networks applications for dependability, risk analysis and maintenance areas. *Engineering Applications of Artificial Intelligence*, 25(4):671–682, 2012.

[6] Rupesh Agrahari, Amir Foroushani, T Roderick Docking, Linda Chang, Gerben Duns, Monika Hudoba, Aly Karsan, and Habil Zare. Applications of bayesian network models in predicting types of hematological malignancies. *Scientific reports*, 8(1):1–12, 2018.

[7] Al-Amin Baksh, Rouzbeh Abbassi, Vikram Garaniya, and Faisal Khan. Marine transportation risk assessment using bayesian network: Application to arctic waters. *Ocean Engineering*, 159:422–436, 2018.

[8] David Maxwell Chickering. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3: 507–554, March 2003.

[9] Peter Spirtes, Clark N. Glymour, and Richard Scheines. *Causation, prediction, and search*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2nd ed edition, 2000.

[10] Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78, 2006.

[11] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. DAGs with NO TEARS: Continuous optimization for structure learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[12] Kevin P Murphy. Active learning of causal bayes net structure. 2001.

[13] Simon Tong and Daphne Koller. Active learning for structure in bayesian networks. 2001.

[14] Hyunghoon Cho, Bonnie Berger, and Jian Peng. Reconstructing causal biological networks through active learning. *PloS one*, 11(3):e0150611, 2016.

[15] Robert Osazuwa Ness, Karen Sachs, Parag Mallick, and Olga Vitek. A bayesian active learning experimental design for inferring signaling networks. In *International Conference on Research in Computational Molecular Biology*, pages 134–156. Springer, 2017.

[16] Raj Agrawal, Chandler Squires, Karren Yang, Karthikeyan Shanmugam, and Caroline Uhler. ABCD-Strategy: Budgeted experimental design for targeted causal structure discovery. In *Proceedings of Machine Learning Research*, volume 89, pages 3400–3409. PMLR, 16–18 Apr 2019.

[17] Dan Geiger and David Heckerman. Parameter priors for directed acyclic graphical models and the characterization of several probability distributions. *Ann. Statist.*, 30(5):1412–1440, October 2002. Publisher: The Institute of Mathematical Statistics.

[18] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20(3):197–243, September 1995.

[19] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 2378–2386, 2016.

[20] Nir Friedman and Daphne Koller. Being Bayesian About Network Structure. A Bayesian Approach to Structure Discovery in Bayesian Networks. *Machine Learning*, 50(1):95–125, January 2003.

[21] David Madigan and Adrian E Raftery. Model selection and accounting for model uncertainty in graphical models using occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546, 1994.

[22] David Madigan, Jonathan Gavrin, and Adrian E Raftery. Eliciting prior information to enhance the predictive performance of bayesian graphical models. *Communications in Statistics-Theory and Methods*, 24(9):2271–2292, 1995.

[23] R.W. Robinson. Counting labeled acyclic digraphs. *New Directions in the Theory of Graphs*, 1973.

[24] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. DAG-GNN: DAG structure learning with graph neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7154–7163. PMLR, 09–15 Jun 2019.

[25] Sébastien Lachapelle, Philippe Brouillard, Tristan Deleu, and Simon Lacoste-Julien. Gradient-based neural dag learning. In *International Conference on Learning Representations*, 2020.

[26] Ignavier Ng, Shengyu Zhu, Zhitang Chen, and Zhuangyan Fang. A graph autoencoder approach to causal structure learning. In *NeurIPS 2019 Workshop "Do the right thing": machine learning and causal inference for improved decision making*, 2019.

[27] Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric Xing. Learning sparse nonparametric dags. In *International Conference on Artificial Intelligence and Statistics*, pages 3414–3425. PMLR, 2020.

[28] Roxana Pamfil, Nisara Sriwattanaworachai, Shaan Desai, Philip Pilgerstorfer, Konstantinos Georgatzis, Paul Beaumont, and Bryon Aragam. Dynotears: Structure learning from time-series data. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1595–1605. PMLR, 2020.

[29] Mengyue Yang, Furui Liu, Zhitang Chen, Xinwei Shen, Jianye Hao, and Jun Wang. Causalvae: Disentangled representation learning via neural structural causal models, 2020.

[30] Muhan Zhang, Shali Jiang, Zhicheng Cui, Roman Garnett, and Yixin Chen. D-vae: A variational autoencoder for directed acyclic graphs. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[31] Nan Rosemary Ke, Olexa Bilaniuk, Anirudh Goyal, Stefan Bauer, Hugo Larochelle, Chris Pal, and Yoshua Bengio. Learning neural causal models from unknown interventions, 2019.

[32] Dennis Wei, Tian Gao, and Yue Yu. Dags with no fears: A closer look at continuous optimization for learning bayesian networks. *Advances in Neural Information Processing Systems*, 2020.

[33] Ignavier Ng, Sébastien Lachapelle, Nan Rosemary Ke, and Simon Lacoste-Julien. On the convergence of continuous constrained optimization for structure learning. 2020.

[34] David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data. *International Statistical Review/Revue Internationale de Statistique*, pages 215–232, 1995.

[35] Paolo Giudici and Robert Castelo. Improving markov chain monte carlo model search for data mining. *Machine learning*, 50(1-2):127–158, 2003.

[36] Daniel Eaton and Kevin Murphy. Bayesian structure learning using dynamic programming and MCMC. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

[37] Marco Grzegorczyk and Dirk Husmeier. Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2):265, April 2008.

[38] Jack Kuipers and Giusi Moffa. Partition mcmc for inference on acyclic digraphs. *Journal of the American Statistical Association*, 112(517):282–299, 2017.

[39] Byron Ellis and Wing Hung Wong. Learning causal bayesian network structures from experimental data. *Journal of the American Statistical Association*, 103(482):778–789, 2008.

[40] N. Friedman, M. Goldszmidt, and A. J. Wyner. Data analysis with Bayesian networks: A bootstrap approach. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

[41] Mikko Koivisto and Kismat Sood. Exact bayesian structure discovery in bayesian networks. *Journal of Machine Learning Research*, 5(May):549–573, 2004.

[42] Mikko Koivisto. Advances in exact bayesian structure discovery in bayesian networks. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, page 241–248, 2006.

[43] Marina Meila and Tommi S Jaakkola. Tractable bayesian learning of tree belief networks. *Statistics and Computing*, (16):77–92, 2006.

[44] Denver Dash and Gregory F Cooper. Model averaging for prediction with discrete bayesian networks. *Journal of Machine Learning Research*, 5(Sep):1177–1203, 2004.

[45] Thomas N Kipf and Max Welling. Variational graph auto-encoders. 2016.

[46] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings International Conference on Learning Representations*, 2017.

[47] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparametrization with gumbel-softmax. In *Proceedings International Conference on Learning Representations 2017*, April 2017.

[48] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[49] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

[50] Ignavier Ng, AmirEmad Ghassami, and Kun Zhang. On the role of sparsity and dag constraints for learning linear dags. *Advances in Neural Information Processing Systems*, 2020.

[51] P. Erdős and A Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[52] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439): 509–512, 1999.

[53] Steen A Andersson, David Madigan, Michael D Perlman, et al. A characterization of markov equivalence classes for acyclic digraphs. *Annals of statistics*, 25(2):505–541, 1997.

[54] Tom Fawcett. ROC graphs: Notes and practical considerations for researchers. *Machine learning*, 31(1): 1–38, 2004.

[55] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[56] Alain Hauser and Peter Bühlmann. Jointly interventional and observational data: estimation of interventional markov equivalence classes of directed acyclic graphs. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 291–318, 2015.

[57] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov chain Monte Carlo*, 2 (11):2, 2011.

[58] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, 2011.

[59] A. Philip Dawid. Beware of the DAG! In *Proceedings of Workshop on Causality: Objectives and Assessment at NIPS 2008*, pages 59–86, 2010.

[60] Jack Kuipers, Giusi Moffa, and David Heckerman. Addendum on the scoring of Gaussian directed acyclic graphical models. *Ann. Statist.*, 42(4):1689–1691, August 2014. Publisher: The Institute of Mathematical Statistics.

[61] Diviyan Kalainathan, Olivier Goudet, Isabelle Guyon, David Lopez-Paz, and Michèle Sebag. Structural agnostic modeling: Adversarial learning of causal graphs. *arXiv preprint arXiv:1803.04929*, 2018.

[62] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. 2012.

[63] Chris Sherlock, Paul Fearnhead, Gareth O Roberts, et al. The random walk metropolis: linking theory and practice through a case study. *Statistical Science*, 25(2):172–190, 2010.

[64] Dorit Dor and Michael Tarsi. A simple algorithm to construct a consistent extension of a partially oriented graph. 1992.

[65] Yangbo He, Jinzhu Jia, and Bin Yu. Counting and exploring sizes of markov equivalence classes of directed acyclic graphs. *The Journal of Machine Learning Research*, 16(1):2589–2609, 2015.

[66] Diviyan Kalainathan and Olivier Goudet. Causal discovery toolbox: Uncover causal relationships in python, 2019.

[67] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[68] Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Nan Rosemary Ke, Sebastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. A meta-transfer objective for learning to disentangle causal mechanisms. In *International Conference on Learning Representations*, 2020.

[69] Zhuangyan Fang, Shengyu Zhu, Jiji Zhang, Yue Liu, Zhitang Chen, and Yangbo He. Low rank directed acyclic graphs and causal structure learning, 2020.

[70] Frederick Eberhardt, Clark Glymour, and Richard Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, page 178–184, 2005.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See, e.g., Section 8.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 8.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] As indicated in the main text, all proofs and derivations are given in Appendix A and B.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code for our implementation of DiBS with SVGD as well as instructions to reproduce the main experimental results are provided in the supplementary ZIP file as well as at https://sites.google.com/view/96707568/overview.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 6 as well as Appendices D.2 and D.3.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All figures and tables contain box plots with error bars and standard deviations, respectively, over 30 random restarts of the experiments.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendix D.4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] Appendices D.3 and D.4 cite the code libraries used to implement the baseline methods and DiBS with SVGD, respectively. Appendix F references the source of the data used in the applications of Section 7, where the original authors are also cited.
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The code for our implementation of DiBS with SVGD is provided in the supplementary ZIP file, as well as via https://sites.google.com/view/96707568/overview.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The data and models we use are published under an open-access licence.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] The data used for the experiments conducted in Section 7 does not contain data at the level of individuals or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A Proofs of the Main Results

## A.1 Proposition 1

**Proof** For ease of understanding, we recall that the generative model in (5) factors the joint distribution as $p(\mathbf{Z}, \mathbf{G}, \mathbf{\Theta}, \mathcal{D}) = p(\mathbf{Z})p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})$. First, let us consider case (a), the setting where the marginal likelihood $p(\mathcal{D} \mid \mathbf{G})$ can be computed in closed form. We get

$$\mathbb{E}_{\mathbf{G} \mid \mathcal{D}} f(\mathbf{G}) = \sum_{\mathbf{G}} p(\mathbf{G} \mid \mathcal{D}) f(\mathbf{G}) \tag{A.1}$$

$$= \sum_{\mathbf{G}} \frac{p(\mathbf{G}, \mathcal{D}) f(\mathbf{G})}{p(\mathcal{D})} \tag{A.2}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z}, \mathbf{G}, \mathcal{D}) f(\mathbf{G})}{p(\mathcal{D})} d\mathbf{Z} \tag{A.3}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z}) p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G}) f(\mathbf{G})}{p(\mathcal{D})} d\mathbf{Z} \tag{A.4}$$

by the generative model in (5)

$$= \sum_{\mathbf{G}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z} \mid \mathcal{D}) p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G}) f(\mathbf{G})}{p(\mathcal{D} \mid \mathbf{Z})} d\mathbf{Z} \tag{A.5}$$

since $p(\mathbf{Z} \mid \mathcal{D}) = \dfrac{p(\mathbf{Z}) p(\mathcal{D} \mid \mathbf{Z})}{p(\mathcal{D})}$

$$= \int_{\mathbf{Z}} p(\mathbf{Z} \mid \mathcal{D}) \, \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G}) f(\mathbf{G})}{p(\mathcal{D} \mid \mathbf{Z})} d\mathbf{Z} \qquad \text{rearranging} \tag{A.6}$$

$$= \int_{\mathbf{Z}} p(\mathbf{Z} \mid \mathcal{D}) \, \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G}) f(\mathbf{G})}{\sum_{\mathbf{G}} p(\mathbf{G}, \mathcal{D} \mid \mathbf{Z})} d\mathbf{Z} \tag{A.7}$$

by the law of total probability

$$= \int_{\mathbf{Z}} p(\mathbf{Z} \mid \mathcal{D}) \, \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G}) f(\mathbf{G})}{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z}) p(\mathcal{D} \mid \mathbf{G})} d\mathbf{Z} \tag{A.8}$$

expanding $p(\mathbf{G}, \mathcal{D} \mid \mathbf{Z})$ by the generative model in (5)

$$= \mathbb{E}_{\mathbf{Z} \mid \mathcal{D}} \left[ \frac{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}} \big[ f(\mathbf{G}) p(\mathcal{D} \mid \mathbf{G}) \big]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}} \big[ p(\mathcal{D} \mid \mathbf{G}) \big]} \right] \tag{A.9}$$

as desired for (a).

Finally, let us consider (b), the general case. The derivation essentially follows the same ideas as for (a) but does not marginalize out $\mathbf{\Theta}$.

$$\mathbb{E}_{\mathbf{G}, \mathbf{\Theta} \mid D} f(\mathbf{G}, \mathbf{\Theta}) \tag{A.10}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{\Theta}} p(\mathbf{G}, \mathbf{\Theta} \mid \mathcal{D}) f(\mathbf{G}, \mathbf{\Theta}) d\mathbf{\Theta} \tag{A.11}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{\Theta}} \frac{p(\mathbf{G}, \mathbf{\Theta}, \mathcal{D}) f(\mathbf{G}, \mathbf{\Theta})}{p(\mathcal{D})} d\mathbf{\Theta} \tag{A.12}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z}, \mathbf{G}, \mathbf{\Theta}, \mathcal{D}) f(\mathbf{G}, \mathbf{\Theta})}{p(\mathcal{D})} d\mathbf{Z} d\mathbf{\Theta} \tag{A.13}$$

$$= \sum_{\mathbf{G}} \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z})p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})f(\mathbf{G}, \mathbf{\Theta})}{p(\mathcal{D})} d\mathbf{Z}d\mathbf{\Theta} \tag{A.14}$$

by the generative model in (5)

$$= \sum_{\mathbf{G}} \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} \frac{p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D})p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})f(\mathbf{G}, \mathbf{\Theta})}{p(\mathcal{D}, \mathbf{\Theta} \mid \mathbf{Z})} d\mathbf{Z}d\mathbf{\Theta} \tag{A.15}$$

since $p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) = \dfrac{p(\mathbf{Z})p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})}{p(\mathcal{D})}$

$$= \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})f(\mathbf{G}, \mathbf{\Theta})}{p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})} d\mathbf{Z}d\mathbf{\Theta} \tag{A.16}$$

rearranging

$$= \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})f(\mathbf{G}, \mathbf{\Theta})}{\sum_{\mathbf{G}} p(\mathbf{G}, \mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})} d\mathbf{Z}d\mathbf{\Theta} \tag{A.17}$$

by the law of total probability

$$= \int_{\mathbf{\Theta}} \int_{\mathbf{Z}} p(\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}) \frac{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})f(\mathbf{G}, \mathbf{\Theta})}{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})} d\mathbf{Z}d\mathbf{\Theta} \tag{A.18}$$

expanding $p(\mathbf{G}, \mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})$ by the generative model in (5)

$$= \mathbb{E}_{\mathbf{Z}, \mathbf{\Theta} \mid \mathcal{D}} \left[ \frac{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[f(\mathbf{G}, \mathbf{\Theta})p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})\big]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[p(\mathbf{\Theta} \mid \mathbf{G})p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})\big]} \right] \tag{A.19}$$

which is the statement in (b). □

### A.2 Proposition 2

**Proof** We will derive the gradients of the unnormalized posterior since

$$\nabla_{\mathbf{Z}} \log p(\mathbf{Z} \mid \mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \mathcal{D}) - \nabla_{\mathbf{Z}} \log p(\mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \mathcal{D}) \tag{A.20}$$

and analogously for the other two expressions. Through straightforward manipulation and using the fact that $\nabla_{\mathbf{x}} \log f(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x}) / f(\mathbf{x})$, we obtain

$$\nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \nabla_{\mathbf{Z}} \log p(\mathcal{D} \mid \mathbf{Z}) \tag{A.21}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}}\, p(\mathcal{D} \mid \mathbf{Z})}{p(\mathcal{D} \mid \mathbf{Z})} \tag{A.22}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}} \big[\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathcal{D} \mid \mathbf{G})\big]}{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathcal{D} \mid \mathbf{G})} \tag{A.23}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}}\, \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}[p(\mathcal{D} \mid \mathbf{G})]}{\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}[p(\mathcal{D} \mid \mathbf{G})]} \tag{A.24}$$

Analogously, we get

$$\nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \mathbf{\Theta}, \mathcal{D}) = \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \nabla_{\mathbf{Z}} \log p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z}) \tag{A.25}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}}\, p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})}{p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{Z})} \tag{A.26}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}} \big[\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})\big]}{\sum_{\mathbf{G}} p(\mathbf{G} \mid \mathbf{Z})p(\mathbf{\Theta}, \mathcal{D} \mid \mathbf{G})} \tag{A.27}$$

$$= \nabla_{\mathbf{Z}} \log p(\mathbf{Z}) + \frac{\nabla_{\mathbf{Z}} \, \mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}}[p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})]}{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}}[p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})]} \tag{A.28}$$

Lastly, using the same ideas as above, we arrive at

$$\begin{aligned}
\nabla_{\mathbf{\Theta}} \log p(\mathbf{Z}, \mathbf{\Theta}, \mathcal{D}) &= \nabla_{\mathbf{\Theta}} \log p(\mathbf{Z}) + \nabla_{\mathbf{\Theta}} \log p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{Z}) && \text{(A.29)}\\
&= \frac{\nabla_{\mathbf{\Theta}} \, p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{Z})}{p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{Z})} && \text{(A.30)}\\
&= \frac{\nabla_{\mathbf{\Theta}} \big[ \sum_{\mathbf{G}} p(\mathbf{G} \,|\, \mathbf{Z}) p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G}) \big]}{\sum_{\mathbf{G}} p(\mathbf{G} \,|\, \mathbf{Z}) p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})} && \text{(A.31)}\\
&= \frac{\sum_{\mathbf{G}} p(\mathbf{G} \,|\, \mathbf{Z}) \nabla_{\mathbf{\Theta}} p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})}{\sum_{\mathbf{G}} p(\mathbf{G} \,|\, \mathbf{Z}) p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})} && \text{(A.32)}\\
&= \frac{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}} \big[ \nabla_{\mathbf{\Theta}} p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G}) \big]}{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}}[p(\mathbf{\Theta}, \mathcal{D} \,|\, \mathbf{G})]} \quad \square && \text{(A.33)}
\end{aligned}$$

In the above, without any additional factor modeling a prior belief over graphs, the score of the latent prior $p_\beta(\mathbf{Z})$ defined as in (8) is given by

$$\nabla_{\mathbf{Z}} \log p_\beta(\mathbf{Z}) = -\beta \, \nabla_{\mathbf{Z}} \, \mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}}[h(\mathbf{G})] - \frac{1}{\sigma_z^2} \mathbf{Z} \tag{A.34}$$

### A.3 Equations (16) and (17)

**Proof** The sigmoid function converges to the unit step function, i.e. $\sigma_\alpha(x) \to \mathbb{1}[x > 0]$ as $\alpha \to \infty$. Hence, the edge probabilities $\mathbf{G}_\alpha(\mathbf{Z})$ defined in (7) converge to a (discrete) matrix $\mathbf{G}$ as $\alpha \to \infty$. Extending the notation of (7), we will denote this single limiting graph implied by $\mathbf{Z}$ as $\mathbf{G}_\infty(\mathbf{Z})$ where

$$\mathbf{G}_\infty(\mathbf{Z})_{ij} := \begin{cases} 1 & \text{if } \mathbf{u}_i^\top \mathbf{v}_j > 0 \text{ and } i \neq j \\ 0 & \text{otherwise} \end{cases}$$

The above implies that when the temperature parameter $\alpha$ is annealed to $\infty$, the probability mass function and correspondingly the expectation simplify:

$$\begin{aligned}
\text{As } \alpha \to \infty : \quad & p_\alpha(\mathbf{G} \,|\, \mathbf{Z}) \to \mathbb{1}[\mathbf{G} = \mathbf{G}_\infty(\mathbf{Z})] \\
& \mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}}[f(\mathbf{G})] \to f(\mathbf{G}_\infty(\mathbf{Z}))
\end{aligned} \tag{A.35}$$

From here onward, the temperature hyperparameters $\alpha, \beta$ will not be specifically indicated to reduce clutter in notation. Again, let us first consider case (a). Starting with Proposition 1(a) in the first step, we can use the above insight to simplify the inner expectations:

$$\begin{aligned}
\mathbb{E}_{\mathbf{G} \,|\, \mathcal{D}} \Big[ f(\mathbf{G}) \Big] &= \mathbb{E}_{\mathbf{Z} \,|\, \mathcal{D}} \left[ \frac{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}} \big[ f(\mathbf{G}) p(\mathcal{D} \,|\, \mathbf{G}) \big]}{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}} \big[ p(\mathcal{D} \,|\, \mathbf{G}) \big]} \right] && \text{(A.36)}\\
&\overset{\alpha \to \infty}{\longrightarrow} \mathbb{E}_{\mathbf{Z} \,|\, \mathcal{D}} \left[ \frac{f(\mathbf{G}_\infty(\mathbf{Z})) p(\mathcal{D} \,|\, \mathbf{G}_\infty(\mathbf{Z}))}{p(\mathcal{D} \,|\, \mathbf{G}_\infty(\mathbf{Z}))} \right] && \text{(A.37)}\\
&= \mathbb{E}_{\mathbf{Z} \,|\, \mathcal{D}} \Big[ f(\mathbf{G}_\infty(\mathbf{Z})) \Big] && \text{(A.38)}
\end{aligned}$$

Analogously, we get for the general case (b):

$$\begin{aligned}
\mathbb{E}_{\mathbf{G}, \mathbf{\Theta} \,|\, \mathcal{D}} \Big[ f(\mathbf{G}, \mathbf{\Theta}) \Big] &= \mathbb{E}_{\mathbf{Z}, \mathbf{\Theta} \,|\, \mathcal{D}} \left[ \frac{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}} \left[ f(\mathbf{G}, \mathbf{\Theta}) p(\mathbf{\Theta} \,|\, \mathbf{G}) p(\mathcal{D} \,|\, \mathbf{G}, \mathbf{\Theta}) \right]}{\mathbb{E}_{\mathbf{G} \,|\, \mathbf{Z}} \left[ p(\mathbf{\Theta} \,|\, \mathbf{G}) p(\mathcal{D} \,|\, \mathbf{G}, \mathbf{\Theta}) \right]} \right] && \text{(A.39)}\\
&\overset{\alpha \to \infty}{\longrightarrow} \mathbb{E}_{\mathbf{Z}, \mathbf{\Theta} \,|\, \mathcal{D}} \left[ \frac{f(\mathbf{G}_\infty(\mathbf{Z}), \mathbf{\Theta}) p(\mathbf{\Theta} \,|\, \mathbf{G}_\infty(\mathbf{Z})) p(\mathcal{D} \,|\, \mathbf{G}_\infty(\mathbf{Z}), \mathbf{\Theta})}{p(\mathbf{\Theta} \,|\, \mathbf{G}_\infty(\mathbf{Z})) p(\mathcal{D} \,|\, \mathbf{G}_\infty(\mathbf{Z}), \mathbf{\Theta})} \right] && \text{(A.40)}\\
&= \mathbb{E}_{\mathbf{Z}, \mathbf{\Theta} \,|\, \mathcal{D}} \Big[ f(\mathbf{G}_\infty(\mathbf{Z}), \mathbf{\Theta}) \Big] \quad \square && \text{(A.41)}
\end{aligned}$$

# B Gradient Estimation for Bayesian Inference

To derive the expressions for the gradient estimators in (14) and (12) for both the marginal likelihood and the likelihood, we will use a generic function $f(\mathbf{G})$ in place of either $p(\mathcal{D} \mid \mathbf{G})$ or $p(\mathcal{D} \mid \mathbf{G}, \mathbf{\Theta})$, since the results hold for general functions $f(\mathbf{G})$.

## B.1 Score Function Estimator for the Likelihood Gradient

To arrive at the estimator in (14), we expand the gradient expression as

$$\nabla_{\mathbf{Z}} \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[f(\mathbf{G})\big] = \nabla_{\mathbf{Z}} \sum_{\mathbf{G}} f(\mathbf{G}) p(\mathbf{G} \mid \mathbf{Z}) \tag{B.1}$$

$$= \sum_{\mathbf{G}} f(\mathbf{G}) \nabla_{\mathbf{Z}} p(\mathbf{G} \mid \mathbf{Z}) \tag{B.2}$$

$$= \sum_{\mathbf{G}} f(\mathbf{G}) p(\mathbf{G} \mid \mathbf{Z}) \nabla_{\mathbf{Z}} \log p(\mathbf{G} \mid \mathbf{Z}) \tag{B.3}$$

$$= \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\Big[f(\mathbf{G})\ \nabla_{\mathbf{Z}} \log p(\mathbf{G} \mid \mathbf{Z})\Big] \tag{B.4}$$

where (B.3) uses the fact that $\nabla_{\mathbf{Z}} \log p(\mathbf{G} \mid \mathbf{Z}) = \nabla_{\mathbf{Z}} p(\mathbf{G} \mid \mathbf{Z}) / p(\mathbf{G} \mid \mathbf{Z})$. Finally, we recall the well-known property of the score function that $\mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}[\nabla_{\mathbf{Z}} \log p(\mathbf{G} \mid \mathbf{Z})] = \mathbf{0}$. Due to this, for any constant $b$ as written in (14), the estimator is unbiased because the additional term involving $b$ has zero expectation. The constant can be used to reduce the variance of the Monte Carlo estimator [49].

## B.2 Gumbel-Softmax Estimator for the Likelihood Gradient

In general, for a Bernoulli random variable $X$ with $p(X = 1) = q$, it holds that

$$X \overset{d}{=} \mathbb{1}\big[G_1 + \log q > G_0 + \log(1 - q)\big] \tag{B.5}$$

when $G_0, G_1 \sim \text{Gumbel}(0, 1)$. This is the so-called Gumbel-max trick. Since the unit step function $\mathbb{1}[\cdot]$ does not have an informative gradient, [46, 47] have proposed to use the sigmoid function $\sigma_\tau(\cdot)$ with parameter $\tau$ as a soft relaxation for $\mathbb{1}[\cdot]$ after re-arranging the inequality into the form "$> 0$".

We can reparameterize the entries of $\mathbf{G}$ under the graph model (6) using the Gumbel-softmax trick. Starting from the Gumbel-max equality in (B.5) where $G_0, G_1 \sim \text{Gumbel}(0, 1)$, we apply the sigmoid relaxation with parameter $\tau$ and obtain:

$$g_{ij} \approx \sigma_\tau\Big(G_1 - G_0 + \log \sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j) - \log(1 - \sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j))\Big) \tag{B.6}$$

$$= \sigma_\tau\left(L + \log\left(\frac{\sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j)}{\sigma_\alpha(-\mathbf{u}_i^\top \mathbf{v}_j)}\right)\right) \tag{B.7}$$

$$= \sigma_\tau\left(L + \log\left(\frac{\exp(\alpha \mathbf{u}_i^\top \mathbf{v}_j)}{\exp(\alpha \mathbf{u}_i^\top \mathbf{v}_j) + 1} \frac{\exp(\alpha \mathbf{u}_i^\top \mathbf{v}_j) + 1}{1}\right)\right) \tag{B.8}$$

$$= \sigma_\tau\left(L + \log\left(\exp(\alpha \mathbf{u}_i^\top \mathbf{v}_j)\right)\right) \tag{B.9}$$

$$= \sigma_\tau\left(L + \alpha \mathbf{u}_i^\top \mathbf{v}_j\right) \tag{B.10}$$

where $L \sim \text{Logistic}(0, 1)$ as $L \overset{d}{=} G_1 - G_0$. For $i = j$, we set $g_{ij} := 0$ by default in accordance with the graph model (6). Since this allows us to separate the randomness in sampling from the distribution $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ from the values of $\mathbf{Z}$, we can move the gradient operator inside the expectation and obtain the estimator given in (12):

$$\begin{aligned} \nabla_{\mathbf{Z}} \mathbb{E}_{\mathbf{G} \mid \mathbf{Z}}\big[f(\mathbf{G})\big] &\approx \mathbb{E}_{\mathbf{L}}\Big[\nabla_{\mathbf{Z}}\, f(\mathbf{G}_\tau(\mathbf{L}, \mathbf{Z}))\Big] \\ &= \mathbb{E}_{\mathbf{L}}\Big[\nabla_{\mathbf{G}}\, f(\mathbf{G})\big|_{\mathbf{G} = \mathbf{G}_\tau(\mathbf{L}, \mathbf{Z})} \cdot \nabla_{\mathbf{Z}}\, \mathbf{G}_\tau(\mathbf{L}, \mathbf{Z})\Big] \end{aligned} \tag{B.11}$$

While the reparameterization trick generally provides a lower variance estimate of the gradient, the form in (12) is biased when $\tau < \infty$ because we use a soft relaxation of the true distribution. In

addition, the estimator in (12) requires that $\nabla_{\mathbf{G}} p(\mathcal{D} \mid \mathbf{G})$ or $\nabla_{\mathbf{G}} p(\boldsymbol{\Theta}, \mathcal{D} \mid \mathbf{G})$ is available, depending on the task. In case $p(\mathcal{D} \mid \mathbf{G})$ or $p(\boldsymbol{\Theta}, \mathcal{D} \mid \mathbf{G})$ is only defined for discrete $\mathbf{G}$, it is possible to evaluate $\nabla_{\mathbf{G}} p(\mathcal{D} \mid \mathbf{G})$ or $\nabla_{\mathbf{G}} p(\boldsymbol{\Theta}, \mathcal{D} \mid \mathbf{G})$ using hard Gumbel-max samples of $\mathbf{G}$ (i.e., with $\tau = \infty$). As before, however, one would use soft Gumbel-softmax samples in $\nabla_{\mathbf{Z}} \mathbf{G}_\tau(\mathbf{L}, \mathbf{Z})$ to obtain an informative gradient. Lastly, we can use this estimator to approximate the gradient of the latent prior $\nabla_{\mathbf{Z}} p_\beta(\mathbf{Z})$ given in (8) because the acyclicity constraint $h(\mathbf{G})$ is differentiable with respect to $\mathbf{G}$.

# C General Algorithm

**Algorithm 2** DiBS for $p(\mathbf{G}, \boldsymbol{\Theta} \mid \mathcal{D})$ using Stein variational gradient descent [19]

**Input:** Initial latent and parameter particles $\{(\mathbf{Z}_0^{(m)}, \boldsymbol{\Theta}_0^{(m)})\}_{m=1}^M$, kernel $k(\cdot,\cdot)$, schedules $\eta(t)$, $\alpha(t)$, $\beta(t)$

**Output:** Set of graph and parameter particles $\{(\mathbf{G}^{(m)}, \boldsymbol{\Theta}^{(m)})\}_{m=1}^M$ approximating $p(\mathbf{G}, \boldsymbol{\Theta} \mid \mathcal{D})$

Incorporate prior belief of $p(\mathbf{G})$ into $p_\beta(\mathbf{Z})$ ▷ See Section 4.2

**for** iteration $t = 0$ to $T-1$ **do**

  Estimate score $\nabla_{\mathbf{Z}} \log p(\mathbf{Z}, \boldsymbol{\Theta} \mid \mathcal{D})$ given in (10) for each $\mathbf{Z}_t^{(m)}$ ▷ See (12) and (14)

  Estimate score $\nabla_{\boldsymbol{\Theta}} \log p(\mathbf{Z}, \boldsymbol{\Theta} \mid \mathcal{D})$ given in (11) for each $\boldsymbol{\Theta}_t^{(m)}$

  **for** particle $m = 1$ to $M$ **do**

    $\mathbf{Z}_{t+1}^{(m)} \leftarrow \mathbf{Z}_t^{(m)} + \eta(t)\, \boldsymbol{\phi}_t^{[\mathbf{Z}]}\big(\mathbf{Z}_t^{(m)}, \boldsymbol{\Theta}_t^{(m)}\big)$ ▷ SVGD transport step

    where $\boldsymbol{\phi}_t^{[\mathbf{Z}]}(\cdot,\cdot) := \frac{1}{M} \sum_{k=1}^{M} \Big[ k\big((\mathbf{Z}_t^{(k)}, \boldsymbol{\Theta}_t^{(k)}),\, (\cdot,\cdot)\big)\, \nabla_{\mathbf{Z}_t^{(k)}} \log p\big(\mathbf{Z}_t^{(k)}, \boldsymbol{\Theta}_t^{(k)} \mid \mathcal{D}\big) + \nabla_{\mathbf{Z}_t^{(k)}} k\big((\mathbf{Z}_t^{(k)}, \boldsymbol{\Theta}_t^{(k)}),\, (\cdot,\cdot)\big) \Big]$

    $\boldsymbol{\Theta}_{t+1}^{(m)} \leftarrow \boldsymbol{\Theta}_t^{(m)} + \eta(t)\, \boldsymbol{\phi}_t^{[\boldsymbol{\Theta}]}\big(\mathbf{Z}_t^{(m)}, \boldsymbol{\Theta}_t^{(m)}\big)$

    where $\boldsymbol{\phi}_t^{[\boldsymbol{\Theta}]}(\cdot,\cdot)$ is analogous to $\boldsymbol{\phi}_t^{[\mathbf{Z}]}(\cdot,\cdot)$ but using gradient $\nabla_{\boldsymbol{\Theta}_t^{(k)}}$ instead of $\nabla_{\mathbf{Z}_t^{(k)}}$

**return** $\{(\mathbf{G}_\infty(\mathbf{Z}_T^{(m)}), \boldsymbol{\Theta}_T^{(m)})\}_{m=1}^M$ ▷ See (16) and (17)

# D Experimental Details

## D.1 Gaussian Bayesian Networks

In our experiments, we consider Bayesian networks with Gaussian local conditional distributions of each variable given its parents. For both linear or nonlinear Gaussian BNs, which will be defined presently, the generative model for data simulation as well as the parameter prior used for joint inference are set to standard Gaussian distributions. We fix the observation noise to $\sigma^2 = 0.1$ both during synthetic data generation and inference.

**Linear** Analogous to linear regression, linear Gaussian BNs model the mean value of a given variable as a linear function of its parents:

$$
\begin{aligned}
p(\mathbf{x} \mid \mathbf{G}, \boldsymbol{\Theta}) &= \prod_{i=1}^{d} \mathcal{N}(x_i; \boldsymbol{\theta}_i^\top \mathbf{x}_{\mathrm{pa}(i)}, \sigma^2) \\
\text{or} \quad p(\mathbf{x} \mid \mathbf{G}, \boldsymbol{\Theta}) &= \mathcal{N}\big(\mathbf{x}; (\mathbf{G} \circ \boldsymbol{\Theta})^\top \mathbf{x}, \sigma^2 \mathbf{I}\big)
\end{aligned}
\tag{D.1}
$$

where "$\circ$" denotes element-wise multiplication. When inferring the marginal posterior $p(\mathbf{G} \mid \mathcal{D})$ for linear Gaussian BNs, we follow the predominant choice in the literature and employ the Bayesian Gaussian Equivalent (BGe) marginal likelihood, under which Markov equivalent structures are scored equally [17]. For the Normal-Wishart parameter prior underlying the BGe score, we use the standard effective sample size hyperparameters $\alpha_\mu = 1$ and $\alpha_\omega = d + 2$, respectively. Details on the computation of the BGe score are provided by Kuipers et al. [60].

**Nonlinear** The interaction between variables $\mathbf{x}$ can straightforwardly be extended to be *nonlinear*, for example, using neural networks. Following [27], we consider a dense (fully connected) neural network of the form

$$
\begin{aligned}
&\mathrm{DenseNN}(\,\cdot\,; \boldsymbol{\Theta}) : \mathbb{R}^d \to \mathbb{R} \\
&\mathrm{DenseNN}(\mathbf{u}; \boldsymbol{\Theta}) := \boldsymbol{\Theta}^{(L)} \sigma\Big( \ldots \boldsymbol{\Theta}^{(2)} \sigma\big(\boldsymbol{\Theta}^{(1)} \mathbf{u} + \boldsymbol{\theta}_b^{(1)}\big) + \boldsymbol{\theta}_b^{(2)} \ldots \Big) + \boldsymbol{\theta}_b^{(L)}
\end{aligned}
\tag{D.2}
$$

with weights $\mathbf{\Theta}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$, biases $\boldsymbol{\theta}_b^{(l)} \in \mathbb{R}^{d_l}$, and elementwise activation function $\sigma : \mathbb{R} \to \mathbb{R}$. Zheng et al. [27] have shown that the class of fully connected neural networks in (D.2) that do not depend on the value of $u_k$ is equivalent to the class of fully connected neural networks in (D.2) where the $k$-th column of $\mathbf{\Theta}^{(1)}$ equals zero. This insight allows us to define a nonlinear Gaussian BN parameterized by a fully connected neural network:

$$p(\mathbf{x} \mid \mathbf{G}, \mathbf{\Theta}) = \prod_{i=1}^{d} \mathcal{N}\Big(x_i;\ \text{DenseNN}\big(\mathbf{G}_i^\top \circ \mathbf{x}; \mathbf{\Theta}_i\big),\ \sigma^2\Big) \tag{D.3}$$

As required for a BN, each variable is independent of its non-descendants given its parents. The mask representation in (D.1) and (D.3) is equivalent to the concept of a structural gate used by Kalainathan et al. [61]. Note that the conditional distribution parameters for a nonlinear Gaussian BN of the form in (D.3) contain the weights and biases of $d$ different neural networks, one for each local conditional distribution.

### D.2 Hyperparameters

In all evaluations, DiBS is run for 3,000 iterations and uses the simple linear constraint schedule $\beta(t) = t$. At $t = 0$, the initial latent particles $\{\mathbf{Z}_0\}_{m=1}^M$ and parameter particles $\{\mathbf{\Theta}_0\}_{m=1}^M$ are initialized by sampling from their prior distributions. For the step size schedule $\eta(t)$, we use the adaptive learning rate method RMSProp [62] with learning rate 0.005. We always use 128 samples for Monte Carlo estimation of the gradients. Finally, the bandwidths $\gamma_z, \gamma_\theta$ of the kernel in (15) and the slope of a linear schedule $\alpha(t)$ are chosen in separate held-out instances of each setting in Section 6 and are listed in Table 2. As illustrated by the application in Section 7, the provided hyperparameters can be expected to apply to problem settings of comparable magnitude.

While the latent variable scale $\sigma_z$ can in principle be set arbitrarily, we always set $\sigma_z = 1/\sqrt{k}$ in the prior $p_\beta(\mathbf{Z})$, which makes the norm in the SVGD kernel roughly invariant with respect to the latent dimension $k$, ignoring the acyclicity term. This follows from the fact that $||\mathbf{u}||^2 \sim \text{Gamma}(k/2, 2\sigma_z^2)$ when $u_i \sim \mathcal{N}(0, \sigma_z^2)$, in which case $\mathbb{E}[||\mathbf{u}||^2] = k\sigma_z^2$.

Table 2: DiBS hyperparameter choices for $\alpha(t)$ and bandwidths $\gamma_z, \gamma_\theta$. Here, $\widetilde{\alpha}$ denotes the slope in the linear schedule $\alpha(t) = \widetilde{\alpha} t$.

| Model | $d$ | $\widetilde{\alpha}$ | $\gamma_z$ | $\gamma_\theta$ |
|---|---|---|---|---|
| BGe | 20 | 2 | 2 | - |
| | 50 | 2 | 50 | - |
| Linear Gaussian | 20 | 0.2 | 5 | 500 |
| | 50 | 0.02 | 15 | 1,000 |
| Nonlinear Gaussian | 20 | 0.02 | 5 | 1,000 |
| | 50 | 0.01 | 15 | 2,000 |

### D.3 Baseline Methods

**Structure MCMC (MC$^3$, M-MC$^3$, G-MC$^3$)** Structure MCMC [34, 35] samples in the space of DAGs and adds, deletes, or reverses one edge at a time without violating acyclicity. The acceptance probability of a proposed graph $\mathbf{G}'$ is given by

$$\min\left\{1, \frac{|\mathcal{N}(\mathbf{G}\ )| \cdot p(\mathcal{D} \mid \mathbf{G}')p(\mathbf{G}')}{|\mathcal{N}(\mathbf{G}')| \cdot p(\mathcal{D} \mid \mathbf{G}\ )p(\mathbf{G}\ )}\right\} \tag{D.4}$$

where $\mathbf{G}$ is the current particle and $\mathcal{N}(\mathbf{G})$ the collection of DAGs reachable from $\mathbf{G}$ with one edge change. Following [35], the ratio of neighborhoods is approximated to equal one, which allows for only computing $\mathcal{N}(\mathbf{G}')$ when accepting $\mathbf{G}'$. We implement MC$^3$ using the efficient ancestor matrix trick for finding acyclic proposals [35]. For marginal inference under the BGe marginal likelihood, we compute the Bayes factor in (D.4) by only taking into account the affected node families. For all of MC$^3$, M-MC$^3$, and G-MC$^3$, we specify a burn-in period of 100k samples and then collect a sample every 10k steps, which makes the run time of MC$^3$ and DiBS comparable. Both M-MC$^3$ and G-MC$^3$ use a simple Gaussian random walk proposal for the parameters, respectively, with scale selected to roughly obtain an acceptance rate of 0.2 in each setting [63], when feasible in combination with the graph proposal.

**Nonparametric DAG Bootstrap (BPC, BGES, BPC$^*$, BGES$^*$** The nonparametric DAG bootstrap [40] performs model averaging by bootstrapping the observations $\mathcal{D}$ to yield a collection of synthetic data sets, each of which is used to learn a single graph, here using the GES and PC algorithms [8, 9]. The union of these single graphs approximates the posterior by weighting each graph by its unnormalized posterior probability in (1), analogous to DiBS+. The closed-form maximum likelihood parameter estimate for linear Gaussian BNs with known $\mathbf{G}$ used by BPC$^*$ and B-GES$^*$ is provided by Hauser and Bühlmann [56]. For joint posterior inference, BPC$^*$ and BGES$^*$ use $p(\mathbf{G}, \mathbf{\Theta}, \mathcal{D})$ rather than $p(\mathbf{G}, \mathcal{D})$ for weighting the inferred graphs.

Since the GES and PC algorithms only return essential graphs, we orient a predicted undirected edge correctly when a ground truth edge exists and only count a falsely predicted undirected edge as a single mistake. The held-out likelihood metrics given in (19) are computed for a random consistent DAG extension of the essential graph [64]. Enumerating the possibly exponential number of DAGs in an MEC is infeasible in general [65]. Implementations of the PC and GES algorithms are given by the `CausalDiscoveryToolbox` [66], which is published under an MIT Licence and executes their commonly used R implementations.

### D.4 Computing Resources

Table 3 summarizes the computing time of DiBS in the a superset of the evaluations on BNs with Erdős-Rényi structures in Section 6. The computing time on BNs with scale-free graphs is similar. DiBS is efficiently implemented with `JAX` [67] as the end-to-end nature of our framework allows for just-in-time compilation and automatic differentiation. In our experiments, we use Oracle `BM.Standard.E2.64` machines (64 OCPUs, 512 GB memory), where a single random restart is assigned between 1-4 OCPUs and 2-30 GB memory, depending on the setting. `JAX` [67] is published under an Apache Licence.

Table 3: Compute time of DiBS for the hyperparameters described in Section D.2. Times are the mean $\pm$ SD over 30 random restarts and are given in minutes. $M$ denotes the number of particles. BGe times are slower because the closed-form marginal likelihood requires computing determinants [17, 60].

| Model | $M$ | $d = 10$ | $d = 20$ | $d = 50$ |
|---|---|---|---|---|
| BGe | 10 | $6.6 \pm 0.2$ | $32.6 \pm 0.5$ | $342.3 \pm 11.2$ |
| | 30 | $19.2 \pm 0.3$ | $124.4 \pm 6.0$ | $1{,}121.3 \pm 46.0$ |
| Linear Gaussian | 10 | $8.9 \pm 0.4$ | $19.3 \pm 0.3$ | $99.3 \pm 1.8$ |
| | 30 | $22.0 \pm 0.2$ | $66.8 \pm 0.6$ | $344.1 \pm 24.8$ |
| Nonlinear Gaussian | 10 | $45.5 \pm 2.0$ | $144.1 \pm 5.6$ | $526.4 \pm 19.5$ |
| | 30 | $127.6 \pm 6.3$ | $473.3 \pm 38.1$ | - |

### D.5 Results for Bayesian networks with $d = 50$ variables

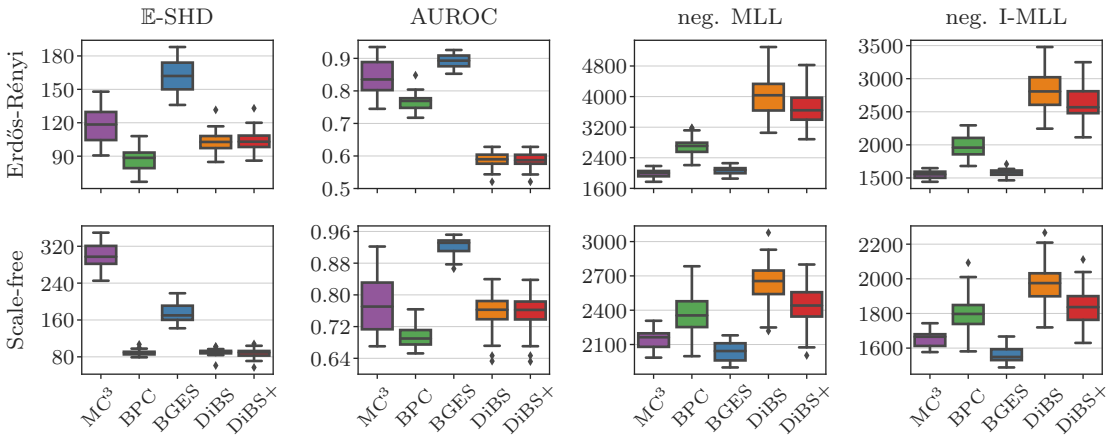



Figure 5: Marginal posterior inference of linear Gaussian BNs with $d = 50$ variables using the BGe marginal likelihood. The first and second rows show the aggregate results for 30 random BNs with Erdős-Rényi and scale-free graphs, respectively. While DiBS and DiBS+ are competitive in the structural $\mathbb{E}$-SHD and AUROC metrics, we find the baselines specifically designed for marginal posterior inference to perform favorably in the likelihood-based metrics. To reach comparable results with DiBS in this high-dimensional setting, the DiBS score function gradient estimator may require additional Monte Carlo samples .

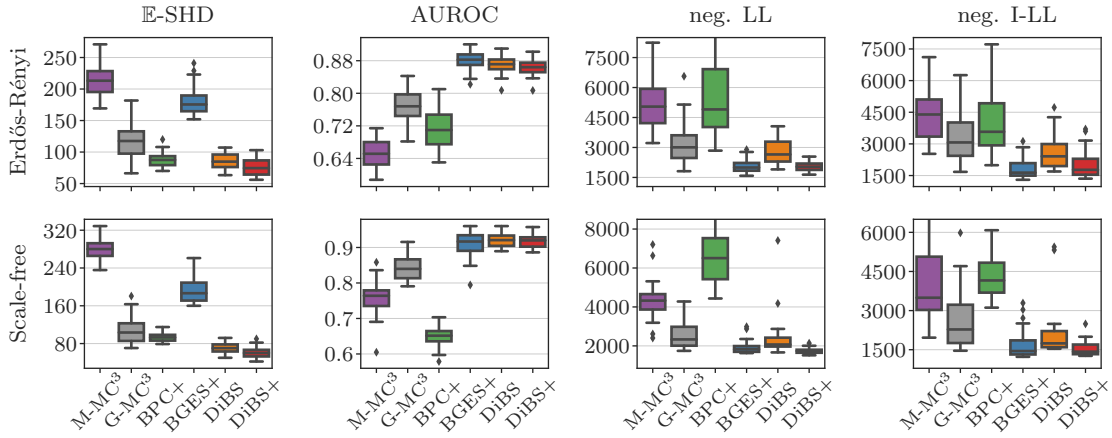



Figure 6: Joint posterior inference of linear Gaussian BNs with $d = 50$ variables. The first and second rows show the aggregate performances for 30 random BNs with Erdős-Rényi and scale-free structures, respectively. Analogous to inference for $d = 20$ variables, DiBS+ outperforms all alternatives to joint posterior inference of the graph and the conditional distribution parameters across the metrics.

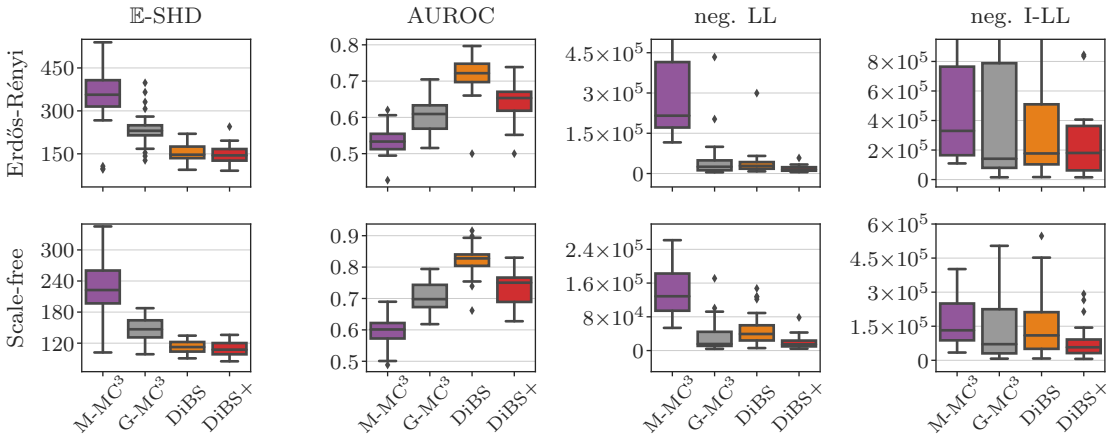



Figure 7: Joint posterior inference of nonlinear Gaussian BNs with $d = 50$ variables, where each local conditional distribution is parameterized by a 2-layer neural network with five hidden nodes. In this setting, the number of overall conditional distribution parameters in a given BN amounts to $|\mathbf{\Theta}| = 13{,}050$ weights and biases. Performances are aggregated for 30 random BNs of each graph type. Here, DiBS only infers 10 particles to make the computation time comparable to MH-MC$^3$ and G-MC$^3$. As for posterior inference of BNs with $d = 20$ variables, DiBS and DiBS+ perform favorably compared to the MC$^3$ baselines.

# E Additional Analyses and Ablation Studies

Having compared DiBS with several alternative approaches to Bayesian structure learning in Section 6, this supplementary section is devoted to a more in depth analysis of some of its properties. This will be done by changing, or leaving out, single design aspects of the algorithm and studying the effect on the previous performance metrics.

As in Section 6, DiBS and its instantiation with SVGD are used interchangeably here, and DiBS+ denotes the weighted particle mixture. Since the performance metrics do not qualitatively differ between inference of Erdős-Rényi and scale-free BN structures in our experiments of Section 6, we only consider the former here. Unless mentioned otherwise, the following experimental setup corresponds to *joint* inference for linear Gaussian BNs with $d = 20$ variables in Section 6.2.

## E.1 Graph Embedding Representation

In Section 4.2, we have proposed to use a generative graph model $p_\alpha(\mathbf{G} \mid \mathbf{Z})$ that is based on the inner product of latent embeddings for each node. In particular, we assume $p_\alpha(g_{ij} = 1 \mid \mathbf{Z}) = \sigma_\alpha(\mathbf{u}_i^\top \mathbf{v}_j)$ with latent variables $\mathbf{Z} = [\mathbf{U}, \mathbf{V}]$. In Figure 8, we contrast this modeling choice with a more trivial variant where $p_\sigma(g_{ij} = 1 \mid \mathbf{Z}) = \sigma_\alpha(\widetilde{\mathbf{z}}_{ij})$, i.e., where single *scalars* rather than inner products

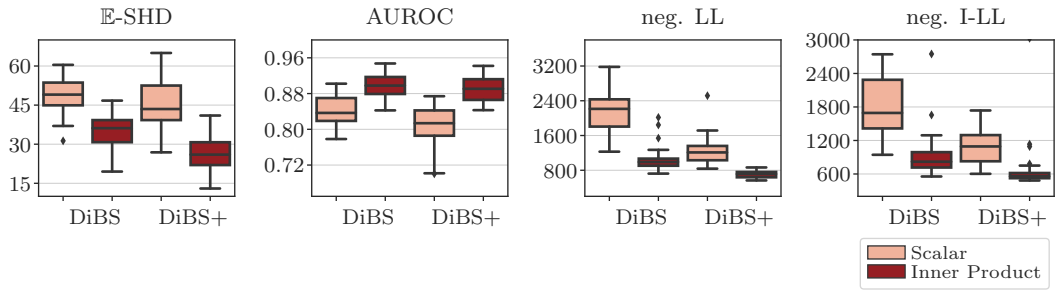



Figure 8: Contrasting the bilinear graph model of Section 4.2 with its more trivial variant, where each latent variable models the edge probabilities directly via the sigmoid. The plots aggregate the results for joint inference of 30 randomly generated linear Gaussian BNs with $d = 20$ variables.

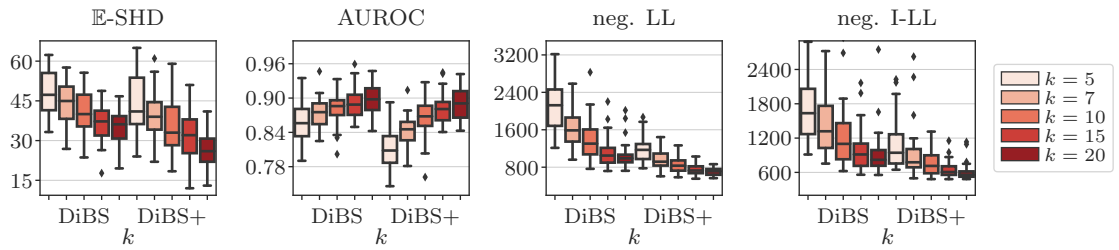



Figure 9: DiBS for joint inference of linear Gaussian BNs with $d = 20$ variables for different sizes of the latent variables $\mathbf{Z} \in \mathbb{R}^{2\times d\times k}$. Lower rank parameterizations of the matrix of edge probabilities balance the tradeoff between computational efficiency and posterior approximation quality.

between latent vectors encode the edge probabilities. Bengio et al. [68] and Ke et al. [31] have previously used the latter with $\alpha = 1$ for causal structure learning.

The comparison in Figure 8 illustrates that adding only the bilinear parameterization in the generative graph model improves performance by a significant margin. We hypothesize that the coupling between edge probabilities results in a smoother density over graphs, which might be less prone to local minima in gradient-based methods such as DiBS.

### E.2 Graph Embedding Dimensionality

Another feature of the inner product representation of graphs is the ability to control the dimensionality of the posterior inference task. As described in Section 5, we generally set $k = d$ for the latent variables $\mathbf{Z} \in \mathbb{R}^{2\times d\times k}$ that parameterizes the graph model $p_\alpha(\mathbf{G} \,|\, \mathbf{Z})$. This leaves the matrix of edge probabilities fully expressible and without a rank constraint. In principle, however, the formulation in (6) allows us to arbitrarily vary $k$. This creates a trade-off between the complexity of the parameterization and the tractability and dimensionality challenges of approximate inference for $p(\mathbf{Z} \,|\, \mathcal{D})$ or $p(\mathbf{Z}, \mathbf{\Theta} \,|\, \mathcal{D})$. Limiting $k < d$ has connections to the theory of low-rank realizations of sign matrices. In the context of structure learning, [69] have extended some of these notions to DAGs and shown that low rank assumptions can be fruitful in combination with the NOTEARS constraint.

We run inference with DiBS for $k \in \{5, 7, 10, 15, 20\}$, leaving all other aspects of the algorithm unchanged. In this context, the corresponding posterior over $\mathbf{Z}$ thus has $\{200, 280, 400, 600, 800\}$ dimensions, respectively. The results in Figure 9 suggest that lower values of $k = 15$, or even $k = 10$, are already able to achieve competitive performances across all metrics. Interestingly, the structural $\mathbb{E}$-SHD metric appears to suffer most from a small loss in complexity.

In this context, one should keep in mind that the bandwidth parameters $\gamma_z$ and $\gamma_h$ were set to achieve good performance with $k = d = 20$. It is conceivable that lower values of $k$ can be even closer to the full-rank variant of DiBS with alternative settings for $\gamma_z$. In addition, a lower-rank DiBS variant

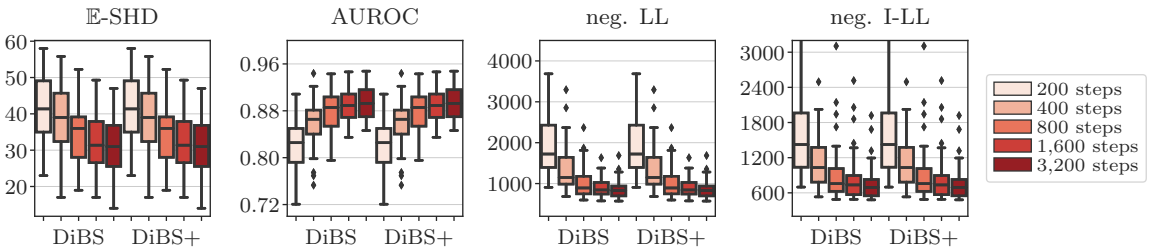



Figure 10: Performance of DiBS and DiBS+ as a function the number of particle transport steps $T$. As previously, the plots aggregate the results for inference of 30 randomly generated linear 20-node Gaussian BNs. The latent variable $\mathbf{Z}$ is specified with its default dimensions $k = d = 20$. After already roughly 1,000 iterations, DiBS and DiBS+ obtain good posterior approximations.

could be particularly promising for inference of very large BNs, where the computational challenges of the full $O(d^2)$ representation might outweigh its benefits in terms of expressibility.

### E.3 Particle Transport Iterations

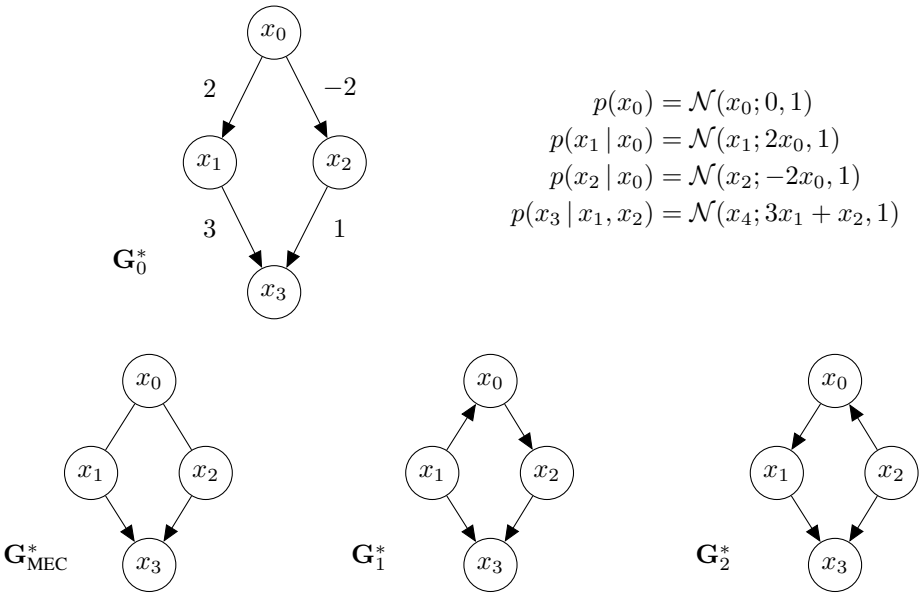
Since DiBS uses Stein variational gradient descent [19] for posterior inference, our method iteratively transports a set of latent graph particles, or latent graph and parameter tuples, for a number of $T$ steps. As the particles are randomly initialized, the approximation quality of SVGD particles (and thus also DiBS particles) improves with the number of steps. We are interested in the degree to which a small number of transport steps already provide a good posterior approximation.

In Figure 10, we show the performance of DiBS as a function of the number of transport steps performed. We find that already a small number of iterations achieves highly competitive results across the board. In addition, the variance of performance in the predictive metrics neg. LL and neg. I-LL decreases monotonically as a function of the performed transforms, whereas the variation in $\mathbb{E}$-SHD remains the same.

### E.4 Uncertainty Quantification Within a Markov Equivalence Class

When performing posterior inference of $p(\mathbf{G} \mid \mathcal{D})$ with the BGe marginal likelihood and a uniform prior $p(\mathbf{G})$, each Markov equivalent structure is scored equally. This might be desirable considering that causal edge directions are often not fully identifiable from purely observational data [70]. As DiBS infers a posterior over DAGs rather than MECs, we are interested in empirically studying the inferred distribution over DAGs within an MEC. In particular, we aim to validate the ability of DiBS to correctly quantify the uncertainty present in non-identifiable edge directions.

To this end, we consider a 4-node example Bayesian network, small enough to allow for the closed-form computation of the ground-truth posterior by exhaustive enumeration of all possible DAGs. This enables us to compute the true single and pairwise edge marginals and contrast them with the approximate posterior marginals inferred by DiBS. The graph structure for this analysis is chosen to contain both an identifiable v-structure and a non-identifiable edge pair. Figure 11 shows the ground truth DAG $\mathbf{G}_0^*$, its linear Gaussian parameters, and the observational model. In addition, Figure 11




$$
\begin{aligned}
p(x_0) &= \mathcal{N}(x_0; 0, 1) \\
p(x_1 \mid x_0) &= \mathcal{N}(x_1; 2x_0, 1) \\
p(x_2 \mid x_0) &= \mathcal{N}(x_2; -2x_0, 1) \\
p(x_3 \mid x_1, x_2) &= \mathcal{N}(x_4; 3x_1 + x_2, 1)
\end{aligned}
\tag{E.1}
$$

Figure 11: Four-node example linear Gaussian Bayesian network. Under the BGe marginal likelihood and a uniform prior, $\mathbf{G}_0^*$, $\mathbf{G}_1^*$, and $\mathbf{G}_2^*$ are scored equally. While the v-structure $x_1 \rightarrow x_3 \leftarrow x_2$ is an identifiable feature of the MEC of $\mathbf{G}_0^*$ and thus present in $\mathbf{G}_{\text{MEC}}^*$, the edge directions of $x_1 - x_0 - x_2$ cannot be distinguished even given infinite observational data.

Table 4: Ground truth and inferred posterior marginals given $N = 100$ observations from the ground truth model in Figure 11. Listed are the probabilities for the non-identifiable edge structure $x_1$ — $x_0$ — $x_2$ (top half) and the identifiable v-structure $x_1 \to x_3 \leftarrow x_2$ (bottom half). Averaged over 30 random particle initilizations, DiBS+ correctly quantifies the confidence and uncertainty in the v-structure and nonidentifable edge pair, respectively.

| | DiBS | DiBS+ | **Ground Truth** |
|---|---|---|---|
| $p(x_1 \to x_0, x_0 \to x_2 \mid \mathcal{D})$ | 0.134 | 0.216 | **0.298** |
| $p(x_1 \to x_0, x_0 \leftarrow x_2 \mid \mathcal{D})$ | 0.201 | 0.037 | **0.052** |
| $p(x_1 \leftarrow x_0, x_0 \to x_2 \mid \mathcal{D})$ | 0.223 | 0.409 | **0.311** |
| $p(x_1 \leftarrow x_0, x_0 \leftarrow x_2 \mid \mathcal{D})$ | 0.103 | 0.298 | **0.297** |
| $p(x_1 \to x_3, x_3 \to x_2 \mid \mathcal{D})$ | 0.157 | 0.013 | **0.017** |
| $p(x_1 \to x_3, x_3 \leftarrow x_2 \mid \mathcal{D})$ | 0.338 | 0.934 | **0.914** |
| $p(x_1 \leftarrow x_3, x_3 \to x_2 \mid \mathcal{D})$ | 0.275 | 0.034 | **0.043** |
| $p(x_1 \leftarrow x_3, x_3 \leftarrow x_2 \mid \mathcal{D})$ | 0.154 | 0.019 | **0.025** |

lists the essential graph $\mathbf{G}^*_{\text{MEC}}$ as well as the two other Markov equivalent DAGs $\mathbf{G}^*_1$ and $\mathbf{G}^*_2$ in the MEC represented by $\mathbf{G}^*_{\text{MEC}}$.

We perform marginal posterior inference with DiBS using the experimental setup and hyperparameters for 20-node linear Gaussian BNs. In this example setting, DiBS employs a uniform prior over graphs, performs only 500 transport iterations, and uses the corresponding default $k = d = 4$. Table 4 shows the ground truth and inferred pairwise edge marginals under the posterior. We find that DiBS+ correctly infers both the uncertainty in edge directions of $x_1$ — $x_0$ — $x_2$ as well as the high-confidence in the presence of the v-structure $x_1 \to x_3 \leftarrow x_2$. In addition, we note that while the unweighted particles of DiBS do not exhibit false confidence in structures that are not present in $\mathbf{G}^*_0$, the inferred degree of uncertainty is too high compared to the ground truth. It is possible that the specific kernel bandwidths in DiBS cause a sufficient coverage of the high-probability regions of the posterior; however, the empirical weighting done in DiBS might be inexact. The DiBS+ variant overcomes this by weighting the particles by their unnormalized posterior probabilities.

These findings are not meant to stand in for a comprehensive theoretical analysis of the implicit inductive biases of DiBS. Rather, we point at important future work to be done in studying the behavior of this new approach—in particular when inferring large-scale Bayesian networks. The experiment should be considered a necessary (but not necessarily sufficient) sanity check.

# F Experimental Details for Application on Protein Signaling Networks

The proteomics data set by Sachs et al. [3] as well as the corresponding consensus graph used in Section 7 are taken as provided by the `CausalDiscoveryToolbox` [66], which is published under an MIT Licence. Because $N$ is large, DiBS uses minibatches of 100 observations to estimate the scores of the posterior. All hyperparameters are chosen exactly as during the synthetic evaluation of linear and nonlinear Gaussian BNs in Table 2, respectively, except that DiBS correspondingly uses $k = d = 11$. We standardize the data and fix the noise level for inference with the explicitly parameterized linear and nonlinear Gaussian networks to $\sigma^2 = 1$. Since the BGe marginal likelihood uses the standard effective sample sizes in the Normal-Wishart prior as described in Appendix D.1, it does not penalize model complexity as much and therefore averages a higher number of edges than the posteriors over Gaussian BNs where the conditional distribution parameters are not marginalized out.