DAGMA: Learning DAGs via M-matrices and a Log-Determinant Acyclicity Characterization

Anonymous Author(s)
Affiliation
Address
email

Abstract

The combinatorial problem of learning directed acyclic graphs (DAGs) from data was recently framed as a purely continuous optimization problem by leveraging a differentiable acyclicity characterization of DAGs based on the trace of a matrix exponential function. Existing acyclicity characterizations are based on the idea that powers of an adjacency matrix contain information about walks and cycles. In this work, we propose a *fundamentally different* acyclicity characterization based on the log-determinant (log-det) function, which leverages the nilpotency property of DAGs. To deal with the inherent asymmetries of a DAG, we relate the domain of our log-det characterization to the set of *M-matrices*, which is a key difference to the classical log-det function defined over the cone of positive definite matrices. Similar to acyclicity functions previously proposed, our characterization is also exact and differentiable. However, when compared to existing characterizations, our log-det function: (1) Is better at detecting large cycles; (2) Has better behaved gradients; and (3) Its runtime is in practice about an order of magnitude faster. From the optimization side, we drop the typically used augmented Lagrangian scheme, and propose DAGMA (*Direct Acyclic Graphs via M-matrices for Acyclicity*), a method that resembles the central path approach for barrier methods. Each point in the central path of DAGMA is a solution to an unconstrained problem regularized by our log-det function, then we show that at the limit of the central path, the solution is guaranteed to be a DAG. Finally, we provide extensive experiments for *linear* and *nonlinear* SEMs, and show that our approach can reach large speed-ups and smaller structural Hamming distance against state-of-the-art methods.

1 Introduction

Structural equation models (SEMs) [36] are a standard modeling tool in several fields such as economics, social sciences, genetics, and causal inference, to name a few. Under this framework, in its general form, the value of each variable in the model is assigned by a general nonlinear, nonparametric functional that takes as input the values of other variables in the model, thus, every SEM can be associated to a graphical model. In particular, we will consider graphical models that are directed acyclic graphs (DAGs).

A long-standing and active research area deals with the problem of learning the graphical structure (DAG) given passively observed data (a.k.a. causal discovery). Computationally, this problem is well-known to be NP-hard in general [6, 8], mainly due to the combinatorial nature of the space of DAGs. In this work, we will follow a score-based approach, a popular learning framework where the

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

goal is to find a DAG that minimizes a *given* score [17, 7]. Recently, Zheng et al. [48] proposed an exact smooth nonconvex characterization of acyclicity which opened the door to solving the originally combinatorial problem via a set of tools that work in the fully continuous regime (e.g., gradient-based methods).

Let $W \in \mathbb{R}^{d \times d}$ be a weighted adjacency matrix of a graph G of d nodes, and let $W \circ W$ denote the Hadamard product. The acyclicity function introduced by Zheng et al. [48] is defined as $h_{\text{expm}}(W) = \text{Tr}(e^{W \circ W}) - d$, where Tr denotes the trace of a matrix, and it was shown that $h_{\text{expm}}(W) = 0$ if and only if W corresponds to a DAG. A follow-up work [46] proposed another acyclicity function, which can be computed slightly faster, defined as $h_{\text{poly}}(W) = \text{Tr}(I + \frac{1}{d} W \circ W) - d$, where I is the identity matrix. It was similarly shown by Yu et al. [46] that $h_{\text{poly}}(W) = 0$ if and only if W corresponds to a DAG. While seemingly different, both h_{expm} and h_{poly} are functions of the form $\text{Tr}(\sum_{k=0}^{d} c_p (W \circ W)^k) - d$ for some $c_k > 0$, as noted by Wei et al. [45]. To the best of our knowledge, all subsequent work [e.g., 49, 21, 50, 30, 29, 20, 33, to name a few] that has built upon the idea of using a continuous acyclicity characterization have used either h_{expm} or h_{poly}, or some acyclicity characterization in the form of a trace of a sum of matrix powers. The latter should come as no surprise, after all, a nonzero diagonal entry of the matrix power $(W \circ W)^k$ reveals the existence of a closed walk of length k in W.

Contributions. In this work, we propose a *fundamentally* different acyclicity function that, as h_{expm} and h_{poly}, is both an exact and a smooth acyclicity characterization but that also possesses several advantages with respect to acyclicity functions of the form of a trace of a sum of matrix powers. Specifically, we make the following set of contributions:

1. We propose a novel acycicity characterization based on the log-determinant (log-det) function (see Theorem 1 and Section 3). In contrast to the classical log-det function defined over the cone of positive definite matrices, we define the domain of our log-det function to be the set of M-matrices due to the inherent asymmetries of DAGs. To our knowledge, we are the first to connect the notion of M-matrices to acyclicity and structure learning for DAGs.
2. We provide a detailed study of the properties of our log-det characterization in Section 3.1. First, we establish the similarities of our log-det function to other existing functions such as h_{expm} and h_{poly}. Second, we formally argue why these functions can be regarded as acyclicity regularizers, similar in spirit to the classical ℓ_1 and ℓ_2 regularizers. Third, we show that our log-det function is an *invex* function, i.e., all stationary points are global minimum, moreover, these stationary points correspond to DAGs.
3. In Section 3.2, we present three arguments as to why our log-det characterization should be preferred over other existing acyclicity functions. Briefly, our log-det function is better at detecting large cycles, has better behaved gradients, and can be computed in about an order of magnitude faster than h_{expm} and h_{poly}.
4. Motivated by the properties of our log-det function, in Section 4, we present DAGMA (*Direct Acyclic Graphs via M-matrices for Acyclicity*), a method that resembles the widely known central path approach for barrier methods [31]. We show that, at the limit of the central path, the solution is guaranteed to be a DAG. In contrast to the NOTEARS method [48], which uses the augmented Lagrangian scheme, each point in the central path of DAGMA is a solution to an unconstrained problem regularized by our log-det function.
5. Finally, in Section 5 and Appendix C, we provide extensive experiments for *linear* and *nonlinear* SEMs, where we show that DAGMA is capable of obtaining DAGs with *better accuracy*, i.e., lower structural Hamming distance (SHD), in a *much faster* way than the state-of-the-art. Specifically, for linear SEMs of moderate size (up to 100 variables), DAGMA obtains on average 10x speedups against NOTEARS [48] and 6x speedups against *tailored* methods to linear SEMs such as GOLEM [30]; moreover, DAGMA attains on average a 75% decrease in SHD w.r.t. NOTEARS and GOLEM. In higher dimensions and in the nonlinear setting, the performance of DAGMA remains consistent across the different set of experiments. We invite the reader to see more details in Section 5 and Appendix C.

Related work The vast majority of methods for learning DAGs can be categorized into two groups: constraint-based algorithms, which rely on conditional independence tests; and score-based algorithms, which focus on finding a DAG that minimizes a given score/loss function. We briefly mention classical constraint-based methods as we follow a score-based approach. [40] developed the PC algorithm, a popular general method that learns the Markov equivalence class. Other algorithms such as [42, 23] are based on local Markov boundary search. Finally, hybrid approaches that combine constraint-based learning with score-based learning, such as [43, 11].

In the line of score-based methods, popular score functions include BDeu [17], BIC [25], and MDL [3]. Works that study linear Gaussian SEMs include [1, 12, 13, 26, 37], and for linear non-Gaussian SEMs [22, 39]. For nonlinear SEMs, we note works on additive models [5, 9, 44], additive noise models [18, 38, 28], generalized linear models [35, 34, 15], and general nonlinear SEMs [27, 14].

More closely related to our work is the line of work built on the nonconvex continuous framework of Zheng et al. [48], such as, [49, 21, 50, 30, 29, 20, 33]. In contrast to our work, all of the aforementioned methods rely on the nonconvex acyclicity functions h_{expm} or h_{poly}, and with the exception of [30], all of these works also use the augmented Lagrangian scheme. Finally, the NoCurl method [47] also departs from using h_{expm}, although no other acyclicity constraint is proposed. Two immediate distinctions can be made to our work. First, we propose a novel acyclicity function based on the log-det function which we show to be prefereable to h_{expm} and h_{poly}. Second, we follow a central path approach where we solve a sequence of unconstrained problems where the score functions is regularized by our log-det function.

Remark 1. *To avoid confunsion, we also note that in the GOLEM method of Ng et al. [30] the score includes a log-determinant function of the form* $\log|\det(I-W)|$ *which stems from the* Gaussian log-likelihood. *While this expression is zero if* W *corresponds to a DAG, it is* ***not*** *an exact acyclicity characterization (i.e.* $\log|\det(I-W)|=0$ *does not imply* W *is a DAG). Moreover, it is not obvious how to extend GOLEM to arbitrary score functions, as their analysis is specific to the Gaussian likelihood function.*

2 Notation and background

Notation. We use $[d]$ to denote the set of integers $\{1\dots d\}$. For a square matrix A, we use $\lambda_i(A)$ to denote its i-th minimum eigenvalue, and use $\rho(A)$ to denote its spectral radius. Also, we use $\mathrm{Tr}(A)$, and $\det(A)$ to denote the trace and determinant of A. For matrices A, B, we let $A\circ B$ represent the element-wise or Hadamard product, moreover, the expression $A\geq B$ is entrywise, i.e., $A_{i,j}\geq B_{i,j}$. Then, we say that a matrix A is nonnegative whenever $A\geq 0$. For a complex number $a+bi$, we let $\Re(a+bi)=a$ denote its real part. We use $\|\cdot\|_p$ to denote the vector ℓ_p-norm, and $\|\cdot\|_{L^p}$ is the L^p-norm on functions. Lastly, $i\to j$ and $i\rightsquigarrow j$ represent an edge from i to j and a directed walk from i to j, respectively.

Let $X=(X_1,\dots,X_d)$ be a d-dimensional random vector. In its general form, a (nonparametric) structural equation model (SEM) consists of a set of equations of the form:

$$X_j = f_j(X, Z_j),\ \forall j\in[d], \tag{1}$$

where each $f_j:\mathbb{R}^{d+1}\to\mathbb{R}$ is a nonlinear nonparametric functional, and Z_j is an exogenous variable representing errors due to omitted factors. We consider the Markovian model, which assumes that each Z_j is an independent random variable. Note that each f_j depends only on a subset of X (i.e., the parents of X_j) and Z_j; nonetheless, to simplify notation we ensure that each f_j is defined on the same space. Then $f=(f_1,\dots,f_d)$ induces a graphical structure, where we focus on acyclic graphs.

For any joint distribution over $Z=(Z_1,\dots,Z_d)$, the functions f_j define a joint distribution $\mathbb{P}(X)$ over the observed data, and a graph $G(f)$ via the dependencies in each f_j. Then, our goal is to learn $G(f)$ given n i.i.d. samples from $\mathbb{P}(X)$. In score-based learning, given a data matrix $\boldsymbol{X}=[\boldsymbol{x}_1,\dots,\boldsymbol{x}_d]\in\mathbb{R}^{n\times d}$, we define a score function $Q(f;\boldsymbol{X})$ to measure the 'quality' of a candidate SEM as follows: $Q(f;\boldsymbol{X})=\sum_{j=1}^{d}\mathsf{loss}(\boldsymbol{x}_j, f_j(\boldsymbol{X}))$, where we adopt the convention that $f_j(\boldsymbol{X})\in\mathbb{R}^n$. Here loss can be any loss function such as least squares $\mathsf{loss}(\boldsymbol{u},\boldsymbol{v})=\frac{1}{n}\sum_{i=1}^{n}(u_i-v_i)^2$ or the

log-likelihood function, often augmented with a penalty such as BIC or ℓ_1. Given the score function Q and a family of functions $\mathcal{F}$, we seek to find the $f \in \mathcal{F}$ that minimizes the score, i.e.,

$$\min_{f \in \mathcal{F}} Q(f; \boldsymbol{X}) \quad \text{subject to} \quad G(f) \in \mathsf{DAGs}. \tag{2}$$

Similar to [49], we consider that each f_j lives in a Sobolev space of square-integrable functions whose derivatives are also square integrable. Then, let $\partial_k f_j$ denote the partial derivative of f_j w.r.t. X_k, it is easy to see that f_j is independent of X_k if and only if $\|\partial_k f_j\|_{L^2} = 0$. With this observation, we construct the matrix $W(f) \in \mathbb{R}^{d \times d}$ with entries $[W(f)]_{i,j} \stackrel{\text{def}}{=} \|\partial_i f_j\|_{L^2}$, which precisely encodes the graphical structure amongst the variables X_j. That is $G(f) \in \mathsf{DAGs} \iff W(f) \in \mathsf{DAGs}$, where W is interpreted as the usual weighted adjacency matrix.

In practice, f is replaced with a flexible family of parametrized functions such as deep neural networks, so that problem (2) is finite dimensional. Finally, note that model (1) includes several models as special cases, e.g., additive noise models, generalized linear models, additive models, polynomial regression, and index models. Previous work has studied the identifiability of several of these models, e.g., [18, 39, 38, 35, 34, 22]. In the sequel, we assume that the model is chosen such that the graph $G(f)$ is uniquely defined from (2).

3 A new characterization of acyclicity via log-determinant and M-matrices

In this section, we present our acyclicity characterization and study its properties. To declutter notation, in this section we simply write W instead of $W(f)$ to denote the *weighted* adjacency matrix of a graph; however, it should be clear that W depends on functions f_j as explained in the previous section.

We develop our characterization by first noting that for any weighted adjacency matrix W, we have that $W \in \mathsf{DAGs}$ if and only if W is a nilpotent matrix, or equivalently, all the eigenvalues of W are zero, i.e., $\lambda_i(W) = 0, \forall i \in [d]$. We then have the following obvious implications:

$$W \in \mathsf{DAGs} \overset{(a)}{\iff} (W \circ W) \in \mathsf{DAGs} \overset{(b)}{\iff} s - \lambda_i(W \circ W) = s, \forall i \in [d], \forall s \in \mathbb{R} \tag{3}$$

$$\implies \prod_{i=1}^{d} s - \lambda_i(W \circ W) = \det(sI - W \circ W) = s^d. \tag{4}$$

Implication (4) can be thought of as a relaxation of acyclicity in the sense that all DAGs satisfy (4), but not all W that satisfy (4) are DAGs. For example, let $s = 1$ and $W \circ W = \left[\begin{smallmatrix} 2 & 0 \\ 0 & 2 \end{smallmatrix}\right]$, then it is clear that $\det(sI - W \circ W) = 1$ and, thus, (4) is satisfied; however, clearly W is not a DAG.

Thus, two immediate questions arise: (i) *Does there exist a domain for W such that (4) $\implies$ (3)?* (ii) *If so, what is the description of such domain?* We answer (i) in the affirmative, and answer (ii) by relating the domain of W to the set of M-matrices, which is defined below.

Definition 1 (M-matrix[1]). *An M-matrix is a matrix $A \in \mathbb{R}^{d \times d}$ of the form $A = sI - B$, where $B \geq 0$ and $s > \rho(B)$.*

M-matrices were introduced by Ostrowski [32] and arise in a variety of areas including input-output analysis in economics, linear complementarity problems in operations research, finite difference methods for partial differential equations, and Markov chains in stochastic processes. To the best of our knowledge, we are the first to connect the notion of M-matrices to graphical model structure learning through an acyclicity characterization.

The following proposition is an immediate consequence of Definition 1:

Proposition 1 (Berman and Plemmons [2]). *Let $A \in \mathbb{R}^{d \times d}$ be an M-matrix, then:*

(i) $\Re(\lambda_i(A)) > 0$, for all $i \in [d]$. *(ii) A^{-1} exists and is nonnegative, i.e., $A^{-1} \geq 0$.*

[1] More precisely, we consider the definition of a non-singular M-matrix, which is sufficient for the purposes of this work.

In the above, item (i) states that the eigenvalues of an M-matrix lie in the open right-half plane. Matrices which satisfy the latter property are also known as positive stable matrices. We thus have that M-matrices are special cases of positive stable matrices. It follows that the determinant of any M-matrix is positive.[2] This fact will be used for defining $h^s_{\text{ldet}}(W)$, our acyclicity characterization given in Theorem 1. Finally, the nonnegativity of the inverse from item (ii) will be used to understand some properties of the gradient of $h^s_{\text{ldet}}(W)$.

We now define the domain over which (4) $\Longrightarrow$ (3). For any $s > 0$, define

$$\mathbb{W}^s = \{W \in \mathbb{R}^{d\times d} \mid s > \rho(W \circ W)\}, \tag{5}$$

i.e. $\mathbb{W}^s$ is the set of real matrices whose entry-wise square given by $W \circ W$ have spectral radius less than s. The following lemma lists some relevant properties of $\mathbb{W}^s$.

Lemma 1. *Let $\mathbb{W}^s$ defined as in (5). Then, for all $s > 0$:*

(i) DAGs $\subset \mathbb{W}^s$. *(ii) $\mathbb{W}^s$ is path-connected.* *(iii) $\mathbb{W}^s \subset \mathbb{W}^t$ for any $t > s$.*

In Lemma 1, item (i) implies that the W we look for is in the interior of $\mathbb{W}^s$; item (ii) indicates that we can find a path from any point in $\mathbb{W}^s$ to any DAG without leaving the set $\mathbb{W}^s$; item (iii) shows that one can vary s to enlarge or shrink the set $\mathbb{W}^s$.

Having defined the domain set $\mathbb{W}^s$, we now define our acyclicity characterization $h^s_{\text{ldet}} : \mathbb{W}^s \to \mathbb{R}$. Recall that item (i) in Proposition 1 implies that applying the logarithm function to the determinant of an M-matrix is always well defined, which motivates our following result.

Theorem 1 (Log-determinant characterization)**.** *Let $s > 0$ and let $h^s_{\text{ldet}} : \mathbb{W}^s \to \mathbb{R}$ be defined as $h^s_{\text{ldet}}(W) \stackrel{\text{def}}{=} -\log\det(sI - W \circ W) + d\log s$. Then, the following holds:*

(i) $h^s_{\text{ldet}}(W) \geq 0, \forall W \in \mathbb{W}^s$, *with* $h^s_{\text{ldet}}(W) = 0$ if and only if W *is a DAG.*

(ii) $\nabla h^s_{\text{ldet}}(W) = 2(sI - W \circ W)^{-\top} \circ W$, *with* $\nabla h^s_{\text{ldet}}(W) = 0$ if and only if W *is a DAG.*

3.1 Properties of $h^s_{\text{ldet}}(W)$

In this section, we list several properties of our acyclicity characterization. The first property we discuss is related to the entries of $\nabla h^s_{\text{ldet}}(W)$.

Lemma 2. *For all $i, j \in [d]$, $[\nabla h^s_{\text{ldet}}(W)]_{i,j} = 0$ if and only if $W_{i,j} = 0$ or there is no directed walk from j to i. Equivalently, $[\nabla h^s_{\text{ldet}}(W)]_{i,j} \neq 0$ if and only if the edge $i \to j$ is part of some cycle in W. Finally, whenever $[\nabla h^s_{\text{ldet}}(W)]_{i,j} \neq 0$, we have that* $\text{sign}([\nabla h^s_{\text{ldet}}(W)]_{i,j}) = \text{sign}(W_{i,j})$.

The lemma above characterizes the nonzero entries and their signs of the gradient of h^s_{ldet}. This property formally offers a *regularizer* perspective for h^s_{ldet}, which we highlight next.

Remark 2 (A regularizer viewpoint)**.** *The function h^s_{ldet} promotes small parameters values in much the same way that the classical ℓ_1 and ℓ_2 regularizers do. In contrast to the latter regularizers, h^s_{ldet} will only shrink the value of a parameter $W_{i,j}$ if and only if the edge (i, j) is part of some cycle in W, as prescribed by Lemma 2.*

Recall that $h_{\text{expm}}(W) = \text{Tr}(e^{W\circ W}) - d$ and $h_{\text{poly}}(W) = \text{Tr}((I + \frac{1}{d}W \circ W)^d) - d$. It was noted by [45] that acyclicity characterizations of the form $\text{Tr}(\sum_{p=1}^d c_p(W \circ W)^d)$ for $c_p > 0$ also have the property in Lemma 2. This implies that h_{expm} and h_{poly} hold the property above and can also be interpreted as acyclicity regularizers. We note that the interesting part here is that h^s_{ldet} holds this property besides being very different in nature to h_{expm} and h_{poly}.

Next, we state two important consequences of Lemma 2. The first consequence is related to the direction of $\nabla h^s_{\text{ldet}}(W)$.

[2] Note that due to asymmetries it is possible for an M-matrix to have complex eigenvalues. However, since we work with matrices with real entries, the complex eigenvalues come in conjugate pairs.

Corollary 1. *At any $W \in \mathbb{W}^s$, the gradient $\nabla h^s_{\text{ldet}}(W)$ points towards the interior of $\mathbb{W}^s$.*

As DAGs are inside $\mathbb{W}^s$ for any $s > 0$, the following is a second consequence of Lemma 2.

Corollary 2. *Starting at any $W^0 \in \mathbb{W}^s$ and for a small enough $\eta > 0$, after a finite number T of gradient descent steps of the form $W^{t+1} = W^t - \eta \nabla h^s_{\text{ldet}}(W^t)$, we have that $W^T \in$ DAGs.*

Note that the corollaries above also hold for h_{expm} and h_{poly} as they are consequences of Lemma 2.

In optimization, the Hessian matrix plays an important role as it contains relevant information about saddle points and local extrema of a function, and is key to Newton-type methods. Another appealing property of $h^s_{\text{ldet}}(W)$ is that it has a Hessian described by a simple closed-form expression.

Lemma 3. *The Hessian of $h^s_{\text{ldet}}(W)$, which resides in $\mathbb{R}^{d^2 \times d^2}$, is given by:*

$$\nabla^2 h^s_{\text{ldet}}(W) = 4\,\text{Diag}(\text{vec}(W))(N \otimes N^\top)\,\text{Diag}(\text{vec}(W^\top))K^{dd} + 2\,\text{Diag}(\text{vec}(N^\top)),$$

where $N = (sI - W \circ W)^{-1}$, and K^{dd} is the $d^2 \times d^2$ commutation matrix such that $K^{dd}\,\text{vec}(A) = \text{vec}(A^\top)$, for any $d \times d$ matrix A.

Here we note that among $h_{\text{expm}}, h_{\text{poly}}$ and h^s_{ldet}, only h^s_{ldet} has a *tractable* expression for the Hessian. Furthermore, note that $\nabla^2 h^s_{\text{ldet}}(W)$ is indexed by vertex pairs so that the entry $[\nabla^2 h^s_{\text{ldet}}(W)]_{(k,l),(p,q)}$ corresponds to the second partial derivative $\frac{\partial^2 h^s_{\text{ldet}}}{\partial W_{k,l} \partial W_{p,q}}$. Using Lemma 3, we can characterize the nonzero entries, and their signs, of the Hessian of h^s_{ldet}.

Corollary 3. *The entries of the Hessian $\nabla^2 h^s_{\text{ldet}}(W)$ are described as follows:*

$$\left[\nabla^2 h^s_{\text{ldet}}(W)\right]_{(k,l),(p,q)} = \begin{cases} 4W_{l,k}N_{k,q}N_{p,l}W_{q,p} & \text{if } (k,l) \neq (p,q), \\ 4(W_{l,k})^2(N_{k,l})^2 + 2N_{k,l} & \text{if } (k,l) = (p,q), \end{cases}$$

where $N = (sI - W \circ W)^{-1}$. Moreover, an off-diagonal entry $[\nabla^2 h^s_{\text{ldet}}(W)]_{(k,l),(p,q)}$ is nonzero if and only if there exists a cycle in W of the form $q \to p \rightsquigarrow l \to k \rightsquigarrow q$, and has a sign equal to $\text{sign}(W_{l,k}W_{q,p})$. Lastly, a diagonal entry $[\nabla^2 h^s_{\text{ldet}}(W)]_{(k,l),(k,l)}$ is nonzero if and only if there exists a directed walk from k to l, and its sign is always positive.

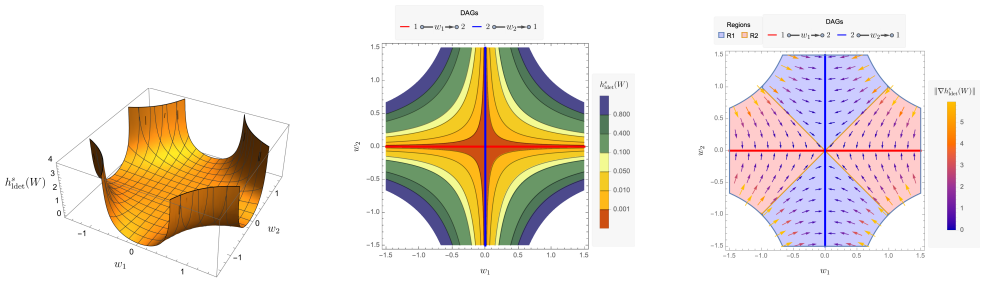
Recall from Theorem 1 that all DAGs attain the minimum value and are critical points of h^s_{ldet}; thus, DAGs are local (and global) minimum of h^s_{ldet} and the Hessian matrix evaluated at a DAG must be positive (semi)definite. Let us corroborate the latter, when W is a DAG, from Corollary 3 we have that the off-diagonal elements of $\nabla^2 h^s_{\text{ldet}}(W)$ are zero, while the diagonal entries are nonnegative. That is, the Hessian is positive semidefinite whenever W is a DAG. This implies that all its stationary points are global minima: Such functions are called *invex* [16, 24].

Corollary 4. *Let $s > 0$. Then, $h^s_{\text{ldet}}(W)$ is an invex function, i.e., all its stationary points are global minima, and these correspond to DAGs.*

We note that even though h_{expm} and h_{poly} are also invex functions, they were not explicitly considered as invex functions before. In fact, it was noted in [48] that DAGs were global minima of h_{expm} but no characterization of its stationary points were given. Wei et al. [45] noted that DAGs were stationary points of h_{expm} and h_{poly} but the notion of invexity was not explicitly stated.

Remark 3 (A dynamical system perspective)**.** *The importance of invexity here is that in the eyes of h^s_{ldet}, all DAGs are the same. That is, DAGs correspond to the set of attractors in h^s_{ldet}, and depending on the initial condition, the system will converge to a different attractor. This offers the following viewpoint for the role of the score function $Q(f; \boldsymbol{X})$, namely, "use the score Q to find a basin of attraction such that the force field of h^s_{ldet} will dictate the trajectory towards a DAG that is equal or close to the ground-truth".*

In Figure 1, we illustrate in a toy example the properties discussed in this subsection.



(a) $h^{s=1}_{\text{ldet}}(W)$ (b) Contours of $h^{s=1}_{\text{ldet}}(W)$ (c) Vector field of $\nabla h^{s=1}_{\text{ldet}}(W)$

Figure 1: Behavior of h^s_{ldet} for $W = \begin{bmatrix} 0 & w_1 \\ w_2 & 0 \end{bmatrix}$. Here clearly W is a DAG whenever one of w_1 or w_2 (or both) are zero. In particular, for (c) we note the perspective given in Remark 3, i.e., starting at any point in region R2 will converge to attractors (DAGs) of the form $X_1 \to X_2$ (red line); while starting at any point in region R1 will converge to attractors (DAGs) of the form $X_2 \to X_1$ (blue line).

3.2 Why the log-determinant regularizer is preferable to existing acyclicity regularizers

In this section, we present three arguments as to why one should use h^s_{ldet} instead of existing functions such as h_{expm} and h_{poly}. We invite the reader to look at Appendix B for additional details.

Argument (i). h^s_{ldet} does not diminish cycles of any length. Let us expand the functions h_{expm} and h_{poly} in their sum of matrix powers form, that is, $h_{\text{expm}}(W) = \sum_{k=0}^{\infty} {}^1\!/_{k!} \operatorname{Tr}((W \circ W)^k) - d$ and $h_{\text{poly}}(W) = \sum_{k=0}^{d} \binom{d}{k}/d^k \operatorname{Tr}((W \circ W)^k) - d$. Recall also that the entry $[(W \circ W)^k]_{i,i}$ represents the sum of weighted walks from node i to node i of length k, where each edge has weight $w^2_{u,v}$. Thus, one can notice that if W has cycles of length k, their contribution to h_{expm} and h_{poly} are *diminished* by ${}^1\!/_{k!}$ and $\binom{d}{k}/d^k$, respectively. Numerically, the latter can be problematic for the following reason: Cycles of length k can go undetected even for small values of d and k. In practice, a value of $h_{\text{expm}}, h_{\text{poly}} \in [10^{-8}, 10^{-5}]$ is typically regarded as zero [45, 48]. Consider a cycle graph of d nodes where each edge weight is $+1$ or -1.[3] The plot in Figure 2 shows how the values of h_{expm} and h_{poly} decay much faster than that of h^s_{ldet}; in fact, at $d = 13$ we already observe $h_{\text{expm}}(W) \approx 10^{-9}$ and $h_{\text{poly}}(W) \approx 10^{-14}$, i.e., cycles of length at least 13 would be numerically undetected by h_{expm} and h_{poly}. In contrast, we observe that the value of the log-det function remains bounded away from zero and is able to detect larger cycles.

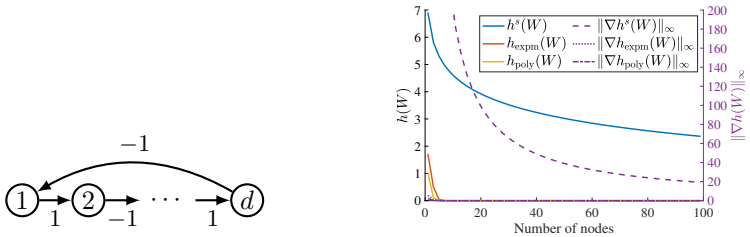


Figure 2: The values of h_{expm} and h_{poly} get very close to zero for a number of nodes as small as ten. In contrast, letting $s = 1.001$, we observe that h^s_{ldet} can stay away from zero even for a cycle graph of 100 nodes. Finally, we note a similar pattern for the entrywise ℓ_∞ norm of the gradients.

At first glance, it might seem difficult to directly compare the value of h^s_{ldet} to h_{expm} and h_{poly}. We next show that when $s = 1$, $h^{s=1}_{\text{ldet}}$ is an upper bound to h_{expm} and h_{poly}.

Lemma 4. *For all $W \in \mathbb{W}^{s=1}$, we have $h_{\text{poly}}(W) \leq h_{\text{expm}}(W) \leq h^{s=1}_{\text{ldet}}(W)$.*

The lemma above shows that in spite of $h^{s=1}_{\text{ldet}}$, h_{expm}, and h_{poly} being exact acyclicity characterizations, $h^{s=1}_{\text{ldet}}$ will attain the largest value.

[3] Note that here the sign of an edge is not important since $W \circ W$ will have all edge weights equal to $+1$.

Argument (ii). h^s_{ldet} has better behaved gradients. Similar to argument (i), we show in Appendix B that h_{expm} and h_{poly} are susceptible to vanishing gradients even when the graph contains cycles (see Figure 2). The following lemma states that the magnitude of each entry of ∇h^s_{ldet} at least as large as the magnitude of the corresponding entry of ∇h_{expm} and ∇h_{poly}, and hence, h_{ldet} has larger gradients to guide optimization.

Lemma 5. $|\nabla h_{\text{poly}}(W)| \leq |\nabla h_{\text{expm}}(W)| \leq |\nabla h^{s=1}_{\text{ldet}}(W)|$.

Argument (iii). Computing h^s_{ldet} and ∇h^s_{ldet} is empirically faster. Even though h^s_{ldet}, h_{expm}, and h_{poly} all three share the same computational complexity of $\mathcal{O}(d^3)$, in practice h^s_{ldet} can be computed in about an order of magnitude faster than h_{expm} and h_{poly}. In Figure 3, we compare the runtimes of h^s_{ldet}, h_{expm} and h_{poly} for randomly generated matrices, where we observe that computing h^s_{ldet} can be 10x faster than h_{expm} and h_{poly}. See Appendix B for further details.

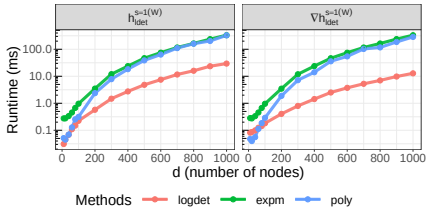


Figure 3: For each d, 30 matrices were randomly sampled from a standard Gaussian distribution.

4 Optimization

In the previous section we argued why the log-det function should be preferred in practice. Let f_θ denote a model with parameters θ for the functions f_j in (1), e.g., neural networks as in [49]. In this section we turn to the problem of minimizing a given score function $Q(f_\theta; \boldsymbol{X})$ constrained to $h^s_{\text{ldet}}(W(\theta)) = 0$. That is, we aim to solve:

$$\min_\theta Q(f_\theta; \boldsymbol{X}) + \beta_1 \|\theta\|_1 \quad \text{subject to} \;\; h^s_{\text{ldet}}(W(\theta)) = 0, \tag{6}$$

where we include the ℓ_1 regularizer to promote sparse solutions.

Since the inception of the purely continuous framework for learning DAGs in [48], almost all follow-up work continued using the NOTEARS approach of [48, 49], that is, using h_{expm}, the augmented Lagrangian (AML) scheme to tackle problem (6), and L-BFGS-B [31] for solving the sequence of unconstrained problems. Motivated by the properties of h^s_{ldet} given in 3.1, we propose a simpler scheme named DAGMA, based on solving a sequence of unconstrained problems in which h^s_{ldet} is simply seen as a regularizer. DAGMA resembles the central path approach of barrier methods [4, 31], or the classical path-following approach for solving lasso problems [e.g. 10]. Our method is given in Algorithm 1.

Algorithm 1 DAGMA

Require: Data matrix $\boldsymbol{X}$, initial central path coefficient $\mu^{(0)}$ (e.g., 1), decay factor $\alpha \in (0, 1)$ (e.g., 0.1), ℓ_1 parameter $\beta_1 > 0$ (e.g., 0.01), log-det parameter $s > 0$ (e.g., 1), number of iterations T.

1: Initialize $\theta^{(0)}$ so that $W(\theta^{(0)}) \in \mathbb{W}^s$.
2: **for** $t = 0, 1, 2, \ldots T-1$ **do**
3: Starting at $\theta^{(t)}$, solve $\theta^{(t+1)} = \arg\min_\theta \mu^{(t)}(Q(f_\theta; \boldsymbol{X}) + \beta_1 \|\theta\|_1) + h^s_{\text{ldet}}(W(\theta))$
4: Set $\mu^{(t+1)} = \alpha \mu^{(t)}$

Ensure: $W(\theta^{(T)})$

The following lemma states that DAGMA will return a DAG at the limit of the central path. This is a critical distinction against existing methods, all of which rely on some type of post-processing (e.g. thresholding) to ensure that the solution is a DAG.

Lemma 6. *Algorithm 1 is guaranteed to return a DAG whenever* $\mu^{(t)} \to 0$.

Practical considerations. Similar as in barrier methods, where it is required to start at the interior of the feasibility region, in Algorithm 1 we require that the initial point $W(\theta^{(0)})$ be inside $\mathbb{W}^s$. This is very easy to achieve since the zero matrix is in the interior of $\mathbb{W}^s$ for any $s > 0$; therefore, in our experiments we simply set $\theta^{(0)} = 0$. Regarding the choice of s, in principle s could take any value greater than zero since DAGs are inside $\mathbb{W}^s$ for any $s > 0$. However, we find that letting s be equal or close to 1 is generally easier to optimize than setting s closer to zero, the reason being that for smaller values of s the volume of $\mathbb{W}^s$ is smaller and will require much smaller learning rates to stay inside $\mathbb{W}^s$, hence affecting convergence. Finally, we do not specify how to solve line 3 in Algorithm 1, this is because we leave the door open for different solvers to be used. For our experiments in the next section, we solve line 3 by using a first-order method with the ADAM optimizer [19], which works remarkably well as shown in our experiments. It remains as future work to exploit the Hessian structure of h_{ldet} given in Lemma 3 for second-order methods.

5 Experiments

We compare our method against GES [7], PC [41], NOTEARS [48], GOLEM [30]. Since the accuracy of certain methods were significantly lower than other methods, we only report the result against competitive methods in the main text; full results for all settings and methods can be found in Appendix C. Our experimental setting is similar to [48, 49], we briefly explain Figure 4, and defer the full details to Appendix C.

In Figure 4, for the linear SEM setting, we observe that DAGMA obtains about 30x speedups over NOTEARS (see e.g., $d = 800$), and about 5x speedups against GOLEM, which is a *tailored* method for linear Gaussian SEMs. In terms of SHD, we note that DAGMA obtains improvement of **at least** a factor of 2 against NOTEARS and a factor of 6 against GOLEM. For the nonlinear case: In the generalized linear model (logistic), we observe that DAGMA attains 4x speedups against NOTEARS and an improvement on SHD of at least a factor of 3. For the general nonlinear model (neural networks), we observe that DAGMA is comparable in SHD to the NONLINEAR NOTEARS [49] but obtains about 3x speedups.

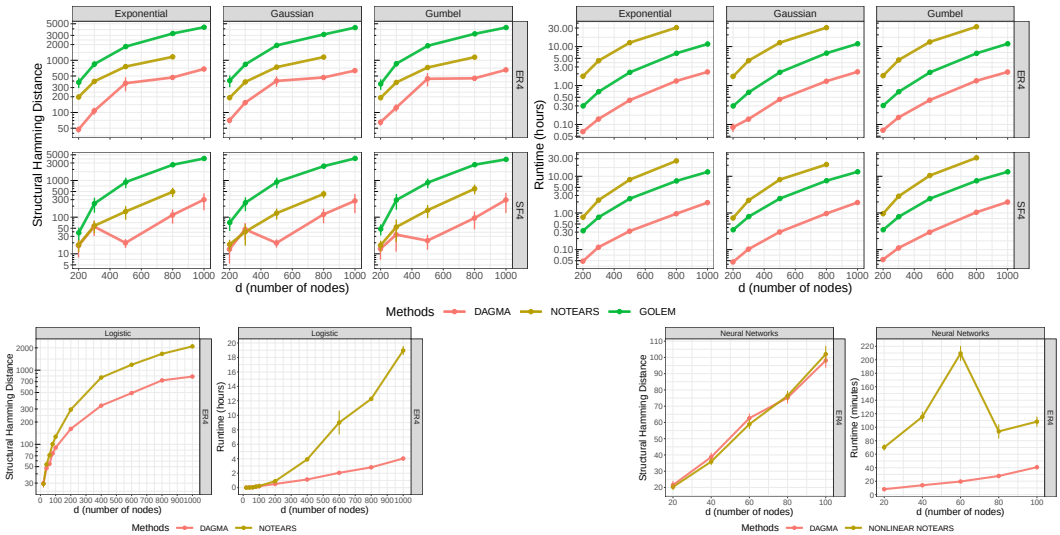


Figure 4: First row: experiments on linear SEMs. Second row: experiments on nonlinear SEMs. For each d, graphs have in expectation $4d$ number of edges. Each point in the plot is estimated over 10 repetitions, where error bars are the standard error. Wall time was 36 hours.

References

[1] Aragam, B. and Zhou, Q. [2015], 'Concave Penalized Estimation of Sparse GaussianBayesian Networks', *Journal of Machine Learning Research* .

[2] Berman, A. and Plemmons, R. J. [1994], *Nonnegative Matrices in the Mathematical Sciences*, Classics in Applied Mathematics, Society for Industrial and Applied Mathematics.

[3] Bouckaert, R. R. [1993], Probabilistic network construction using the minimum description length principle, *in* 'European conference on symbolic and quantitative approaches to reasoning and uncertainty', Springer, pp. 41–48.

[4] Boyd, S. and Vandenberghe, L. [2004], *Convex Optimization*, Cambridge University Press.

[5] Bühlmann, P., Peters, J. and Ernest, J. [2014], 'Cam: Causal additive models, high-dimensional order search and penalized regression', *The Annals of Statistics* **42**(6), 2526–2556.

[6] Chickering, D. M. [1996], Learning bayesian networks is np-complete, *in* 'Learning from data', Springer, pp. 121–130.

[7] Chickering, D. M. [2003], 'Optimal structure identification with greedy search', *JMLR* **3**, 507–554.

[8] Chickering, D. M., Heckerman, D. and Meek, C. [2004], 'Large-sample learning of Bayesian networks is NP-hard', *Journal of Machine Learning Research* **5**, 1287–1330.

[9] Ernest, J., Rothenhäusler, D. and Bühlmann, P. [2016], 'Causal inference in partially linear structural equation models: identifiability and estimation', *arXiv preprint arXiv:1607.05980* .

[10] Friedman, J., Hastie, T., Höfling, H. and Tibshirani, R. [2007], 'Pathwise coordinate optimization', *The annals of applied statistics* **1**(2), 302–332.

[11] Gámez, J. A., Mateo, J. L. and Puerta, J. M. [2011], 'Learning bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood', *Data Mining and Knowledge Discovery* **22**(1), 106–148.

[12] Ghoshal, A. and Honorio, J. [2017], Learning identifiable gaussian bayesian networks in polynomial time and sample complexity, *in* 'Proceedings of the 31st International Conference on Neural Information Processing Systems', pp. 6460–6469.

[13] Ghoshal, A. and Honorio, J. [2018], Learning linear structural equation models in polynomial time and sample complexity, *in* 'Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics', Vol. 84 of *Proceedings of Machine Learning Research*, PMLR, pp. 1466–1475.

[14] Goudet, O., Kalainathan, D., Caillou, P., Guyon, I., Lopez-Paz, D. and Sebag, M. [2018], Learning functional causal models with generative neural networks, *in* 'Explainable and interpretable models in computer vision and machine learning', Springer, pp. 39–80.

[15] Gu, J., Fu, F. and Zhou, Q. [2019], 'Penalized estimation of directed acyclic graphs from discrete data', *Statistics and Computing* **29**(1), 161–176.

[16] Hanson, M. A. [1981], 'On sufficiency of the Kuhn-Tucker conditions', *Journal of Mathematical Analysis and Applications* **80**(2), 545–550.

[17] Heckerman, D., Geiger, D. and Chickering, D. M. [1995], 'Learning bayesian networks: The combination of knowledge and statistical data', *Machine learning* **20**(3), 197–243.

[18] Hoyer, P., Janzing, D., Mooij, J. M., Peters, J. and Schölkopf, B. [2008], 'Nonlinear causal discovery with additive noise models', *Advances in neural information processing systems* **21**.

[19] Kingma, D. P. and Ba, J. [2014], 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980* .

[20] Kyono, T., Zhang, Y. and van der Schaar, M. [2020], 'Castle: Regularization via auxiliary causal graph discovery', *Advances in Neural Information Processing Systems* **33**, 1501–1512.

[21] Lachapelle, S., Brouillard, P., Deleu, T. and Lacoste-Julien, S. [2020], Gradient-based neural dag learning, *in* 'International Conference on Learning Representations'.

[22] Loh, P.-L. and Buhlmann, P. [2014], 'High-Dimensional Learning of Linear Causal Networks via Inverse Covariance Estimation', *Journal of Machine Learning Research* .

[23] Margaritis, D. and Thrun, S. [1999], Bayesian network induction via local neighborhoods, *in* 'Proceedings of the 12th International Conference on Neural Information Processing Systems', pp. 505–511.

[24] Martin, D. H. [1985], 'The essence of invexity', *Journal of Optimization Theory and Applications* **47**(1), 65–76.

[25] Maxwell Chickering, D. and Heckerman, D. [1997], 'Efficient approximations for the marginal likelihood of bayesian networks with hidden variables', *Machine learning* **29**(2), 181–212.

[26] Meinshausen, N. and Bühlmann, P. [2006], 'High-dimensional graphs and variable selection with the Lasso', *The Annals of Statistics* **34**(3).

[27] Monti, R. P., Zhang, K. and Hyvärinen, A. [2020], Causal discovery with general non-linear relationships using non-linear ica, *in* 'Uncertainty in Artificial Intelligence', PMLR, pp. 186–195.

[28] Mooij, J. M., Peters, J., Janzing, D., Zscheischler, J. and Schölkopf, B. [2016], 'Distinguishing cause from effect using observational data: methods and benchmarks', *The Journal of Machine Learning Research* **17**(1), 1103–1204.

[29] Moraffah, R., Moraffah, B., Karami, M., Raglin, A. and Liu, H. [2020], 'Causal adversarial network for learning conditional and interventional distributions', *arXiv preprint arXiv:2008.11376* .

[30] Ng, I., Ghassami, A. and Zhang, K. [2020], 'On the Role of Sparsity and DAG Constraints for Learning Linear DAGs', *arXiv* .

[31] Nocedal, J. and Wright, S. J. [2006], *Numerical Optimization*, Springer Series in Operations Research and Financial Engineering.

[32] Ostrowski, A. [1937], 'Über die determinanten mit überwiegender Hauptdiagonale', *Commentarii Mathematici Helvetici* **10**(1), 69–96.

[33] Pamfil, R., Sriwattanaworachai, N., Desai, S., Pilgerstorfer, P., Georgatzis, K., Beaumont, P. and Aragam, B. [2020], Dynotears: Structure learning from time-series data, *in* 'International Conference on Artificial Intelligence and Statistics', PMLR, pp. 1595–1605.

[34] Park, G. and Park, S. [2019], 'High-dimensional poisson structural equation model learning via \ell_1-regularized regression.', *J. Mach. Learn. Res.* **20**, 95–1.

[35] Park, G. and Raskutti, G. [2017], 'Learning quadratic variance function (qvf) dag models via overdispersion scoring (ods).', *J. Mach. Learn. Res.* **18**, 224–1.

[36] Pearl, J. [2009], *CAUSALITY: Models, Reasoning, and Inference*, 2nd edn, Cambridge University Press.

[37] Peters, J. and Bühlmann, P. [2014], 'Identifiability of gaussian structural equation models with equal error variances', *Biometrika* **101**(1), 219–228.

[38] Peters, J., Mooij, J. M., Janzing, D. and Bernhard [2014], 'Causal Discovery with Continuous Additive Noise Models', *Journal of Machine Learning Research* .

[39] Shimizu, S., Hoyer, P. O., Hyvärinen, A., Kerminen, A. and Jordan, M. [2006], 'A linear non-gaussian acyclic model for causal discovery.', *Journal of Machine Learning Research* **7**(10).

[40] Spirtes, P. and Glymour, C. [1991], 'An algorithm for fast recovery of sparse causal graphs', *Social Science Computer Review* **9**(1), 62–72.

[41] Spirtes, P., Glymour, C. and Scheines, R. [2000], *Causation, Prediction, and Search*, The MIT Press.

[42] Tsamardinos, I., Aliferis, C. F., Statnikov, A. R. and Statnikov, E. [2003], Algorithms for large scale markov blanket discovery, *in* 'FLAIRS conference', Vol. 2, pp. 376–380.

[43] Tsamardinos, I., Brown, L. E. and Aliferis, C. F. [2006], 'The max-min hill-climbing Bayesian network structure learning algorithm', *Machine Learning* **65**(1), 31–78.

[44] Voorman, A., Shojaie, A. and Witten, D. [2014], 'Graph estimation with joint additive models', *Biometrika* **101**(1), 85–101.

[45] Wei, D., Gao, T. and Yu, Y. [2020], 'DAGs with No Fears: A Closer Look at Continuous Optimization for Learning Bayesian Networks', *arXiv* .

[46] Yu, Y., Chen, J., Gao, T. and Yu, M. [2019], 'DAG-GNN: DAG Structure Learning with Graph Neural Networks', *arXiv* .

[47] Yu, Y., Gao, T., Yin, N. and Ji, Q. [2021], 'DAGs with No Curl: An Efficient DAG Structure Learning Approach', *arXiv* .

[48] Zheng, X., Aragam, B., Ravikumar, P. and Xing, E. P. [2018], 'DAGs with NO TEARS: Continuous Optimization for Structure Learning', *arXiv* .

[49] Zheng, X., Dan, C., Aragam, B., Ravikumar, P. and Xing, E. P. [2020], 'Learning Sparse Nonparametric DAGs', *International Conference on Artificial Intelligence and Statistics* .

[50] Zhu, S., Ng, I. and Chen, Z. [2020], Causal discovery with reinforcement learning, *in* 'International Conference on Learning Representations'.

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]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (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]
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]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (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]
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? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]