Novel Ordering-based Approaches for Causal Structure Learning in the Presence of Unobserved Variables

Ehsan Mokhtarian,[1] **Mohammadsadegh Khorasani,**[1] **Jalal Etesami,**[1] **Negar Kiyavash**[1,2]

[1]School of Computer and Communication Sciences EPFL, Lausanne, Switzerland
[2]College of Management of Technology EPFL, Lausanne, Switzerland
{ehsan.mokhtarian, sadegh.khorasani, seyed.etesami, negar.kiyavash}@epfl.ch

Abstract

We propose ordering-based approaches for learning the maximal ancestral graph (MAG) of a structural equation model (SEM) up to its Markov equivalence class (MEC) in the presence of unobserved variables. Existing ordering-based methods in the literature recover a graph through learning a causal order (c-order). We advocate for a novel order called removable order (r-order) as they are advantageous over c-orders for structure learning. This is because r-orders are the minimizers of an appropriately defined optimization problem that could be either solved exactly (using a reinforcement learning approach) or approximately (using a hill-climbing search). Moreover, the r-orders (unlike c-orders) are invariant among all the graphs in a MEC and include c-orders as a subset. Given that set of r-orders is often significantly larger than the set of c-orders, it is easier for the optimization problem to find an r-order instead of a c-order. We evaluate the performance and the scalability of our proposed approaches on both real-world and randomly generated networks.

Introduction

A causal graph is a probabilistic graphical model that represents conditional independencies (CIs) among a set of observed variables $\mathbf{V}$ with a joint distribution $P_{\mathbf{V}}$. When all the variables in the system are observed (i.e., causal sufficiency holds), a causal graph is commonly modeled with a directed acyclic graph (DAG), $\mathcal{G}$. It is well-known that from mere observational distribution $P_{\mathbf{V}}$, graph $\mathcal{G}$ can only be learned up to its Markov equivalence class (MEC) (Spirtes et al. 2000; Pearl 2009). Therefore, the problem of causal structure learning (aka causal discovery) from observational distribution in the absence of latent variables refers to identifying the MEC of $\mathcal{G}$ using a finite set of samples from $P_{\mathbf{V}}$ and has important applications in many areas such as biology (Sachs et al. 2005), advertisements (Bottou et al. 2013), social science (Russo 2010), etc.

There are three main classes of algorithms for causal structure learning: constraint-based, score-based, and hybrid methods. Constraint-based methods use the available data from $P_{\mathbf{V}}$ to test for CI relations in the distribution, from which they learn the MEC of $\mathcal{G}$. Score-based methods define a score function (e.g., regularized likelihood function or Bayesian information criterion (BIC)) over the space of graphs and search for a structure that maximizes the score function. Hybrid methods combine the strength of both constraint-based and score-based methods to improve score-based algorithms by applying constraint-based techniques.

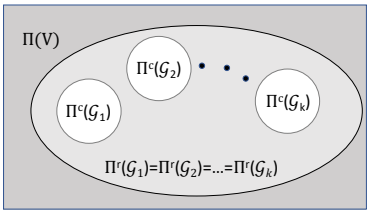


Figure 1: In this figure, $\{\mathcal{G}_1, \cdots, \mathcal{G}_k\}$ denotes a set of Markov equivalent DAGs. $\Pi(\mathbf{V})$ denotes the set of orders over $\mathbf{V}$, which is the search space of ordering-based methods. $\Pi^c(\mathcal{G}_i)$ denotes the set of c-orders of $\mathcal{G}_i$, the target space of existing ordering-based methods in the literature. $\Pi^r(\mathcal{G}_i)$ denotes the set of r-orders of $\mathcal{G}_i$, which is the target space of the proposed methods in this paper.

Under causal sufficiency assumption, the search space of most of the score-based algorithms is the space of DAGs, which contain $2^{\Omega(n^2)}$ members when there are n variables in the system. In score-based methods, a variety of search strategies are proposed to solve the maximization problem. Teyssier and Koller (2005) introduced the first ordering-based search strategy to solve the score-based optimization. The search space of such ordering-based methods is the space of orders over the vertices of the DAG, which includes $2^{\mathcal{O}(n \log(n))}$ orders. Note that the space of orders is significantly smaller than the space of DAGs. Ordering-based methods divide the learning task into two stages. In the first stage, they use the available data to find a causal order (c-order) over the set vertices of $\mathcal{G}$. They use the learned order in the second stage to identify the MEC of $\mathcal{G}$ (Zhu, Ng, and Chen 2020; Larranaga et al. 1996; Teyssier and Koller 2005; Friedman and Koller 2003).

All the aforementioned ordering-based approaches for causal discovery require causal sufficiency. In practice, presence of unobserved variables is more the norm rather than the exception. In such cases, instead of a DAG, graphical models such as maximal ancestral graph (MAG) and induc-

ing path graph (IPG) are developed in the literature to represent a causal model (Richardson and Spirtes 2002). We introduce a novel type of order for MAGs, called removable order (in short, r-order), and argue that r-orders are advantageous over c-orders for structure learning. For one, as r-orders are defined for MAGs (as opposed to DAGs), they can be used to design algorithms for causal graph structure discovery in the absence of causal sufficiency. Moreover, even in the absence of latent variables, r-orders are better suited for learning the MEC. This is because, as depicted in Figure 1, r-orders include c-orders. As a consequence, the problem of searching for an r-order is easier than finding a c-order as the search space remains the same, but the set of feasible solutions is larger. Our main contributions are summarized as follows.

1. We introduce a novel type of order for MAGs, called r-order, which is invariant among the MAGs in a MEC (Proposition 2). In the case of DAGs, we note that this property does not hold for c-orders as they are mutually exclusive across the graphs in a MEC (Figure 1).
2. We propose ordering-based approaches for identifying the MEC of a MAG using r-orders. In particular, our methods do not require causal sufficiency. Furthermore, we show that the problem of finding an r-order can be cast as a minimization problem and prove that r-orders are the unique minimizers of this problem (Theorem 2).
3. We show that our minimization problem can be formulated with appropriately defined costs as a reinforcement learning problem. Accordingly, any reinforcement learning algorithm can be applied to find a solution to our problem. Additionally, we propose a hill-climbing search algorithm to approximate the solution to the optimization problem of our interest.

Related Work

Under causal sufficiency assumption, several causal structure discovery approaches have been proposed in the literature: constraint-based (Zhang et al. 2011, 2017; Spirtes et al. 2000; Margaritis and Thrun 1999; Pellet and Elisseeff 2008b; Mokhtarian et al. 2021; Zhang et al. 2019; Tsamardinos, Aliferis, and Statnikov 2003; Sun et al. 2007; Mokhtarian et al. 2022), score-based (Nandy, Hauser, and Maathuis 2018; Zheng et al. 2018; Bottou et al. 2013; Yu et al. 2019; Yang et al. 2022), and hybrid (Tsamardinos, Brown, and Aliferis 2006; Nandy, Hauser, and Maathuis 2018; Gámez, Mateo, and Puerta 2011; Schulte et al. 2010; Schmidt et al. 2007; Alonso-Barba et al. 2013). Some score-based methods formulate the structure learning problem as a smooth continuous optimization and exploit gradient descent to solve it (Yu et al. 2019; Lachapelle et al. 2020; Ng et al. 2022; Zheng et al. 2020). Zhu, Ng, and Chen (2020) and Wang et al. (2021) formulated the optimization problem as a reinforcement learning problem, where the score function is defined over DAGs and orders, respectively. Furthermore, among score-based approaches, various ordering-based methods have been proposed that exploit different search strategies to find a c-order (Zhu, Ng, and Chen 2020; Larranaga et al. 1996; Teyssier and Koller 2005; Friedman and Koller 2003). Ordering-based approaches are also useful for a variety of causal discovery-related tasks (Rolland et al. 2022; Ghoshal, Bello, and Honorio 2019). All of these approaches are heuristics and provide no guarantees of finding a correct c-order.

There are a few papers in the literature that do not require causal sufficiency. FCI (Spirtes et al. 2000) is a constraint-based algorithm that starts with the skeleton of the graph learned by PC algorithm and then performs more CI tests to learn a MAG up to its MEC. RFCI (Colombo et al. 2012), FCI+ (Claassen, Mooij, and Heskes 2013), and MBCS*(Pellet and Elisseeff 2008a) are three modifications of FCI. L-MARVEL (Akbari et al. 2021) is a recursive algorithm that iteratively eliminates specific variables and learns the skeleton of a MAG. M3HC (Tsirlis et al. 2018) is a hybrid method that can learn a MAG up to its MEC. To the best of our knowledge, the only other work in the literature that uses an ordering-based approach for causal discovery in MAGs (i.e., in the presence of latent variable) is GSPo which proposes a greedy algorithm that is only consistent as long as there are no latent variables in the system (the graph is a DAG) (Raskutti and Uhler 2018), but there are no theoretical guarantees in case of MAGs (Bernstein et al. 2020).

Preliminary and Problem Description

Throughout the paper, we denote random variables by capital letters (e.g., X) and sets of variables by bold letters (e.g., $\mathbf{X}$). A *mixed graph* (MG) is a graph $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$, where $\mathbf{V}$ is a set of vertices, $\mathbf{E}_1$ is a set of directed edges, i.e., $\mathbf{E}_1 \subseteq \{(X, Y) \mid X, Y \in \mathbf{V}\}$, and $\mathbf{E}_2$ is a set of bidirected edges, i.e., $\mathbf{E}_2 \subseteq \{\{X, Y\} \mid X, Y \in \mathbf{V}\}$. For a subset $\mathbf{Z} \subseteq \mathbf{V}$, MG $\mathcal{G}[\mathbf{Z}] = (\mathbf{Z}, \mathbf{E}_1^{\mathbf{Z}}, \mathbf{E}_2^{\mathbf{Z}})$ denotes the induced subgraph of $\mathcal{G}$ over $\mathbf{Z}$, that is $\mathbf{E}_1^{\mathbf{Z}} = \{(X, Y) \in \mathbf{E}_1 \mid X, Y \in \mathbf{Z}\}$ and $\mathbf{E}_2^{\mathbf{Z}} = \{\{X, Y\} \in \mathbf{E}_2 \mid X, Y \in \mathbf{Z}\}$. For each directed edge (X, Y) in $\mathbf{E}_1$, we say X is a *parent* of Y and Y is a *child* of X. Further, we say X and Y are neighbors if a directed or undirected edge exists between them in $\mathcal{G}$. The *skeleton* of $\mathcal{G}$ is the undirected graph obtained by removing the directions of the edges of $\mathcal{G}$. A path $(X_1, X_2, \cdots, X_k)$ in $\mathcal{G}$ is called a *directed path* from X_1 to X_k if $(X_i, X_{i+1}) \in \mathbf{E}_1$ for all $1 \leq i < k$. If a directed path exists from X to Y, X is called an *ancestor* of Y. We denote the set of parents, children, and ancestors of X in $\mathcal{G}$ by $Pa_{\mathcal{G}}(X)$, $Ch_{\mathcal{G}}(X)$, and $Anc_{\mathcal{G}}(X)$, respectively. We also apply these definitions disjunctively to sets of variables, e.g., $Anc_{\mathcal{G}}(\mathbf{X}) = \bigcup_{X \in \mathbf{X}} Anc_{\mathcal{G}}(X)$. A non-endpoint vertex X_i on a path $(X_1, X_2, \cdots, X_k)$ is called a *collider*, if one of the following situations arises.

$$X_{i-1} \to X_i \leftarrow X_{i+1}, \quad X_{i-1} \leftrightarrow X_i \leftarrow X_{i+1},$$
$$X_{i-1} \to X_i \leftrightarrow X_{i+1}, \quad X_{i-1} \leftrightarrow X_i \leftrightarrow X_{i+1}.$$

A path $\mathcal{P} = (X, W_1, \cdots, W_k, Y)$ between two distinct variables X and Y is said to be *blocked* by a set $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ in $\mathcal{G}$ if there exists $1 \leq i \leq k$ such that (i) W_i is a collider on $\mathcal{P}$ and $W_i \notin Anc_{\mathcal{G}}(\mathbf{Z} \cup \{X, Y\})$, or (ii) W_i is not a collider on $\mathcal{P}$ and $W_i \in \mathbf{Z}$. We say $\mathbf{Z}$ *m-separates* X and Y in $\mathcal{G}$ and denote it by $(X \perp\!\!\!\perp Y | \mathbf{Z})_{\mathcal{G}}$ if all the paths in $\mathcal{G}$ between X and Y are blocked by $\mathbf{Z}$.

A *directed cycle* exists in an MG $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$ when there exists $X, Y \in \mathbf{V}$ such that $(X, Y) \in \mathbf{E}_1$ and

$Y \in Anc_{\mathcal{G}}(X)$. Similarly, an *almost directed cycle* exists in $\mathcal{G}$ when there exists $X, Y \in \mathbf{V}$ such that $\{X, Y\} \in \mathbf{E}_2$ and $Y \in Anc_{\mathcal{G}}(X)$. An MG with no directed cycles or almost-directed cycles is said to be *ancestral*. An ancestral MG is called *maximal* if every pair of non-neighbor vertices are m-separable, i.e., there exists a set of vertices that m-separates them. An MG is called a *maximal ancestral graph* (MAG) if it is both ancestral and maximal. A MAG with no bidirected edges is called a *directed acyclic graph (DAG)*. Two MAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ are *Markov equivalent* if they impose the same set of m-separations, i.e., $(X \perp\!\!\!\perp Y|\mathbf{Z})_{\mathcal{G}_1} \iff (X \perp\!\!\!\perp Y|\mathbf{Z})_{\mathcal{G}_2}$. We denote by $[\mathcal{G}]$ the Markov equivalence class (MEC) of MAG $\mathcal{G}$, i.e., the set of Markov equivalent MAGs of $\mathcal{G}$. Moreover, if $\mathcal{G}$ is a DAG, we denote by $[\mathcal{G}]^{\text{DAG}}$ the set of Markov equivalent DAGs of $\mathcal{G}$.

Suppose $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$ is a MAG and let $P_{\mathbf{V}}$ denote a joint distribution over $\mathbf{V}$. For two distinct variables X and Y in $\mathbf{V}$ and a subset $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$, if $P(X, Y \mid \mathbf{Z}) = P(X \mid \mathbf{Z})P(Y \mid \mathbf{Z})$, then X and Y are said be conditionally independent given $\mathbf{Z}$ and it is denoted by $(X \perp\!\!\!\perp Y|\mathbf{Z})_{P_{\mathbf{V}}}$. A Conditional Independence (CI) test refers to detecting whether $(X \perp\!\!\!\perp Y|\mathbf{Z})_{P_{\mathbf{V}}}$. MAG $\mathcal{G}$ satisfies *faithfulness* w.r.t. (is faithful to) $P_{\mathbf{V}}$ if m-separations in $\mathcal{G}$ is equivalent to CIs in $P_{\mathbf{V}}$, i.e., $(X \perp\!\!\!\perp Y|\mathbf{Z})_{\mathcal{G}} \iff (X \perp\!\!\!\perp Y|\mathbf{Z})_{P_{\mathbf{V}}}$.

Consider a set of variables $\mathbf{V} \cup \mathbf{U}$, where $\mathbf{V}$ and $\mathbf{U}$ denote the set of observed and unobserved variables, respectively. In a *structural equation model* (SEM), each variable $X \in \mathbf{V} \cup \mathbf{U}$ is generated as $X = f_X(Pa(X), \epsilon_X)$, where f_X is a deterministic function, $Pa(X) \subseteq \mathbf{V} \cup \mathbf{U} \setminus \{X\}$, and ϵ_X is the exogenous variable corresponding to X with an additional assumption that the exogenous variables are jointly independent (Pearl 2009). The causal graph of an acyclic SEM is a directed acyclic graph (DAG) over $\mathbf{V} \cup \mathbf{U}$ obtained by adding a directed edge from each variable in $Pa(X)$ to X, for $X \in \mathbf{V} \cup \mathbf{U}$. The *latent projection* of this DAG over $\mathbf{V}$ is a MAG over $\mathbf{V}$ such that for $X, Y \in \mathbf{V}$ and $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$, we have

$$(X \perp\!\!\!\perp Y|\mathbf{Z})_{\text{DAG}} \iff (X \perp\!\!\!\perp Y|\mathbf{Z})_{\text{MAG}}.$$

For more details regarding the latent projection, please refer to Verma and Pearl (1991); Akbari et al. (2021). Let us denote by $\mathcal{G}$ the resulting MAG over $\mathbf{V}$. In this paper, we assume faithfulness, i.e.,

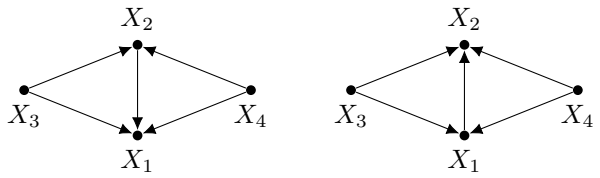
$$(X \perp\!\!\!\perp Y|\mathbf{Z})_{P_{\mathbf{V}}} \iff (X \perp\!\!\!\perp Y|\mathbf{Z})_{\mathcal{G}}.$$

The assumption of causal sufficiency refers to assuming that $\mathbf{U} = \varnothing$. Note that with causal sufficiency, $\mathcal{G}$ is a DAG.

The problem of causal discovery refers to identifying the MEC of $\mathcal{G}$ using samples from the observational distribution. We propose three methods for identifying MEC $[\mathcal{G}]$ using a finite set of samples from $P_{\mathbf{V}}$. It is noteworthy that our proposed methods do not require causal sufficiency.

Ordering-based Methods: Removable Orders vs Causal Orders

In this section, we first define *orders* and *c-orders*. Then, we introduce our novel order, *r-order*, and provide some of its appealing properties.



(a) DAG $\mathcal{G}_1$, $\Pi^c(\mathcal{G}_1) = \{(X_1, X_2, X_3, X_4), (X_1, X_2, X_4, X_3)\}$.

(b) DAG $\mathcal{G}_2$, $\Pi^c(\mathcal{G}_2) = \{(X_2, X_1, X_3, X_4), (X_2, X_1, X_4, X_3)\}$.

Figure 2: Two Markov equivalent DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ that form a MEC together and their disjoint sets of c-orders. In this example, any order over $\mathbf{V} = \{X_1, X_2, X_3, X_4\}$ is an r-order, i.e., $\Pi^r(\mathcal{G}_1) = \Pi^r(\mathcal{G}_2) = \Pi(\mathbf{V})$. Note that $|\Pi^r(\mathcal{G}_1)| = |\Pi^r(\mathcal{G}_2)| = 24 > 2 = |\Pi^c(\mathcal{G}_1)| = |\Pi^c(\mathcal{G}_2)|$.

Definition 1 (order)**.** *An n-tuple $(X_1, \cdots, X_n)$ is called an* order *over a set $\mathbf{V}$ if $|\mathbf{V}| = n$ and $\mathbf{V} = \{X_1, \cdots, X_n\}$. We denote by $\Pi(\mathbf{V})$, the set of all orders over $\mathbf{V}$.*

Definition 2 (c-order)**.** *An order $(X_1, \cdots, X_n) \in \Pi(\mathbf{V})$ is called a causal order (in short c-order[1]) of a DAG $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \varnothing)$ if $i > j$ for each $(X_i, X_j) \in \mathbf{E}_1$. We denote by $\Pi^c(\mathcal{G})$ the set of c-orders of $\mathcal{G}$.*

Consider the two DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ and their sets of c-orders depicted in Figure 2. In this case, $\mathcal{G}_1$ and $\mathcal{G}_2$ are Markov equivalent and together form a MEC. Furthermore, $\Pi^c(\mathcal{G}_1)$ and $\Pi^c(\mathcal{G}_2)$ are disjoint and each contain 2 orders.

As mentioned earlier, nearly all existing ordering-based methods assume causal sufficiency. These methods divide the learning task into two stages. In the first stage, they search in $\Pi(\mathbf{V})$ (which we refer to as the *search space*) to find an order in $\Pi^c(\mathcal{G})$ (which we refer to as the *target space*). In the second stage, they use the discovered order to identify MEC $[\mathcal{G}]^{\text{DAG}}$.

Next, we show that different DAGs in a MEC have a disjoint set of c-orders. In other words, c-orders are not invariant among the DAGs in a MEC. Note that detailed proofs appear in the appendix.

Proposition 1. *Let $\mathcal{G}$ denotes a DAG with MEC $[\mathcal{G}]^{DAG} = \{\mathcal{G}_1, ..., \mathcal{G}_k\}$. For any two distinct DAGs $\mathcal{G}_i$ and $\mathcal{G}_j$ in $[\mathcal{G}]^{DAG}$, we have $\Pi^c(\mathcal{G}_i) \cap \Pi^c(\mathcal{G}_j) = \varnothing$.*

Removable Orders: In this section, we propose a novel set of orders over the vertices of a MAG, called *removable order (in short r-order)*, and show that r-orders are advantageous for structure learning. First, we review the notion of a removable variable in a MAG, which was recently proposed in the structure learning literature (Mokhtarian et al. 2021, 2022; Akbari et al. 2021).

Definition 3 (removable variable)**.** *Suppose $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$ is a MAG. A variable $X \in \mathbf{V}$ is called removable in $\mathcal{G}$ if $\mathcal{G}$ and $\mathcal{G}[\mathbf{V}']$ impose the same set of m-separation relations over $\mathbf{V}'$, where $\mathbf{V}' = \mathbf{V} \setminus \{X\}$. That is, for any variables $Y, W \in \mathbf{V}'$ and $\mathbf{Z} \subseteq \mathbf{V}' \setminus \{Y, W\}$,*

$$(Y \perp\!\!\!\perp W|\mathbf{Z})_{\mathcal{G}} \iff (Y \perp\!\!\!\perp W|\mathbf{Z})_{\mathcal{G}[\mathbf{V}']}.$$

[1]Note that this definition is in the opposite direction than usually c-order is defined in the literature.

Below, we introduce the notion of *r-order*.

Definition 4 (r-order)**.** *An order $\pi = (X_1, \cdots, X_n)$ over set $\mathbf{V}$ is called a removable order (r-order) of a MAG $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$ if X_i is a removable variable in $\mathcal{G}[\{X_i, \cdots, X_n\}]$ for each $1 \leq i \leq n$. We denote by $\Pi^r(\mathcal{G})$ the set of r-orders of $\mathcal{G}$.*

Back to the example in Figure 2 where $\mathcal{G}_1$ and $\mathcal{G}_2$ are two Markov equivalent DAGs. In this case, any order over the set of vertices is an r-order for both $\mathcal{G}_1$ and $\mathcal{G}_2$. Hence, each graph has 24 r-orders.

In general, all MAGs in a MEC have the same set of r-orders. Furthermore, in DAGs, r-orders include all the c-orders as subsets (See Figure 1). The following propositions formalize these assertions.

Proposition 2. *If $\mathcal{G}_1$ and $\mathcal{G}_2$ are two Markov equivalent MAGs, then $\Pi^r(\mathcal{G}_1) = \Pi^r(\mathcal{G}_2)$.*

Proposition 3. *For any DAG $\mathcal{G}$, we have $\Pi^c(\mathcal{G}) \subseteq \Pi^r(\mathcal{G})$.*

In light of the above propositions, we can summarize some clear advantages of r-orders as follows:

(i) Implication of Proposition 2 is that, unlike c-ordering-based methods, which fail to find a c-order consistent with all the DAGs within a MEC (Proposition 1), r-ordering-based methods can find an order which is an r-order for all the MAGs in its corresponding MEC.

(ii) Proposition 3 implies that in DAGs, the space of r-orders is, in general, bigger than the space of c-orders. Hence, the target space of an r-ordering-based method is larger than the target space of a c-ordering-based method. For instance, in Figure 2, a c-ordering-based method must find one of the two c-orders of either $\mathcal{G}_1$ or $\mathcal{G}_2$, while an r-ordering-based method can find any of the 24 r-orders in $\Pi^r(\mathcal{G}_1) = \Pi^r(\mathcal{G}_2)$.

(iii) Since r-orders are defined for MAGs (instead of DAGs), they could be used in ordering-based structure learning approaches without requiring causal sufficiency.

Learning an R-order

In this section, we describe our approach for learning an r-order of the MAGs in $[\mathcal{G}]$. Recall that all MAGs in $[\mathcal{G}]$ have the same set of r-orders. We first propose an algorithm that constructs an undirected graph $\mathcal{G}^\pi$ corresponding to an arbitrary order $\pi \in \Pi(\mathbf{V})$. Subsequently, we assign a cost to an order π based on the constructed graph $\mathcal{G}^\pi$, which is simply the number of edges in $\mathcal{G}^\pi$, and show that finding an r-order for $[\mathcal{G}]$ can be cast as an optimization problem with the aforementioned cost. Then, we propose three algorithms to solve the optimization problem.

Learning an Undirected Graph From an Order

Algorithm 1 iteratively constructs an undirected graph $\mathcal{G}^\pi = (\mathbf{V}, \mathbf{E}^\pi)$ from a given order $\pi \in \Pi(\mathbf{V})$. The inputs of Algorithm 1 are an order π over $\mathbf{V}$ and observational data *Data*$(\mathbf{V})$ sampled from a joint distribution $P_{\mathbf{V}}$. The algorithm initializes $\mathbf{V}_1$ with $\mathbf{V}$ and $\mathbf{E}^\pi$ with the empty set in lines 2 and 3, respectively. Then in lines 4-8, it iteratively selects a variable X_t according to the given order π (line 5) and calls function *FindNeighbors* in line 6 to learn a set $\mathbf{N}_{X_t} \subseteq \mathbf{V}_t \setminus \{X_t\}$. Then, the algorithm adds undirected edges to $\mathcal{G}^\pi$ to connect X_t and its discovered neighbors $\mathbf{N}_{X_t}$ (line 7). Finally, it updates $\mathbf{V}_{t+1}$ by removing X_t from $\mathbf{V}_t$ (line 8) and repeats the process.

Algorithm 1: Learning $\mathcal{G}^\pi$.

1: **Function LearnGPi** (π, *Data*$(\mathbf{V})$)
2: $\mathbf{V}_1 \leftarrow \mathbf{V}$, $\mathbf{E}^\pi \leftarrow \varnothing$
3: **for** $t = 1$ to $|\mathbf{V}| - 1$ **do**
4: $\quad X_t \leftarrow \pi(t)$
5: $\quad \mathbf{N}_{X_t} \leftarrow$ **FindNeighbors**(X_t, *Data*$(\mathbf{V}_t)$)
6: $\quad$ Add undirected edges between X_t and the variables in $\mathbf{N}_{X_t}$ to $\mathbf{E}^\pi$.
7: $\quad \mathbf{V}_{t+1} \leftarrow \mathbf{V}_t \setminus \{X\}$
8: **Return** $\mathcal{G}^\pi = (\mathbf{V}, \mathbf{E}^\pi)$

The output of function *FindNeighbors*, i.e., $\mathbf{N}_{X_t}$, is the set of variables in $\mathbf{V}_t$ that are not m-separable from X_t using the variables in $\mathbf{V}_t$. Hence, if MAG $\mathcal{G}[\mathbf{V}_t]$ is faithful to $P_{\mathbf{V}_t}$, then N_{X_t} would be the set of neighbors of X_t among the variables in $\mathbf{V}_t$.[2] However, since π is arbitrary, $\mathcal{G}[\mathbf{V}_t]$ is not necessarily faithful to $P_{\mathbf{V}_t}$ and therefore, $\mathbf{N}_{X_t}$ can include some vertices that are not neighbors of X_t. There exist several constraint-based algorithms in the literature that are designed to verify whether two given variables are m-separable (Spirtes et al. 2000; Pellet and Elisseeff 2008a; Colombo et al. 2012; Akbari et al. 2021). Accordingly, *FindNeighbors* can use any of such algorithms. Please note that unlike the methods in (Mokhtarian et al. 2022; Akbari et al. 2021) where removable variables are discovered in each iteration, Algorithm 1 selects variables according to the given order π (line 4).

Cost of an Order

Suppose $\mathcal{G}$ is faithful to $P_{\mathbf{V}}$. It is shown in (Akbari et al. 2021) that omitting a removable variable does not violate faithfulness in the remaining graph. Hence, due to the definition of r-order, if $\pi \in \Pi^r(\mathcal{G})$, then after each iteration t, MAG $\mathcal{G}[\mathbf{V}_t]$ remains faithful to $P_{\mathbf{V}_t}$. The next result shows that Algorithm 1 constructs the skeleton of $\mathcal{G}$ correctly if and only if π is an r-order of $\mathcal{G}$.

Theorem 1. *Suppose $\mathcal{G} = (\mathbf{V}, \mathbf{E}_1, \mathbf{E}_2)$ is a MAG and is faithful to $P_{\mathbf{V}}$, and let Data$(\mathbf{V})$ be a collection of i.i.d. samples from $P_{\mathbf{V}}$ with a sufficient number of samples to recover the CI relations in $P_{\mathbf{V}}$. Then, we have the following.*

- *The output of Algorithm 1 (i.e., $\mathcal{G}^\pi$) equals the skeleton of $\mathcal{G}$ if and only if $\pi \in \Pi^r(\mathcal{G})$.*
- *For an arbitrary order π over set $\mathbf{V}$, $\mathcal{G}^\pi$ is a supergraph of the skeleton of $\mathcal{G}$.*

Theorem 1 implies that if $\pi \in \Pi^r(\mathcal{G})$, then $\mathcal{G}^\pi$ is the skeleton of $\mathcal{G}$, and if $\pi \notin \Pi^r(\mathcal{G})$, then $\mathcal{G}^\pi$ is a supergraph of the skeleton of $\mathcal{G}$ that contains at least one extra edge. Therefore, by defining the cost of an order in $\Pi(\mathbf{V})$ equal to the number of edges in $\mathcal{G}^\pi$, r-orders will be the minimizers, which implies the following.

[2] Note that non-neighbor variables in any MAG are m-separable.

Algorithm 2: Hill-climbing approach (ROL_{HC})

1: **Input:** $Data(\mathbf{V})$, *maxSwap*, *maxIter*
2: Initialize $\pi \in \Pi(\mathbf{V})$ as discussed in Appendix A.1
3: $C_\pi \leftarrow$ **ComputeCost**$(\pi, Data(\mathbf{V}))$
4: **for** 1 to *maxIter* **do**
5: Denote π by $(X_1, \cdots, X_n)$
6: $\Pi^{\text{new}} \leftarrow \{(X_1, \cdots X_{a-1}, X_b, X_{a+1}, \cdots, X_{b-1}, X_a, X_{b+1}, \cdots, X_n) | 1 \leq b - a \leq \textit{maxSwap}\}$
7: **for** $\pi_{\text{new}} \in \Pi^{\text{new}}$ **do**
8: $C_{\pi_{\text{new}}} \leftarrow$ **ComputeCost**$(\pi_{\text{new}}, Data(\mathbf{V}))$
9: **if** $C_{\pi_{\text{new}}} < C_\pi$ **then**
10: $\pi \leftarrow \pi_{\text{new}}, C_\pi \leftarrow C_{\pi_{\text{new}}}$
11: **Break** go to line 5
12: **Return** π

1: **Function ComputeCost** (π, $Data(\mathbf{V})$)
2: $\mathcal{G}^\pi = (\mathbf{V}, \mathbf{E}^\pi) \leftarrow$ **LearnGPi**$(\pi, Data(\mathbf{V}))$
3: **Return** $|\mathbf{E}^\pi|$

Theorem 2 (Consistency of the score function)**.** *Any solution of the optimization problem*

$$\underset{\pi \in \Pi(\mathbf{V})}{\arg\min} |\mathbf{E}^\pi|, \tag{1}$$

is an r-order, i.e., a member of $\Pi^r(\mathcal{G})$*. Conversely, every member of* $\Pi^r(\mathcal{G})$ *is also a solution of* (1)*.*

Next, we propose both exact and heuristic algorithms for solving the above optimization problem.

Algorithmic Approaches to Finding an R-order

In this section, we propose three algorithms for solving the optimization problem in (1).

Hill-climbing Approach (ROL_{HC}) In Algorithm 2, we propose a hill-climbing approach, called ROL_{HC}[3] for finding an r-order. In general, the output of Algorithm 2 is a suboptimal solution to (1) as it takes an initial order π and gradually modifies it to another order with less cost, but it is not guaranteed to find a minimizer of (1) by taking such greedy approach. Nevertheless, this algorithm is suitable for practice as it is scalable to large graphs and also achieves superior accuracy compared to the state-of-the-art methods (please refer to the experiment section).

Inputs to Algorithm 2 are the observational data $Data(\mathbf{V})$ and two parameters *maxIter* and *maxSwap*. *maxIter* denotes the maximum number of iterations before the algorithm terminates, and *maxSwap* is an upper bound on the index difference of two variables that can get swapped in an iteration (line 6). Initial order π in line 2 can be any arbitrary order, but selecting it cleverly will improve the performance of the algorithm. In Appendix A.1, we describe several ideas for selecting the initial order, such as initialization using the output of other approaches. The algorithm computes the cost of π (denoted by C_π) in line 3 by calling subroutine *ComputeCost* which itself calls subroutine *LearnGPi* (See Algorithm 1). The remainder of the algorithm (lines 4-12) updates π iteratively, *maxIter* number of times. It updates the current order $\pi = (X_1, \cdots, X_n)$ as follows: first, it constructs a set of orders $\Pi^{\text{new}} \subseteq \Pi(\mathbf{V})$ from π by swapping any two variables X_a and X_b in π as long as $1 \leq b - a \leq$ *maxSwap*. Next, for each $\pi_{\text{new}} \in \Pi^{\text{new}}$, it computes the cost of π_{new} and if it has a lower cost compared to the current order, the algorithm replaces π by that order and repeats the process.

In Appendix A.2, we present a slightly modified version of Algorithm 2, called Algorithm 4, which does not compute the cost of an order as in line 8 of Algorithm 2 but rather uses the information of C_π for computing the cost of the new permutation $C_{\pi_{\text{new}}}$ (using Algorithm 3 also presented in the Appendix A.2). By doing so, Algorithm 4 significantly reduces the computational complexity.

Exact Reinforcement Learning Approach (ROL_{VI}) In this section, we show that the optimization problem in (1) can be cast as a reinforcement learning (RL) problem.

Recall the process of recovering $\mathcal{G}^\pi$ from a given order π in Algorithm 1. This process can be interpreted as a Markov decision process (MDP) in which the iteration index t denotes time, the set of variables $\mathbf{V}$ represents the action space, and the state space is the set of all subsets of $\mathbf{V}$. More precisely, let s_t and a_t denote the state and the action of the MDP at time/iteration t, respectively. In our setting, s_t is the remaining variables at time t, i.e., $s_t = \mathbf{V}_t$, and action a_t is the variable that is getting removed from $\mathbf{V}_t$ in that iteration, i.e., $a_t = X_t$. Accordingly, the state transition due to action a_t is $s_{t+1} = \mathbf{V}_t \setminus \{a_t\}$. The immediate reward of selecting action a_t at state s_t will be the negative of the instant cost that is the number of discovered neighbors for a_t by *FindNeighbors* in line 6 of Algorithm 1, i.e.,

$$r(s_t, a_t) = |\textbf{FindNeighbors}(a_t, Data(s_t))| = -|\mathbf{N}_{a_t}|.$$

Since the form of the function $r(s_t, a_t)$ is not known, this is an RL as opposed to a classic MDP setting. We denote by π_θ, a deterministic policy parameterized by θ. That is, for any state s_t, $a_t = \pi_\theta(s_t)$ is an action in s_t. Accordingly, we modify Algorithm 1 as follows: it gets a policy π_θ instead of a permutation π as input. Furthermore, it selects X_t in line 5 as $X_t = \pi_\theta(\mathbf{V}_t)$. Given a policy π_θ and the initial state $s_1 = \mathbf{V}$, a trajectory $\tau = (s_1, a_1, s_2, a_2, \cdots, s_{n-1}, a_{n-1})$ denotes the sequence of states and actions selected by π_θ. The cumulative reward of this trajectory, denoted by $R(\tau_\theta)$, is the sum of the immediate rewards.

$$R(\tau_\theta) = \sum_{t=1}^{n-1} r(s_t, a_t) = -\sum_{t=1}^{n-1} |\mathbf{N}_{a_t}|.$$

Hence, if we denote the output of this modified algorithm by $\mathcal{G}^\theta = (\mathbf{V}, \mathbf{E}^\theta)$, then $R(\tau_\theta) = -|\mathbf{E}^\theta|$. In this case, any algorithm that finds the optimal policy for RL, such as Value iteration (Sutton and Barto 2018) or Q-learning (Watkins and Dayan 1992) can be used to find a minimum-cost policy π_θ.

Remark 1. *According to the introduced RL setting, value-iteration can be used to find the optimal policy with the time complexity of* $\mathcal{O}(n2^n)$*, which is much less than* $\mathcal{O}(n!)$ *for naively iterating over all orders.*

[3] ROL stands for **R**-**O**rder **L**earning.

Approximate Reinforcement Learning Approach (ROL_{PG}) Although any algorithm suited for RL is capable of finding an optimal deterministic policy for us, the complexity does not scale well as the graph size. Therefore, we advocate searching for a stochastic policy that increases the exploration during the training of an RL algorithm. As discussed earlier, we could exploit stochastic policies parameterized by neural networks to further improve scalability. However, this could come at the price of approximating the optimal solution instead of finding the exact one. In the stochastic setting, an action a_t is selected according to a distribution over the remaining variables, i.e., $a_t \sim P_\theta(\cdot|s_t = \mathbf{V}_t)$, where θ denotes the parameters of the policy (e.g., the weights used in training of a neural network). In this case, the objective of the algorithm is to minimize the expected total number of edges learned by policy $P_\theta(\cdot|s_t = \mathbf{V}_t)$, i.e.,

$$\arg\max_{\theta} \mathbb{E}_{\tau_\theta \sim P_\theta}\big[-|\mathbf{E}^\theta|\big], \tag{2}$$

where the expectation is taken w.r.t. randomness of the stochastic policy. Many algorithms have been developed in the literature for finding stochastic policies and solving (2). Some examples include Vanilla Policy Gradient (VPG) (Williams 1992), REINFORCE (Sutton et al. 1999),and Deep Q-Networks (DQN) (Mnih et al. 2013).

Second Stage: Identifying the MEC

In the previous section, we proposed three algorithms for finding an r-order $\pi^* \in \Pi^r(\mathbf{V})$. Recall that our goal in this paper is to identify the MEC $[\mathcal{G}]$ using the available data from $P_{\mathbf{V}}$. To this end, we can recover the skeleton of the MAGs in $[\mathcal{G}]$ by calling Algorithm 1 with input π^*. Moreover, since FindNeighbors finds a separating set for non-neighbor variables of X_t in $\mathcal{G}[\mathbf{V}_t]$, we can modify Algorithm 1 to further return a set of separating sets for all the non-neighbor variables in MAG $\mathcal{G}$. This information suffices to identify $[\mathcal{G}]$ by maximally orienting the edges using the complete set of orientation rules introduced in Zhang (2008).

Experiments

In this section, we evaluate and compare our algorithms[4] against two types of methods: (i) those assuming causal sufficiency (DAG learning): PC (Spirtes et al. 2000), NOTREARS (Zheng et al. 2018), CORL (Wang et al. 2021), and ARGES (Nandy, Hauser, and Maathuis 2018); (ii) those that do not require causal sufficiency (MAG learning): RFCI (Colombo et al. 2012), FCI+ (Claassen, Mooij, and Heskes 2013), L-MARVEL (Akbari et al. 2021), MBCS* (Pellet and Elisseeff 2008a), and GSPo (Bernstein et al. 2020).

We evaluated the aforementioned algorithms[5] on finite sets of samples, where they were generated using a linear SEM. The coefficients were chosen uniformly at random from $[-1.5, -1] \cup [1, 1.5]$; the exogenous noises were generated from normal distribution $\mathcal{N}(0, \sigma^2)$, where σ was selected uniformly at random from $[0.7, 1.2]$. We measured the performance of the algorithms by two commonly used metrics in the literature: F1-score and Structural Hamming Distance (SHD) (the discrepancy between the number of extra and missing edges in the learned vs the ground truth graph).

Each point on the plots is reported as the average of 10 runs with 80% confidence interval. Also, each entry in the tables is reported as an average of 10 runs.

Structure (#nodes, #edges)		Earthquake (5, 4)	Survey (6, 6)	Asia (8, 8)	Sachs (11, 17)
ROL_{VI}	F1	0.96	1	0.97	0.97
	SHD	0.4	0	0.4	1
ROL_{HC}	F1	0.96	0.98	0.97	0.95
	SHD	0.4	0.2	0.4	1.6

Table 1: Comparing ROL_{HC} and ROL_{VI} on small graphs.

DAG Learning

We consider two types of graphs: random graphs generated from Erdös-Rènyi model $Er(n, p)$ and real-world networks[6]. To generate a DAG from $Er(n, p)$, the skeleton is first sampled using the Erdös-Rènyi model (Erdős and Rényi 1960) in which undirected edges are sampled independently with probability p. Then, the edges are oriented according to a randomly selected c-order.

Figure 3 shows the results for learning DAGs. In Figure 3a, DAGs are generated from $Er(n, p = n^{-0.7})$ and n varies from 10 to 100. The size of the datasets generated for this part is $50n$. Figures 3b, 3c, and 3d depict the performance of the algorithms on three real-world structures, called Alarm, Barley, and Hepar2, for a various number of samples. As shown in these figures, ROL_{PG} and ROL_{HC} outperform the state of the art in both SHD and F1-score metrics.

Table 1 illustrates the performance of ROL_{VI} in comparison to ROL_{HC} on four small real-world structures. This table shows that ROL_{VI} achieves better accuracy on small graphs. Note that ROL_{VI} unlike ROL_{HC} has theoretical guarantees, but is not scalable to large graphs. However, ROL_{VI}'s performance is limited to the accuracy of CI tests, and by increasing the dataset sizes, ROL_{VI} performs without any errors.

MAG Learning

We selected seven real-world DAGs for this part. For each structure, we randomly removed 5% to 10% of the variables and constructed a MAG over the set of observed variables (those not eliminated) using the latent projection approach of Verma and Pearl (1991). Finally, we generated a finite set of samples over all the variables and fed the data pertaining to the observed variables as the input to all the algorithms. The goal of all algorithms is to learn the MEC of the corresponding MAG from the samples they have.

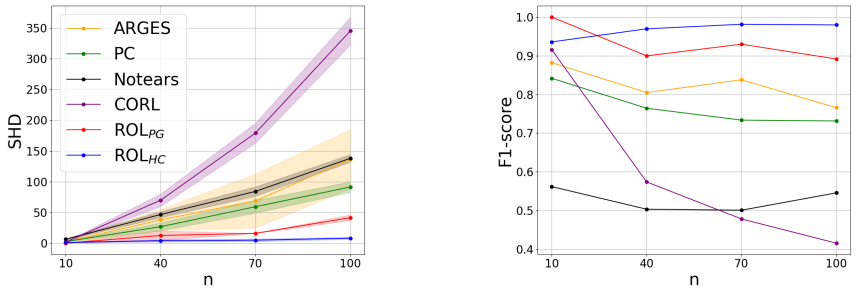
Table 2 presents the results. As demonstrated by the bold entries in the table, ROL_{HC} achieves the best F1-score and SHD in almost all the cases.

Remark 2. *Recall that prior to this work, GSPo was the only ordering-based method in the literature that does not*

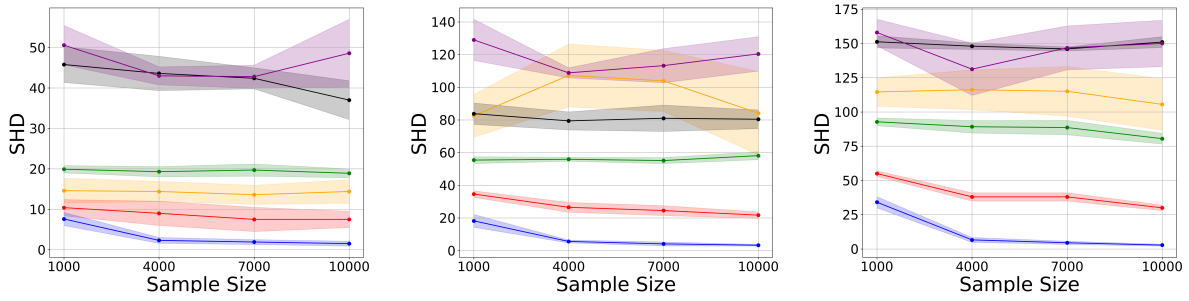
[4] github.com/ban-epfl/ROL

[5] Details pertaining to the reproducibility, hyperparameters, and additional experiments are provided in Appendix C.

[6] bnlearn.com/bnrepository



(a) Erdös-Rènyi $Er(n, p = n^{-0.7})$, sample size $= 50n$.



(b) Alarm ($n = 37, e = 46$). (c) Barley ($n = 48, e = 84$). (d) Hepar2 ($n = 70, e = 123$).

Figure 3: DAG learning; n and e denote the number of variables and the number of edges, respectively.

Structure (#Observed, #Unobserved)		Insurance (24, 3)	Water (29, 3)	Ecoli70 (43, 3)	Hailfinder (53, 3)	Carpo (57, 4)	Hepar2 (65, 5)	Arth150 (100, 7)
ROL$_{HC}$	F1-score	**0.89**	**0.86**	**0.93**	**0.90**	**1.00**	**0.97**	**0.93**
	SHD	**10.1**	**18.3**	**8.1**	13.9	**0.2**	**7.6**	**19.3**
ROL$_{PG}$	F1-score	0.86	0.76	0.90	0.87	0.97	0.84	**0.93**
	SHD	12.9	35.3	12.7	18.4	4.1	36.5	21.5
RFCI	F1-score	0.74	0.68	0.84	0.84	0.86	0.70	0.86
	SHD	20.5	34.0	17.7	19.7	16.8	52.6	36.3
FCI+	F1-score	0.60	0.55	0.78	0.77	0.80	0.57	0.78
	SHD	31.2	50.0	23.5	27.2	24.4	81.4	56.7
L-MARVEL	F1-score	0.87	0.78	**0.93**	**0.90**	0.99	0.94	0.92
	SHD	11.5	26.2	8.5	**12.8**	0.8	12.3	21.2
MBCS*	F1-score	0.77	0.62	0.90	0.83	0.99	0.92	0.87
	SHD	17.8	38.7	12.0	20.1	1.1	17.0	34.2
GSPo	F1-score	0.75	0.60	0.66	0.58	0.84	0.58	0.45
	SHD	32.3	89.1	67.6	101.4	31.2	170.6	358.5

Table 2: MAG learning; performance of various algorithms on seven real-world structures, when sample size is $50n$.

require causal sufficiency. However, the table shows that it has the worst performance among the algorithms and is not scalable to large graphs. For instance, it has a poor performance on Arth150, which is a graph with 100 variables.

Conclusion

We advocated for a novel type of order, called an r-order, and argued that r-orders are advantageous over the previously used orders in the literature. Accordingly, we proposed three algorithms for causal structure learning in the presence of unobserved variables: ROL$_{HC}$, a Hill-climbing-based heuristic algorithm that is scalable to large graphs; ROL$_{VI}$, an exact RL-based algorithm that has theoretical guarantees but is not scalable to large graphs; ROL$_{PG}$, an approximate RL-based algorithm that exploits stochastic policy gradient. We showed in our experiments that ROL$_{VI}$ on small graphs and ROL$_{HC}$ on larger graphs outperform the state-of-the-art algorithms. Although ROL$_{PG}$ is scalable to large graphs and outperforms the existing methods, ROL$_{HC}$ performs slightly better, mainly due to better initialization. The weights of the neural networks in ROL$_{PG}$ are selected randomly, while we proposed clever methods for the initialization step in ROL$_{HC}$. Nevertheless, an important future work is to improve the policy gradient approaches.

Acknowledgements

This work was supported in part by the SNF project 200021_204355/1, Causal Reasoning Beyond Markov Equivalencies.

References

Akbari, S.; Mokhtarian, E.; Ghassami, A.; and Kiyavash, N. 2021. Recursive Causal Structure Learning in the Presence of Latent Variables and Selection Bias. *Advances in Neural Information Processing Systems*, 34.

Alonso-Barba, J. I.; Gámez, J. A.; Puerta, J. M.; et al. 2013. Scaling up the greedy equivalence search algorithm by constraining the search space of equivalence classes. *International journal of approximate reasoning*, 54(4): 429–451.

Bernstein, D.; Saeed, B.; Squires, C.; and Uhler, C. 2020. Ordering-based causal structure learning in the presence of latent variables. In *International Conference on Artificial Intelligence and Statistics*, 4098–4108. PMLR.

Bottou, L.; Peters, J.; Quiñonero-Candela, J.; Charles, D. X.; Chickering, D. M.; Portugaly, E.; Ray, D.; Simard, P.; and Snelson, E. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research*, 14(11).

Claassen, T.; Mooij, J. M.; and Heskes, T. 2013. Learning Sparse Causal Models is not NP-hard. In *In Proceedings of the 29th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 172–181.

Colombo, D.; Maathuis, M. H.; Kalisch, M.; and Richardson, T. S. 2012. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 294–321.

Erdős, P.; and Rényi, A. 1960. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5: 17–61.

Friedman, N.; and Koller, D. 2003. Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50(1): 95–125.

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.

Ghoshal, A.; Bello, K.; and Honorio, J. 2019. Direct learning with guarantees of the difference dag between structural equation models. *arXiv preprint arXiv:1906.12024*.

Lachapelle, S.; Brouillard, P.; Deleu, T.; and Lacoste-Julien, S. 2020. Gradient-based neural dag learning. *ICLR*.

Larranaga, P.; Kuijpers, C. M.; Murga, R. H.; and Yurramendi, Y. 1996. Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE transactions on systems, man, and cybernetics-part A: systems and humans*, 26(4): 487–493.

Margaritis, D.; and Thrun, S. 1999. Bayesian network induction via local neighborhoods. *Advances in Neural Information Processing Systems*, 12: 505–511.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mokhtarian, E.; Akbari, S.; Ghassami, A.; and Kiyavash, N. 2021. A recursive Markov boundary-based approach to causal structure learning. In *The KDD'21 Workshop on Causal Discovery*, 26–54. PMLR.

Mokhtarian, E.; Akbari, S.; Jamshidi, F.; Etesami, J.; and Kiyavash, N. 2022. Learning Bayesian networks in the presence of structural side information. *Proceeding of AAAI-22, the Thirty-Sixth AAAI Conference on Artificial Intelligence*.

Nandy, P.; Hauser, A.; and Maathuis, M. H. 2018. High-dimensional consistency in score-based and hybrid structure learning. *The Annals of Statistics*, 46(6A): 3151–3183.

Ng, I.; Zhu, S.; Fang, Z.; Li, H.; Chen, Z.; and Wang, J. 2022. Masked gradient-based causal structure learning. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, 424–432. SIAM.

Pearl, J. 2009. *Causality*. Cambridge university press.

Pellet, J.-P.; and Elisseeff, A. 2008a. Finding latent causes in causal networks: an efficient approach based on Markov blankets. *Neural Information Processing Systems Foundation*.

Pellet, J.-P.; and Elisseeff, A. 2008b. Using Markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9(Jul): 1295–1342.

Raskutti, G.; and Uhler, C. 2018. Learning directed acyclic graph models based on sparsest permutations. *ISI Journal for the Rapid Dissemination of Statistics Research*.

Richardson, T.; and Spirtes, P. 2002. Ancestral graph Markov models. *The Annals of Statistics*, 30(4): 962–1030.

Rolland, P.; Cevher, V.; Kleindessner, M.; Russell, C.; Janzing, D.; Schölkopf, B.; and Locatello, F. 2022. Score matching enables causal discovery of nonlinear additive noise models. In *International Conference on Machine Learning*, 18741–18753. PMLR.

Russo, F. 2010. *Causality and causal modelling in the social sciences*. Springer Dordrecht.

Sachs, K.; Perez, O.; Pe'er, D.; Lauffenburger, D. A.; and Nolan, G. P. 2005. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529.

Schmidt, M.; Niculescu-Mizil, A.; Murphy, K.; et al. 2007. Learning graphical model structure using L1-regularization paths. In *AAAI*, volume 7, 1278–1283.

Schulte, O.; Frigo, G.; Greiner, R.; and Khosravi, H. 2010. The IMAP hybrid method for learning Gaussian Bayes nets. In *Canadian Conference on Artificial Intelligence*, 123–134. Springer.

Spirtes, P.; Glymour, C. N.; Scheines, R.; and Heckerman, D. 2000. *Causation, prediction, and search*. MIT press.

Sun, X.; Janzing, D.; Schölkopf, B.; and Fukumizu, K. 2007. A kernel-based causal learning algorithm. In *Proceedings of the 24th international conference on Machine learning*, 855–862.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

Teyssier, M.; and Koller, D. 2005. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. *In Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 584–590.

Tsamardinos, I.; Aliferis, C. F.; and Statnikov, A. 2003. Time and sample efficient discovery of Markov blankets and direct causal relations. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 673–678.

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.

Tsirlis, K.; Lagani, V.; Triantafillou, S.; and Tsamardinos, I. 2018. On scoring maximal ancestral graphs with the max–min hill climbing algorithm. *International Journal of Approximate Reasoning*, 102: 74–85.

Verma, T.; and Pearl, J. 1991. *Equivalence and synthesis of causal models*. UCLA, Computer Science Department.

Wang, X.; Du, Y.; Zhu, S.; Ke, L.; Chen, Z.; Hao, J.; and Wang, J. 2021. Ordering-based causal discovery with reinforcement learning. *International Joint Conference on Artificial Intelligence (IJCAI)*.

Watkins, C. J.; and Dayan, P. 1992. Q-learning. *Machine learning*, 8(3): 279–292.

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

Yang, Y.; Ghassami, A.; Nafea, M.; Kiyavash, N.; Zhang, K.; and Shpitser, I. 2022. Causal Discovery in Linear Latent Variable Models Subject to Measurement Error. *arXiv preprint arXiv:2211.03984*.

Yu, Y.; Chen, J.; Gao, T.; and Yu, M. 2019. DAG-GNN: DAG structure learning with graph neural networks. In *International Conference on Machine Learning*, 7154–7163. PMLR.

Zhang, H.; Zhou, S.; Yan, C.; Guan, J.; and Wang, X. 2019. Recursively learning causal structures using regression-based conditional independence test. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3108–3115.

Zhang, J. 2008. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9: 1437–1474.

Zhang, K.; Peters, J.; Janzing, D.; and Schölkopf, B. 2011. Kernel-based conditional independence test and application in causal discovery. *In Proceedings of the 27th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 804–813.

Zhang, K.; Schölkopf, B.; Spirtes, P.; and Glymour, C. 2017. Learning causality and causality-related learning: Some recent progress. *National Science Review*, 5(1): 26–29.

Zheng, X.; Aragam, B.; Ravikumar, P. K.; and Xing, E. P. 2018. Dags with no tears: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, 31.

Zheng, X.; Dan, C.; Aragam, B.; Ravikumar, P.; and Xing, E. 2020. Learning sparse nonparametric dags. In *International Conference on Artificial Intelligence and Statistics*, 3414–3425. PMLR.

Zhu, S.; Ng, I.; and Chen, Z. 2020. Causal discovery with reinforcement learning. *ICLR*.