On Low-Rank Directed Acyclical Graphs and Causal Structure Learning

Zhuangyan Fang, Shengyu Zhu, Jiji Zhang, Yue Liu, Zhitang Chen, and Yangbo He

Abstract—Despite several advances in recent years, learning causal structures represented by directed acyclical graphs (DAGs) remains a challenging task in high-dimensional settings when the graphs to be learned are not sparse. In this article, we propose to exploit a low-rank assumption regarding the (weighted) adjacency matrix of a DAG causal model to help address this problem. We utilize existing low-rank techniques to adapt causal structure learning methods to take advantage of this assumption and establish several useful results relating interpretable graphical conditions to the low-rank assumption. Specifically, we show that the maximum rank is highly related to hubs, suggesting that scale-free (SF) networks, which are frequently encountered in practice, tend to be low rank. Our experiments demonstrate the utility of the low-rank adaptations for a variety of data models, especially with relatively large and dense graphs. Moreover, with a validation procedure, the adaptations maintain a superior or comparable performance even when graphs are not restricted to be low rank.

Index Terms—Bipartite graph, directed acyclical graph (DAG), graphical model, low rank, structure learning.

I. INTRODUCTION

A N IMPORTANT goal in many empirical sciences is to discover the underlying causal structures in various domains, for the purpose of explaining and understanding phenomena and predicting effects of interventions [1]. Due to the relative abundance of passively observed data as opposed to experimental data, learning causal structures from purely observational data has been vigorously investigated [2], [3]. In this context, causal structures are usually represented by directed acyclical graphs (DAGs) over a set of random variables, and existing structure learning methods fall roughly into two classes: constraint-based [3], [4], [5] and score-based [6], [7], [8].

Constraint-based methods, such as Peter-Clark (PC) and fast causal inference [3], use conditional independence tests to find a skeleton and then determine the edge directions using some orientation rules [4], [5]. Due to the combinatorial nature of the acyclicity constraint [6], traditional score-based methods mostly rely on local heuristics to search for a DAG according to a predefined score function. Recently, a smooth acyclicity constraint with respect to (w.r.t.) graph adjacency matrix was introduced in [9], and the score-based approach on linear data models was then formulated as a continuous optimization problem with least-squares loss. This change of perspective allows using deep-learning techniques to model causal mechanisms and has already given rise to several new algorithms for causal structure learning with nonlinear data (see [10], [11], [12], [13], [14], [15], among others). We refer to these algorithms as gradient-based, as the formulated problems are solved using first-order numerical methods. While gradient-based methods represent the current state of the art in many settings, their performance generally degrades when the target DAG is large and relatively dense, as seen from the empirical results in the above references.

This issue is of course a challenge to many other approaches. Ramsey et al. [8] proposed a fast greedy equivalence search [fast greedy equivalence search (GES)] for impressively large problems, but it works reasonably well only when the structure is very sparse. The max-min hill-climbing (MMHC) method [16] relies on local learning methods that often do not perform well when target nodes have large neighborhoods. A potential reason is that dense graphs tend to have high in-degrees that affect the performance of many causal structure learning methods: the search space of PC [3] and GES grows exponentially in the maximum in-degree and the sample complexity of the methods proposed in [17] and [18] also increases fast with the maximum in-degree. How to improve the performance on relatively large and dense DAGs is, therefore, an important question.

In this work, we study the potential of exploiting a kind of low-rank assumption on the DAG structure to mitigate this problem. The rank of a graph that concerns us is the algebraic rank of its associated weighted adjacency matrix. Similar to the role of a sparsity assumption, the low-rank assumption
is methodological and is not restricted to a particular DAG learning method. However, unlike sparsity, the graph rank is an algebraic concept—it is much less apparent when DAGs tend to be low rank and how low-rank DAGs behave. Thus, besides demonstrating the utility of exploiting a low-rank assumption in causal structure learning, the primary goal of this article is to improve our understanding of the low-rank assumption by relating the rank of a graph to its graphical structure. Such results may also help characterize the rank of a graph in certain cases, which in turn inform the choice of rank-related hyperparameters for learning algorithms.

Contributions: To improve our understanding of low-rank DAGs, we establish some lower bounds on the rank of a DAG in terms of simple graphical conditions, which imply necessary conditions for DAGs to be low rank. We also prove that the maximum possible rank of weighted adjacency matrices associated with a directed graph is closely related to hubs in the graph, which suggests that scale-free (SF) networks tend to be low rank. From this result, we derive several graphical conditions to bind the rank of a DAG from above, providing sufficient conditions for a low rank. We then show how to utilize existing low-rank techniques to adapt a class of recently developed gradient-based causal structure learning methods with little extra effort. Empirically, the low-rank adaptations are indeed useful, especially with relatively large and dense graphs that tend to have high in-degrees. Not only do they outperform the original algorithms when the low-rank condition is satisfied, but the performance also remains very competitive even when graphs are not restricted to low rank.

Other Related Work: A low-rank assumption is frequently utilized to solve large-scale problems in practice, for example, in graph-based applications [19], [20], [21], [22], [23], [24], [25], matrix completion and factorization [26], [27], [28], [29], [30], [31], [32], [33], network sciences [34], [35], [36], and so on. In the context of DAG structure learning, Barik and Honorio [37] and Tichavský and Vomlel [38] focused on discrete Bayesian networks and placed the low-rank assumption on the parameter space encoded by conditional probability tables, which is not easily applied to continuous cases. Instead, in this work, we mostly consider continuous variables and place the low-rank assumption on the graph structure, which, to the best of our knowledge, has not been studied in the literature.

The recently developed gradient-based method brings in an opportunity to employ existing techniques such as matrix factorization (see [39], [40], [41]) and nuclear norm (see [42], [43], [44], [45], [46]) for low-rank DAG learning, as we will show in Section IV. However, there is a limited understanding of low-rank assumption on the DAG structure, as algebraic knowledge about graph rank is less apparent than structural knowledge. In graph theory, many works have tried to relate graph rank to graph structure. The minimum rank problem studies the minimum rank among all real matrices whose zero nonzero pattern of off-diagonal entries is described by a simple graph [47], [48], [49]. Here, the diagonal entries of real matrices can pick any value (including zero). This problem has not been fully solved, but some bounds characterized by structural information do exist, for example, in [48]. An analogous maximum rank problem, as mentioned in [47], is trivial: the maximum rank among all real matrices with the same zero nonzero pattern of off-diagonal entries always equals the number of vertices, which can be achieved by setting a sufficiently large value to the diagonal elements. We comment again that the diagonal entries of the considered matrices can take any real values in both the minimum and maximum rank problems. In contrast, for DAG learning, we must place constraints on the diagonal entries and fix them at zero. Related to this new setting is [50] that studied the maximum rank for matrices with a common zero-nonzero pattern (including diagonal entries). In Section III, this result will be used to relate the maximum possible rank to a more interpretable graphical condition, which further implies several structural properties of low-rank DAGs.

Article Organization: After introducing preliminaries in Section II, we present the bounds on graph ranks with interpretable graphical conditions in Section III. Section IV discusses the adaptations of existing methods to learn low-rank DAGs, followed by experimental investigations in Section V. Section VI concludes this article.

II. PRELIMINARIES

We first introduce graph terminology, notations, and recently developed gradient-based causal structure learning methods.

A. Graph Terminology

A graph \( G = (V, E) \) consists of a vertex set \( V = \{X_1, X_2, \ldots, X_d\} \) and an edge set \( E \subset V \times V \). We particularly focus on directed (acyclic) graphs. We use \( \text{pa}(S, G) \), \( \text{ch}(S, G) \), and \( \text{adj}(S, G) \) to denote the union of all parents, children, and adjacent vertices of the nodes in \( S \subset V \) in \( G \), respectively. A graph is weighted if every edge in the graph is associated with a nonzero value. We will work with weighted graphs and treat unweighted graphs as weighted ones whose edge weights are all 1. A \( d \)-node weighted graph can be represented by a \( d \times d \) matrix \( W \), called a weighted adjacency matrix, where \( W(i, j) = 0 \) if \( X_i \rightarrow X_j \) and is the weight of \( X_i \rightarrow X_j \) otherwise. The binary adjacency matrix \( A \in \{0,1\}^{d \times d} \) is such that \( A(i, j) = 1 \) if \( X_i \rightarrow X_j \) in \( G \) and \( A(i, j) = 0 \) otherwise. The rank of a weighted graph is defined as the rank of the associated weighted adjacency matrix.

A bipartite graph is a graph whose vertex set \( V \) can be partitioned into two disjoint subsets \( V_0 \) and \( V_1 \), such that the vertices within each subset are not adjacent to one another. \( V_0 \) and \( V_1 \) are called the parts of the graph. A matching of a graph is a subset of its edges where no two of them share a common endpoint. A vertex cover of a graph is a subset of the vertex set where every edge in the graph has at least one endpoint in the subset. The size of a matching (vertex cover) is the number of edges (vertices) in the matching (vertex cover). A maximum matching of a graph is a matching of the largest possible size and a minimum vertex cover is a vertex cover of the smallest possible size. König’s theorem [51] states that the size of a minimum vertex cover is equal to the size of a maximum matching in a bipartite graph.
B. Causal Structure Learning and Recent Methods

We assume that the data-generating procedure follows a structural causal model (SCM). Let $\mathcal{G}$ be a DAG. Then an SCM defined w.r.t. $\mathcal{G}$ is given by

$$X_i = f_i(p(X_i, \mathcal{G}), \epsilon_i), \quad i = 1, 2, \ldots, d$$

where $f_i$ is a deterministic function and $\epsilon_i$ are jointly independent noises. The SCM induces a marginal distribution $P(X)$ over $X = [X_1, X_2, \ldots, X_d]^T$. In this article, we assume that there is no latent variable or selection bias.

The problem of causal structure learning is to infer $\mathcal{G}$ based on a number of independent and identically distributed (i.i.d.) observations from $P(X)$. In the following, we briefly review the recently developed gradient-based methods that use a smooth characterization of the acyclicity of directed graphs, which can be categorized into two classes.

The first class explicitly associates the target causal model with a weighted adjacency matrix $W$ and then estimates $W$ by solving optimization problems in the following form:

$$\min_{W, \phi} \mathbb{E}_{P(X)} S(X, h(X; W, \phi)), \quad \text{s.t. } g(W) = 0 \quad (1)$$

where $h : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is a model function parameterized by $W$ (and possibly other parameters $\phi$) that aims to reconstruct $X$ from $p(X_i, \mathcal{G})$ and $S(\cdot, \cdot)$ denotes a score function between the true and reconstructed variables. Here, $g(W) : = \text{tr}(e^{W \cdot W}) - d$, where $\text{tr}(\cdot)$ is the trace function, $\cdot$ denotes the elementwise product, and $e^W$ is the matrix exponential of a square matrix $M$. The constraint was proposed by Zheng et al. [9], which is smooth and holds if and only if $W$ indicates a DAG. Methods in this class include: non-combinatorial optimization via trace exponential and augmented lagrangian for structure learning (NOTEARS) [9] targeting linear models, and DAG structure learning with graph neural networks (DAG-GNN) [10] and graph autoencoder (GAE) approaches [11] that adopt neural networks to model nonlinear relationships. Notice that an extra sparsity-inducing term may also be added when the causal graph is known to be sparse and that the objective is usually chosen from some well-studied score functions (or their variants), for example, penalized maximum likelihood [7, 52, 53]. Further details like parameter optimization techniques can be found in the cited works and references therein.

The second class uses certain functions, with parameters denoted by $\theta$, to construct a weighted adjacency matrix $W(\theta)$ [or a binary one $A(\theta)$] to represent the causal structure. These methods can be summarized as

$$\min_{\theta, \phi} \mathbb{E}_{P(X)} S(X, h(X; W(\theta), \phi)), \quad \text{s.t. } g(W(\theta)) = 0. \quad (2)$$

For example, gradient-based neural DAG learning (GraNeDAG) [13] and NOTEARS multilayer perceptron (NOTEARS-MLP) [14], respectively, use neural network path products and partial derivatives between variables to construct $W(\theta)$. The binary matrix $A(\theta)$ can be sampled according to some distributions with learnable parameters, as in [12, 15, 54, 55, and 56].

Remark 1: While these methods intend to learn a DAG, the learned DAG may not be identical to the true one due to Markov equivalence [3]. For such cases, we may convert the obtained DAG to its completed partially DAG (CPDAG) as the final estimate. On the other hand, several forms of SCMs have been shown to be identifiable; if a proper score function is used, then the exact solution to the optimization problem is consistent, that is, the same as the true graph with probability asymptotically approaching 1 (see [57, 58, 59, 60], among others).

III. Graphical Bounds on Ranks

This section is devoted to relating the rank of a graph to interpretable graphical conditions, for a better understanding of what kinds of DAGs tend to be low rank. Section III-A gives a formal definition of the problem, and Sections III-B and III-C, respectively, present lower and upper bounds on the graph rank based on some structural priors. If this structural information, such as connectivity, distributions of in-degrees and out-degrees, and an estimate of the number of hubs, is accessible, we can then obtain an estimate of the graph rank.

A. Problem Formulation

Consider a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with weighted adjacency matrix $W$ and binary adjacency matrix $A$. Our goal is to find upper and lower bounds on $\text{rank}(W)$ based on graphical structure; specifically, we focus on the weighted adjacency matrices with the same binary adjacency matrix $A$, that is, $\mathcal{W}_A = \{W \in \mathbb{R}^{d \times d} : \text{sign}(|W|) = A\}$, where $\text{sign}(\cdot)$ and $|\cdot|$ are the pointwise sign and absolute value functions, respectively. While trivial upper-bound $(d - 1)$ and lower-bound 0 exist for any DAG, they are generally too loose for practical use. In the following, we investigate the maximum rank $\max \text{rank}(W); W \in \mathcal{W}_A$ and minimum rank $\min \text{rank}(W); W \in \mathcal{W}_A$, for tighter upper and lower bounds for any $W \in \mathcal{W}_A$. For ease of presentation, we will use $\min \text{rank}(\mathcal{W}_A)$ and $\max \text{rank}(\mathcal{W}_A)$ to represent the minimum and maximum ranks, respectively.

To proceed, we introduce two useful graph concepts, namely, height and head-tail vertex cover.

Definition 1 (Height): Given a DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a vertex $X_i$, the height of $X_i$, denoted by $l(X_i)$, is the length of the longest directed path starting from $X_i$. The height of $\mathcal{G}$, denoted by $l(\mathcal{G})$, is the length of the longest path in $\mathcal{G}$.

Based on the heights of vertices in $\mathcal{V}$, we can define a weak ordering among the vertices: $X_i > X_j$ if and only if $l(X_i) > l(X_j)$, and $X_i \sim X_j$ if and only if $l(X_i) = l(X_j)$. Let $\mathcal{V}_s = \{X_i; l(X_i) = s\}$, $s = 0, 1, \ldots, l(\mathcal{G})$, and let $\mathcal{V}_{-1} = \emptyset$. It can be verified that: 1) for any given $s \in \{0, 1, \ldots, l(\mathcal{G})\}$ and two distinct vertices $X_1, X_2 \in \mathcal{V}_s$, $X_1$ and $X_2$ are not adjacent, and 2) for any given $s \in \{1, 2, \ldots, l(\mathcal{G})\}$ and $X_i \in \mathcal{V}_s$, there is at least one vertex in $\mathcal{V}_{s-1}$ which is a child of $X_i$. If we denote the induced subgraph of $\mathcal{G}$ over $\mathcal{V}_s \cup \mathcal{V}_{s-1}$ by $\mathcal{G}_{s,s-1}$, then $\mathcal{G}_{s,s-1}$ is a bipartite graph with $\mathcal{V}_s$ and $\mathcal{V}_{s-1}$ as its vertex sets, and singleton vertices in $\mathcal{G}_{s,s-1}$ (i.e., vertices that are not endpoints of any edge) only appear in $\mathcal{V}_{s-1}$.

Definition 2 (Head-Tail Vertex Cover): Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph and $\mathcal{H}, \mathcal{T}$ be two subsets of $\mathcal{V}$. Then $(\mathcal{H}, \mathcal{T})$ is called a head–tail vertex cover of $\mathcal{G}$ if every edge in $\mathcal{G}$ has its head vertex in $\mathcal{H}$ or its tail vertex in $\mathcal{T}$. The size of a head–tail vertex cover $(\mathcal{H}, \mathcal{T})$ is defined as $|\mathcal{H}| + |\mathcal{T}|$.

As an example, consider a directed line graph with $d$ vertices and $d - 1$ edges, that is, $X_1 \rightarrow X_2 \rightarrow \ldots \rightarrow X_d$. The minimum size of a head–tail vertex cover is $d - 1$. The maximum size of a head–tail vertex cover is $d$. The height of $\mathcal{G}$ is $d$.
Then \( \mathbf{H} = \{X_i\}_{i=1}^d \) and \( \mathbf{T} = \emptyset \) form a head–tail vertex cover, with size \( d-1 \).

**B. Lower Bounds**

We first study lower bounds on the rank of a weighted DAG. Let \( C(G_{s_1 s_2}) \) be the set of nonsingleton connected components of \( G_{s_1 s_2} \). As \( G_{s_1 s_2} \) has at least one nonsingleton connected component, we obtain the second inequality. That is, the rank of a weighted DAG is at least as high as the length of the longest directed path.

**Theorem 1:** Let \( \mathcal{G} \) be a DAG with binary adjacency matrix \( A \). Then we have

\[
\min \{ \text{rank}(W_A) \} \geq \sum_{i=1}^{l(\mathcal{G})} |C(G_{s_1 s_2})| \geq l(\mathcal{G}).
\]

The complete proofs in this article are provided in Appendix A. Theorem 1 shows that \( \text{rank}(W) \) is greater than or equal to the sum of the number of nonsingleton connected components in each \( G_{s_1 s_2} \). As \( G_{s_1 s_2} \) has at least one nonsingleton connected component, we obtain the second inequality. That is, the rank of a weighted DAG is at least as high as the length of the longest directed path.

Theorem 1 also indicates that a sparse graph may have a very high rank. For example, according to Theorem 1, the minimum rank of a directed line graph is \( d - 1 \). As \( d - 1 \) is a trivial upper bound of any DAG, the rank of a directed line graph always equals its number of edges. On the other hand, for some nonsparse graphs, we can assign the edge weights so that the resulting graphs have low ranks. A simple example would be a fully connected directed balanced bipartite graph.

A bipartite graph is called balanced if its two parts contain the same number of vertices. The rank of a fully connected balanced bipartite graph with \( d \) vertices is \( 1 \) if the edge weights are the same, but the number of edges is \( d^2/4 \). We remark that to exactly characterize the minimum rank is still an on-going research problem [48].

**C. Upper Bounds**

We turn to another important issue regarding the upper bounds on \( \text{rank}(W) \). The next theorem characterizes \( \max \{ \text{rank}(W_A) \} \) w.r.t. graphical terms.

**Theorem 2:** Let \( \mathcal{G} \) be a directed graph with a binary adjacency matrix \( A \). Then \( \max \{ \text{rank}(W_A) \} \) is equal to the minimum size of the head–tail vertex cover of \( \mathcal{G} \).

We comment that Theorem 2 holds for all directed graphs (not only DAGs), which may be of independent interest in other applications. A head–tail vertex cover of minimum size is called a minimum head–tail vertex cover, which, in general, is not unique. For a head–tail vertex cover \( (\mathbf{H}, \mathbf{T}) \), the vertices in \( \mathbf{H} \) cover all the edges pointing toward these vertices while the vertices in \( \mathbf{T} \) cover the edges pointing away. A head–tail cover of a relatively small size then indicates the presence of hubs, that is, vertices with relatively high in-degrees or out-degrees. Therefore, Theorem 2 suggests that the maximum rank of a weighted DAG is highly related to the presence of hubs: a DAG with many hubs tends to have a low rank. Intuitively, a hub of high in-degree (resp. out-degree) is a common effect (resp. cause) of a number of direct causes (resp. effect variables), comprising many V-structures (resp. inverted V-structures).

Fig. 1. Graph ranks (left) and maximum in-degrees (right) versus average degrees with 100-node ER and SF random graphs.

Such features are frequently encountered in real-world causal networks, and the existence of such features may be part of domain knowledge. For instance, it was claimed in [61] that some protein networks, which are directed and acyclic due to irreversible reactions, are the results of growth processes and preferential attachments, due to gene duplication. Thus, many gene and protein networks tend to have hubs of high out-degree. Another example is from a financial network that describes the direction of risk contagion: the insurance companies take risks from many other financial institutions and thus are likely to have high in-degrees [62].

In addition, the frequently encountered SF graphs have hubs and tend to be low rank [8], [61], [62], [63], [64], [65]. An SF graph is one whose distribution of degree \( k \) follows a power law: \( P(k) \sim k^{-\gamma} \), where \( \gamma \) is the power parameter typically within \([2, 3]\) and \( P(k) \) denotes the fraction of nodes with degree \( k \) [66]. Empirically, the ranks of SF graphs are relatively low, especially in comparison to Erdös-Rényi (ER) random graphs [67].

Next, we report some simpler upper bounds.

**Theorem 3:** Let \( \mathcal{G} \) be a DAG with a binary adjacency matrix \( A \). Denote by \( V_{\text{ch}} \) the set of vertices that are children of some vertices and by \( V_{\text{pa}} \) those that are parents of some vertices. Then we have

\[
\max \{ \text{rank}(W_A) \} \leq \left\{ \sum_{l=1}^{|V|} \min \{ |V_i|, |\text{ch}(V_i)| \} \leq |V_{\text{pa}}| \right. \\
\left. \sum_{l=1}^{|V|} \min \{ |V_i|, |\text{pa}(V_i)| \} \leq |V_{\text{ch}}| \right. \\
\left. \left| V \right| - \max \{ |V_i|; 0 \leq s \leq l(\mathcal{G}) \}. \right.
\]

As \( V_{\text{ch}} \) and \( V_{\text{pa}} \), respectively, represent the sets of nonroot and nonleaf nodes, the first two bounds in Theorem 3 indicate that the maximum rank is bounded from above by the number of nonroot nodes and also the number of nonleaf nodes. These simple bounds are useful when there is background knowledge indicating that many variables are roots or are leaves in the underlying causal network, for example, the ecoli70 [68] and magic-niab [69] networks. The last inequality...
Algorithm 1 Generating Random DAGs

Require: Number of nodes $d$, average degree $k$, and rank $r$.
Ensure: A randomly sampled DAG with the number of nodes $d$, average degree $k$, and rank $r$.
1: Set $M = \emptyset$, $M_p = \emptyset$, $R = \{(i, j); i < j, i, j = 1, 2, \ldots, d\}$, and $p = k/(d-1)$.
2: Sample a number $N \sim B(d(d-1)/2, p)$, where $B(n, p)$ is a binomial distribution with parameters $n$ and $p$.
3: if $N < r$ then
  4:    return FAIL
5:   end if
6:   Sample $r$ indices from $1, \ldots, d - 1$ and store them in $M_p$ in descending order.
7:   for each $i$ in $M_p$ do
8:     Sample an index $j$ from $i + 1$ to $d$, add edge $(i, j)$ to $M$, and remove $(i, j)$ from $R$.
9:   end for
10:  while $R \neq \emptyset$ and $|M| < N$ do
11:    Sample an edge $(i, j)$ from $R$ and remove it from $R$.
12:    Add $(i, j)$ to $M$ if it does not change the size of the minimum head–tail vertex cover of $M$.
13:  end while
14:  if $|M| < N$ then
15:    return FAIL
16:  end if
17:  Permute randomly the vertex indexes in $M$.
18:  Assign the edge weights of $M$ randomly according to a continuous distribution.
19:  return $M$

is a generalization of the first two, which implies that the rank is likely to be low if most vertices have the same height.

**Theorem 4:** Let $G$ be a DAG with a binary adjacency matrix $A$. Denote by $\text{skel}(A)$ and $\text{moral}(A)$ the binary adjacency matrices of the skeleton and moral graphs of $G$, respectively. Then we have

$$\max\{\text{rank}(W_A)\} \leq \max\{\text{rank}(W); \text{sign}(W) = \text{skel}(A)\} \leq \max\{\text{rank}(W); \text{sign}(W) = \text{moral}(A)\}.$$

The skeleton of a DAG is the undirected graph obtained by removing all the arrowheads, and the moral graph is the undirected graph where two vertices are adjacent if they are adjacent or if they share a common child in the DAG. This theorem may be useful when the skeleton or the moral graph can be easily estimated from data, for example, in a linear Gaussian model where the zero nonzero pattern of the precision matrix indicates the moral graph.

D. Generating Low-Rank DAGs

In this section, we discuss another useful result from Theorem 2, that is, to generate a random DAG with a given rank $r$ and a properly specified average degree $k$, as shown in Algorithm 1. The first part of Algorithm 1f is to sample a number $N$, representing the total number of edges, from a binomial distribution $B(d(d-1)/2, p)$ where $p = k/(d-1)$. If $N < r$, Algorithm 1 would return FAIL since a graph with $N < r$ edges could never have rank $r$. Otherwise, an initial graph with $r$ edges and rank $r$ is sampled, which can be done by choosing the edges such that no two of them share the same head point or the same tail point, that is, each row and each column of the corresponding adjacency matrix have at most one nonzero entry. Then Algorithm 1 samples an edge from the set $R$ that contains all possible edges and checks whether adding this edge to the graph changes the size of the minimum head–tail vertex cover. If not, the edge will be added to the graph; otherwise, it will be removed from $R$. This is because if a graph $G$ is a super-graph of another graph $H$, then the size of the minimum head–tail cover of $G$ is no less than that of $H$. We repeat the above sampling procedure until there is no edge in $R$ or the number of edges in the resulting graph reaches $N$. If the latter happens, the algorithm will return the generated graph; otherwise, it will return FAIL.

Note that the algorithm may not return a valid graph if the desired number $N$ of edges cannot be reached. This could happen if the input rank is too low while the input average degree is too high. Fortunately, with our experiment settings, we find it rare for Algorithm 1 to fail to return a desired graph. In fact, there exist some connections between the maximum rank and the number of edges in the graph, according to Theorem 2. Intuitively, if the graph is dense, then we need more vertices to cover all the edges. Thus, the size of the minimum head–tail vertex cover tends to be large. Explicitly providing a formula to characterize these two graph parameters is an interesting future direction.

IV. Exploiting Low-Rank Assumption in Causal Structure Learning

Thanks to the studies of low-rankness in various contexts, we can adapt gradient-based causal structure learning methods with little extra effort based on existing techniques. We remark that our adaptations with the low-rank assumption are not restricted to a particular DAG learning algorithm; other gradient-based methods may incorporate one of the modifications too, if they have similar formulations and optimization procedures.

**Matrix Factorization:** Since the weighted adjacency matrix $W$ is explicitly optimized in the first class of methods, we can apply the matrix factorization technique (see [39], [40], [41]). Specifically, with an estimate $\hat{r}$ for the graph rank, we can factorize $W$ as $W = UV^T$ with $U, V \in \mathbb{R}^{d \times \hat{r}}$. Problem (1) is then to optimize $U$ and $V$, which minimizes the score function under the DAG constraint and has the same solution $W$ (obtained from $UV^T$) as the original one if $\hat{r}$ is greater than or equal to the true rank. Furthermore, if $\hat{r} \ll d$, we have a much reduced number of parameters to optimize.

**Nuclear Norm:** For the second class of methods, the adjacency matrix $W(\theta)$ is not an explicit parameter to be optimized. In such a case, we can add a nuclear norm term $\lambda \|W(\theta)\|_*$, with $\lambda > 0$ being a tuning parameter, to the objective to induce low-rankness (see [42], [43], [44], [45], [46]). Notice that this approach is also feasible for the first class of methods, but we find that it does not work as well as the matrix factorization approach, possibly due to the singular...
value decomposition to compute the (sub-)gradient w.r.t. $W$ at each optimization step.

Both the matrix factorization and nuclear norm approaches enjoy favorable theoretical guarantees and empirical performance in many low-rank applications. Similar to existing cases, the optimization procedures in these structure learning methods are gradient-based and can easily handle the two adaptations. Below we describe details about the optimization procedures. It would be beneficial to tailor the optimization for causal structure learning, but this is beyond the scope of this article.

A. Optimization

For this part, we consider a dataset consisting of $n$ i.i.d. observations from $P(X)$ and consequently the expectations in problems (1) and (2) are replaced by empirical means. Denote the design matrix by $X \in \mathbb{R}^{d \times d}$, where each row of $X$ corresponds to an observation and each column represents a variable. Here, we use NOTEARS [9] (and similarly gradient-based optimization of DAG-penalized likelihood for learning linear DAG models (GOLEM) [70]) and Gran-DAG [13] from each class as examples and will describe their low-rank versions in the following. Other gradient-based methods and their optimization procedures can be similarly modified to incorporate the low-rank assumption.

1) NOTEARS With Low-Rank Assumption: Following Section IV as well as the work of NOTEARS [9], the optimization problem can be written as

$$\min_{U,V} \frac{1}{2n} \|X - XUV^T\|_F^2, \quad \text{s.t.} \quad g(UV^T) = 0 \quad (3)$$

where $U, V \in \mathbb{R}^{d \times d}$, $\| \cdot \|_F$ denotes the Frobenius norm, and $g(\cdot)$ was introduced in Section II-B to induce a DAG. This problem can then be solved by standard numeric optimization methods such as the augmented Lagrangian method [71]. In particular, the augmented Lagrangian is given by

$$L_p(U, V, \alpha) = \frac{1}{2n} \|X - XUV^T\|_F^2 + \alpha g(UV^T) + \frac{\rho}{2} |g(UV^T)|^2$$

where $\alpha$ is the Lagrange multiplier and $\rho > 0$ is the penalty parameter. The optimization procedure is summarized in Algorithm 2, similar to [9, Algorithm 1].

The unconstrained subproblem in Step 2 can be solved by existing optimization methods such as limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm (L-BFGS) and Newton conjugate gradient method. Notice that the DAG constraint may not be satisfied exactly using iterative numeric methods, so it is a common practice to pick a small tolerance, followed by a thresholding procedure on the estimated entries to obtain an exact DAG. This heuristic is made possible by virtue of the DAG penalty term, which pushes the cycle-inducing edges to small values. In our implementation, we choose $U_0$ and $V_0$ to be the first $\hat{r}$ columns of the $d \times d$ identity matrices. Other parameter choices are: $\alpha_0 = 0, \epsilon = 0.25, \epsilon = 10^{-6}$, and $w = 0.3$, similar to those used in related methods on the same datasets (see [9], [10], [55]). The chosen threshold $w = 0.3$ works well in our experiments as well as in the experiments of related works that use the same data model.

Algorithm 2: Optimization Procedure for NOTEARS-Low-Rank

**Require:** Design matrix $X$, starting point $(U_0, V_0, \alpha_0)$, rate $c \in (0, 1)$, tolerance $\epsilon > 0$, and threshold $w > 0$.

**Ensure:** Locally optimal parameter $W^*$.

1: for $t = 1, 2, \ldots$ do
2: $U_{t+1}, V_{t+1} \leftarrow \arg \min_{U,V} L_p(U, V, \alpha_t)$ with $\rho$ such that $g(U_{t+1}V_{t+1}^T) < \epsilon g(U_tV_t^T)$.
3: $\alpha_{t+1} = \alpha_t + \rho g(U_{t+1}V_{t+1}^T)$.
4: if $g(U_{t+1}V_{t+1}^T) < \epsilon$ then
5: Set $U^* = U_{t+1}$ and $V^* = V_{t+1}$.
6: break
7: end if
8: end for
9: Set $W^* = (U^*V^*) \odot 1(|U^*V^*| > w)$.
10: return $W^*$

In case the thresholded matrix is not a DAG, one can further increase the threshold until the resulting matrix corresponds to a DAG.

After obtaining $W^*$, we apply an additional pruning step: we use linear regression to refit the dataset based on the structure indicated by $W^*$ and then apply another thresholding (with the same threshold as the previous thresholding procedure) to the refitted weighted adjacency matrix. The additional pruning step is applied to the original NOTEARS, which also improves its performance with lower structural Hamming distance (SHD), particularly for large and dense graphs.

Note that we do not include the $\ell_1$ penalty term w.r.t. $UV^T$ in the above version, for the following reasons: 1) the thresholding procedure can also control false discoveries; 2) we consider relatively sufficient data for the experiments and NOTEARS with thresholding has been shown in [9] to perform consistently well even when the graph is sparse; and 3) we are more concerned with relatively large and dense graphs, so a sparsity assumption may be harmful, as shown also in [9]. Nevertheless, we include NOTEARS with $\ell_1$ penalty in the first and last experiments in Section V for more information regarding the role of sparsity, which verifies the advantage of NOTEARS without $\ell_1$ penalty over that with $\ell_1$ penalty for relatively dense graphs.

2) GOLEM With Low-Rank Assumption: Instead of the hard acyclicity constraint, GOLEM [70] formulates causal structure learning with linear data models with a soft one, in the following form:

$$\min_{W} \frac{1}{2n} \log(\|X - XW\|_F^2) + \lambda_1 \|W\|_1 + \lambda_2 tr(e^{W^T}W)$$

The authors show that such a formulation works well on linear Gaussian data models with equal noise variances, even for very large problems. As such, GOLEM and its low-rank version (as described below) will be used for the large problems with 800 nodes in our experiment in Section V-A. For linear Gaussian models with nonequal noise variances, the authors replace log($\|X - XW\|_F^2$) with $\sum_{i=1}^d (\log(\sum_{j=1}^d (X_{ij}^{(ij)} - W_{ij}^{(ij)}))^2))$, Where $W_{ij}$ is the $i$th column of $W$, $X_{ij}$ is the $i$th observation, and $X_{ij}^{(ij)}$ is the $i$th entry in the $j$th observation.
Similar to NOTEARS, GOLEM-low-rank is then given by
\[
\min_{U, V} \frac{1}{2n} \|X - UV^T\|_F^2 + \lambda_1 \|UV^T\|_1 + \lambda_2 \left( e^{(U^TV)(e^TU)} \right)
\]
which can be solved in a similar manner to Algorithm 2. The estimate \( W^* \) is then calculated by the solutions \( U^* \) and \( V^* \), that is, \( W^* = U^*V^{*T} \). The same postprocessing procedure of the original GOLEM is then applied. Here, the choices of \( \lambda_1, \lambda_2 \), and the threshold are made in the same way as in [70].

3) Gran-DAG With Low-Rank Assumption: We next consider a low-rank version of GraN-DAG. The optimization problem can be written as
\[
\min_{\theta} \frac{1}{n} \sum_{i=1}^{d} \log p(X_i^{(l)} | \text{pa}(X_i, W(\theta))^{(l)}; \theta) + \lambda_1 \|W(\theta)\|_1
\]
s.t. \( g(W(\theta)) = 0 \)

where \( X_i^{(l)} \) is the \( l \)th sample of variable \( X_i \) and \( \text{pa}(X_i, W(\theta))^{(l)} \) means the \( l \)th sample of \( X_i \)’s parents indicated by the adjacency matrix \( W(\theta) \). Here, \( \theta \) denotes the parameters of neural networks and \( W(\theta) \) with nonnegative entries is obtained from the neural network path products.

The above problem can be solved similarly using augmented Lagrangian. The procedure is similar to Algorithm 2 and is the same as that used by GraN-DAG, with slight modifications: 1) the subproblem in Step 2 is approximately solved using first-order methods and 2) the thresholding at Step 9 is replaced by a variable selection method proposed by Bühlmann et al. [72]. The same variable selection or pruning method is adopted by two other benchmark methods causal additive models (CAM) and NOTEARS-MLP in our experiment. Refer to [13] and [72] for further details.

B. Hyperparameters

In practice, we need to pick hyperparameters \( \hat{r} \) and \( \lambda \) for the low-rank modifications. For the first class of methods, if we know that an estimate \( \hat{r} \) is identical or close to the true rank, then we can directly plug it into the factorized method. While this case may be rare in practice, we use it as a sanity check of the low-rank methods (see Section V-A). Perhaps, it is more often to know only a range of possible ranks based on certain prior information, for example, some structural information in Theorems 1–4, or when the underlying DAG is SF and a rough estimate of the average degree and power is available (as experimentally shown in Section V-C). Similar to selecting other hyperparameters, we may determine \( \hat{r} \) assisted by a validation dataset (or by cross-validation if the observed dataset is not sufficiently large), or try different choices and then apply a traditional score-based method where the search space is restricted to the resulting DAGs. As shown in Section V-B, the modified algorithm performs consistently well over a large range of rank parameters, so we may evenly pick 5–10 choices within the range; this strategy is further verified by our experiments in Section V.

For the second class of methods with a nuclear norm regularization, we may also apply the validation approach with several penalty weights. While the graph rank does not directly indicate an appropriate choice and a fixed set of candidate weights may not work well for all problems, the existing methods for selecting similar hyperparameters (like the weights of \( \ell_1 \) or \( \ell_2 \) penalties) can be very useful here. In our experience, choosing from \{0.1, 0.3, 0.5, 0.8, 1, 2, 5\} works reasonably well, as shown in Section V-D.

Remark 2: When it is not certain that the underlying DAG is a low rank, we can also include the original algorithms in the validation approach. Section V-D empirically verifies this strategy with nonlinear SCMs when the graphs to be learned are not low rank.

V. Experiments

This section reports empirical results of the low-rank adaptations of existing methods, compared with their original versions. We focus on NOTEARS [9] and GOLEM [70] for linear data models by adopting the matrix factorization approach, denoted, respectively, as NOTEARS-low-rank and GOLEM-low-rank and use the nuclear norm approach in combination with GraN-DAG [13] for a nonlinear SCM. These modifications have been described in Section IV-A, and an implementation has been released in the gCastle package [73]. For more information, we also include several benchmark methods: fast GES [8], PC [3], MMHC [16], the polynomial-time method from [18], DirectLiNGAM [74], and ICA-LiNGAM [57] that are specifically designed with non-Gaussian noises, for linear cases; and DAG-GNN [10], NOTEARS-MLP [14], and CAM [72] for the nonlinear SCMs. Here, Fast GES and PC are called from the py-causal package2 and MMHC is called from the bnlearn package. The rest methods have available implementations released in the respective papers and we use default hyperparameters unless otherwise stated.

We consider randomly sampled DAGs with specified ranks generated by Algorithm 1, SF graphs, and two real networks. For the linear data model of the following form:
\[
X_i = \sum_{X_j \in \text{pa}(X_i, G)} W(j, i)X_j + \epsilon_i, \quad i = 1, 2, \ldots, d
\]
we follow existing works (such as NOTEARS and DAG-GNN) to draw the weights from Unif([-2, -0.5] \cup [0.5, 2]) and assume \( \epsilon_i \)’s to be jointly independent noises with three cases: 1) standard Gaussian distributions; 2) standard exponential distributions; and 3) zero-mean Gaussian distributions with variances drawn from Unif([0.5, 2]). For nonlinear SCM, we consider the following model which is also used in the works of CAM and NOTEARS-MLP:
\[
X_i = f_i(\text{pa}(X_i, G)) + \epsilon_i, \quad i = 1, 2, \ldots, d
\]
where \( \epsilon_i \) are jointly independent standard Gaussian noises and \( f_i \) are functions sampled from Gaussian processes with RBF kernel of bandwidth one. Below, we mainly report SHD (or SHD-CPDAG, the SHD between CPDAGs of the estimated graph and ground truth, for nonidentifiable cases), which takes into account both false positives and false negatives, and a

1https://github.com/huawei-noah/trustworthyAI
2https://github.com/bd2kccd/py-causal
3https://CRAN.R-project.org/package=bnlearn

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
smaller SHD indicates a better estimate [16]. Some more evaluation metrics are provided in Appendix B.

A. Linear Data Models With Rank-Specified Graphs

We use linear data models on rank-specified graphs, with a number of nodes \( d \in \{100, 300\} \), rank \( r = \lceil 0.1d \rceil \), and average degree \( k \in \{2, 4, 6, 8\} \). This experiment serves as a sanity check of the low-rank approach—the true rank is assumed to be known and is used as the rank parameter \( \hat{r} \) in NOTEARS-low-rank and GOLEM-low-rank. From each data model, we then generate \( n = 3000 \) observations. We repeat 20 times over different seeds for each experimental setting. The experiments are run on a Linux workstation with a 64-core Intel Xeon 3.20-GHz CPU, Nvidia Quadro RTX 6000 GPU, and 128-GB RAM.

**Identifiable Case:** We first consider additive noises following standard Gaussian, which is known to be identifiable [58]. For a better visualization, Fig. 2 only reports the average SHDs, while the true positive rate (TPR), false discovery rate (FDR), and running time are left to Appendix B. We observe that the performance of NOTEARS and GOLEM degrades when the graph becomes dense and the maximum in-degree increases; in particular, the maximum in-degrees are \( [11.0 \pm 1.6, 22.0 \pm 2.2, 33.9 \pm 4.1, 44.7 \pm 6.3] \) for 100-node graphs with average degrees \( [2, 4, 6, 8] \), respectively. The low-rank versions can greatly improve the performance of original algorithms, reducing SHDs by at least half. The fast GES, PC, and MMHC have much higher SHDs. For the polynomial-time method in [18], it has a similar performance as NOTEARS-low-rank and GOLEM-low-rank with sparse graphs of degree 2, but degrades much when the graph becomes denser; specifically, for 100-node graphs, its SHDs are \( 1.25 \pm 2.4 \) and \( 307.25 \pm 52.75 \) for degrees 2 and 8, respectively. More detailed results are provided in Table I in the Appendix.

For more information regarding the role of sparsity, we include NOTEARS with an \( \ell_1 \) penalty, denoted as NOTEARS-L1, where the \( \ell_1 \) penalty weight is chosen from \( \{0.01, 0.02, 0.05, 0.1, 0.2, 0.5\} \). Instead of relying on an additional validation dataset, we treat NOTEARS-L1 favorably by picking the lowest SHD obtained from different weights for each dataset. Yet NOTEARS-L1 is slightly better than NOTEARS when the average degree is 2 and is largely outperformed with relatively dense graphs. This observation was also reported in [9]. We conjecture that it is because our experiments consider relatively sufficient data and dense graphs. The utility of low-rank adaptations for relatively dense graphs is also verified with additive exponential noises, where we include DirectLiNGAM for 100-node graphs and ICA-LiNGAM for 300-node graphs (because DirectLiNGAM is too slow for the latter case).

**Nonidentifiable Case:** We proceed to nonidentifiable linear data models with zero-mean Gaussian noises whose variances are drawn from \( \text{Unif}([0.5, 2]) \). We adopt a version of GOLEM that is specifically developed for nonequal variances and use SHD-CPDAG as the metric. We report the average SHD-CPDAGs for 100-node graphs with \( r = 10 \) and average degrees \( k \in \{2, 4, 6, 8\} \), which are \( [33.5 \pm 29.1, 144.2 \pm 32.4, 221.9 \pm 47.7, 283.7 \pm 64.1] \) for GOLEM and \( [3.9 \pm 5.9, 38.0 \pm 40.5, 116.5 \pm 73.8, 200.9 \pm 79.1] \) for GOLEM-low-rank. This again verifies the utility of a low-rank adaptation.

**Different Sample Size:** We conduct an empirical analysis with different sample sizes. Again, we use rank-specified random graphs (sampled according to Algorithm 1) with \( d = 100 \) nodes, degree \( k = 8 \), rank \( r = 10 \), and linear Gaussian SEMs. We also assume that the true rank is known. We fix the rank parameter \( \hat{r} = 10 \) and use different sample sizes ranging from 200 to 5000. As reported in Fig. 3, NOTEARS-low-rank performs reasonably well when the sample size is small and tends to have a better performance with more samples.

**Higher Rank:** This experiment considers graphs of higher ranks. We use rank-specified random graphs with \( d = 100 \) nodes and rank \( r \in \{30, 35, 40, 45, 50\} \) on linear Gaussian data models. The results are shown in Fig. 4 (left and middle), with average degrees 2 and 8, respectively. We observe that when the rank of the underlying graph becomes higher, the advantage of NOTEARS-low-rank over NOTEARS decreases. Nonetheless, NOTEARS-low-rank with rank \( r = 50 \) is still comparable to NOTEARS and has a lower average SHD after removing outlier SHDs using the interquartile range rule. A possible reason is that the maximum in-degree of the rank-specified graphs, given a fixed average degree, tends to decrease when the rank becomes higher; in particular, for average degree 8, the maximum in-degrees are \( [22.6 \pm 2.2, 19.7 \pm 3.4, 17.8 \pm 2.6, 17.5 \pm 2.4, 14.6 \pm 1.6] \) for ranks \( \{30, 35, 40, 45, 50\} \), respectively. We then increase the average degree to 14 and 20, while fixing rank \( r = 50 \); the maximum
in-degrees become \(\{4.4 \pm 0.8, 14.6 \pm 1.6, 25.4 \pm 2.7, 34.6 \pm 3.2\}\) for average degrees \(\{2, 8, 14, 20\}\), respectively. As shown in Fig. 4 (right), both NOTEARS and NOTEARS-low-rank have increasing SHDs, as the graphs are of both high in-degrees and high ranks.

**Larger Graph:** Finally, we test the proposed approach for larger problems with 800 nodes, average degree 8, and rank 80. We run GOLEM and its low-rank version due to its computational efficiency for large problems. GOLEM achieves SHD 3698.9 \(\pm\) 30.9, compared with 284.2 \(\pm\) 380.0 for GOLEM-low-rank. The high variance is due to a dataset where GOLEM-low-rank performs poorly. Excluding this dataset gives SHD being 158.2 \(\pm\) 140.2.

**B. Sensitivity of Rank Parameters and Validation**

Section V-A assumed that the true rank is known, serving as an initial check of the low-rank modification. In this experiment, we conduct an empirical analysis with different rank parameters \(\hat{r} \in \{5, 8, 10, 15, 20\}\) for the linear Gaussian data model on rank-specified graphs with 100 nodes, degree 8, and rank 10. As reported in Fig. 5, NOTEARS-low-rank performs the best when the rank parameter is identical to the true rank. Compared with NOTEARS on the same datasets, the low-rank version performs well across a range of rank parameters.

For our purpose, we further include the validation-based approach that was described in Section IV. We choose 2000 samples as the training dataset and the rest as the validation dataset. We assume only a range of possible ranks available, and here we simply consider that the rank parameters are also taken from \(\{5, 8, 10, 15, 20\}\). We apply NOTEARS-low-rank with each of them and then evaluate each learned DAG using the validation dataset. The DAG with the best score would be selected as our estimate. We see that the validation-based approach has almost the same performance as that of the true rank. Although this validation approach increases the total running time which depends on the number of candidate rank parameters, we believe that it is acceptable given the gain in accuracy and the fact that this strategy is commonly adopted for tuning other hyperparameters like the \(\ell_1\) penalty weight.

**C. Linear Data Models With Scale-Free Graphs**

We continue to consider SF graphs with \(d = 100\) nodes, average degrees \(k \in \{6, 8\}\), and power \(\gamma = 2.5\). Such graphs generally have hubs, and if we know a priori a rough estimate of \(k\) and \(\gamma\), we may use simulations (see Fig. 1) to infer the range of possible ranks with high confidence. In this experiment, we pick \(\hat{r} \in \{15, 20, 25, 30, 35, 40\}\) for NOTEARS-low-rank. For clarity of presentation, we only show the experimental results of some rank parameters in Fig. 6, along with the validation-based approach. The low-rank adaptations again result in improved performance for causal structure learning. We also note that the validation-based approach achieves a competitive result. Using Student’s t-test, the validation-based approach is better than NOTEARS, with a significance level of 0.1.

**D. Nonlinear SCMs**

For nonlinear data models, we pick rank-specified graphs with 50 nodes, rank 5, and average degree \(k \in \{2, 4, 6, 8\}\). To our knowledge, the selected benchmark methods CAM, NOTEARS-MLP, and GraN-DAG are state-of-the-art methods in this data model which is known to be identifiable [59]. As a demonstration of the low-rank assumption, we apply the nuclear norm approach to GraN-DAG and choose from \(\{0.3, 0.5, 1\}\) as penalty weights. For the validation approach, we use the same splitting ratio as in Section V-B and consider more penalty weights from \(\{0.1, 0.3, 0.5, 0.8, 1, 2, 5\}\). The learned graph that achieves the best score on the validation...
dataset is chosen as the final estimate. Fig. 7 (and Appendix C with more detailed results) shows that adding a nuclear norm can improve the performance of GraN-DAG across a large range of weights when the graph is relatively dense. For degree 8, the low-rank version with validation achieves an average SHD of 77.4, while the SHDs of CAM, NOTEARS-MLP, and original GraN-DAG are 131.9, 119.4, and 109.4, respectively.

We further check how the low-rank method behaves when the underlying graph is not a low rank. We consider regular ER graphs with 50 nodes and the same data models. With a large penalty weight, the learned DAGs tend to have high SHDs in this case. The original GraN-DAG achieves SHD 3.4 ± 2.1 and 44.2 ± 12.4 for average degrees 2 and 8, respectively, while the validation-based approach, including original GraN-DAG, has similar SHDs 2.2 ± 1.2 and 48.2 ± 11.2, respectively. Indeed, the final estimates are mostly obtained by the original GraN-DAG or the low-rank version with small penalty weights.

E. Real Network

We apply the proposed method to the arth150 gene network available at the Bayesian network repository, which is a DAG containing 107 genes and 150 edges. The maximum rank is 40 and the maximum in-degree is 6. Since the real dataset has only 22 samples, we instead use simulated data from a model containing 107 genes and 150 edges. The maximum rank is 40 and the maximum in-degree is small. Even so, the low-rank version has a much lower SHD than the original NOTEARS.

We next consider another benchmark network pathfinder, also from the Bayesian network repository. The network has 109 nodes and 195 edges, and the average degree is 3.57. The maximum rank of this graph is 31, and the lower bound on the minimum rank computed using our proposed method is 26. We adopt a similar experimental setup and pick \( \hat{r} \) from \( \{27, 31, 35\} \). As reported in Fig. 9, the low-rank version has a much lower SHD than the original NOTEARS.

VI. Conclusion

Causal DAG models have been applied in many fields of machine learning and data science [75], [76], such as computer vision [77] and algorithmic fairness [78], to improve explainability and interpretability. This article studies the problem of learning DAGs from observational data and makes a first step toward incorporating low-rankness into causal DAG learning, with which we aim to deal with relatively large and dense causal DAGs. Empirically, we show that the low-rank adaptations perform noticeably better than existing algorithms when the low-rank condition is satisfied and also deliver competitive performance when the rank is not restricted to low. Our theoretical results provide a better understanding of low-rank graphs in terms of graphical features and hence of the low-rank assumption itself.

APPENDIX

A. Proofs

For ease of presentation, we occasionally use index \( i \) to represent variable \( X_i \) in Appendixes B and C.

1) Proof of Theorem 1: Let \( \mathcal{G} = (V, E) \). Consider an equivalence relation, denoted by \( \sim \), among vertices in \( V \) defined as follows: for any \( X_i, X_j \in V \), \( X_i \sim X_j \) if and only if \( l(X_i) = l(X_j) \) and \( X_i \) and \( X_j \) are connected. Here, connected means that there is a path between \( X_i \) and \( X_j \). Below, we use \( C(X_i) \) to denote the equivalence class containing \( X_i \). Next, we define a weak ordering \( \pi \) on \( V/\sim \), that is, the equivalence classes induced by \( \sim \), by letting \( C(X_i) \succeq \pi \ C(X_j) \) if and only if \( l(X_i) \geq l(X_j) \). Then, we extend \( \succeq \pi \) to a total ordering \( \rho \) on \( V/\sim \). The ordering \( \rho \) also induces a weak ordering (denoted by \( \tilde{\rho} \) on \( V \): \( X_i \succeq \tilde{\rho} X_j \) if and only if \( C(X_i) \geq \rho C(X_j) \). Finally, we extend \( \tilde{\rho} \) to a total ordering \( \gamma \) on \( V \). It can be verified that \( \gamma \) is a topological ordering of \( \mathcal{G} \), that is, if we relabel the vertices according to \( \gamma \), then \( X_i \in pa(X_j, \mathcal{G}) \) if and only if \( i > j \) and \( X_i \) and \( X_j \) are adjacent, and the adjacency matrix of \( \mathcal{G} \) becomes lower triangular.

Assume that the vertices of \( \mathcal{G} \) are relabeled according to \( \gamma \) and we will consider the binary adjacency matrix \( A \) of
the resulting graph throughout the rest of this proof. Note that relabeling is equivalent to applying a permutation onto the adjacency matrix, which does not change the rank. Let \( V_0 = \{1, 2, \ldots, k_1 - 1\} \) for some \( k_1 \geq 2 \). Then the \( k_1 \)th row of \( A \), denoted by \( A(k_1, \cdot) \), is the first nonzero row vector of \( A \). Letting \( S = \{A(k_1, \cdot)\} \), then \( S \) contains a subset of linearly independent vector(s) of the first \( k_1 \) rows of \( A \). Suppose that we have visited the first \( m \) rows of \( A \) and \( S = \{A(k_1, \cdot), A(k_2, \cdot), \ldots, A(k_t, \cdot)\} \) contains a subset of linearly independent vector(s) of the first \( m \) rows of \( A \), where \( k_1 \leq m < d \). If \( X_{m+1} \approx X_k \), then we add \( A(m + 1, \cdot) \) to \( S \); otherwise, we keep \( S \) unchanged. We claim that the vectors in \( S \) are still linearly independent after the above step. Clearly, if we do not add any new vector, then \( S \) contains only linearly independent vectors. To show the other case, note that if \( l(X_{m+1}) > l(X_k) \geq \cdots \geq l(X_e) \), then there is an index \( i \in V \) such that \( A(m + 1, i) \neq 0 \), by the definition of height. Since \( l(X_{m+1}) > l(X_k) \), we have \( l(X_k) \leq l(X_{m+1}) - 1 \) and thus \( A(k_j, i) = 0 \) for all \( j = 1, 2, \ldots, t \). Therefore, \( A(m + 1, \cdot) \) cannot be linearly represented by \( \{A(k_j, \cdot); j = 1, 2, \ldots, t\} \) and the vectors in \( S \) are linearly independent. On the other hand, if \( l(X_{m+1}) = l(X_k) \), then the definition of the equivalence relation \( \sim \) implies that \( X_{m+1} \) and \( X_k \) are disconnected, which means that \( X_{m+1} \) and \( X_k \) do not share a common child in \( V \). Consequently, there is an index \( i \in V \) such that \( A(m + 1, i) \neq 0 \) but \( A(k_i, i) = 0 \). Similarly, we can show that \( A(k_j, i) = 0 \) for all \( j = 1, 2, \ldots, t \). Thus, the vectors in \( S \) are still linearly independent.

After visiting all the rows in \( A \), the number of vectors in \( S \) is equal to \( \sum_{i=1}^{l(G)} |C(G_{s_i})| \) based on the definition of \( \sim \). The second inequality can be shown by noting that \( |C(G_{s_i})| \) has at least one element. The proof is complete.

2) Proof of Theorem 2: Denote the directed graph by \( G = (V, E) \). [50, Th. 1] showed that \( \max \{\text{rank}(W); \ W \in W_A\} \) is equal to the maximum number of nonzero entries of \( A \), no two of which lie in a common row or column. Therefore, it suffices to show that the latter quantity is equal to the size of the minimum head–tail vertex cover. Let \( V' = V_0 \cup V_1 \), where \( V_0 = V \setminus \{0\} = \{(X, 0); X \in V\} \) and \( V_1 = \{1\} = \{(X, 1); X \in V\} \). Now define a bipartite graph \( B = (V', E') \), where \( E' = \{(X, 0) \rightarrow (X, 1); (X, X_j) \in E\} \). Denote by \( M \) a set of nonzero entries of \( A \) so that no two entries lie in the same row or column. Notice that \( M \) can be viewed as an edge set and no two edges in \( M \) share a common endpoint. Thus, \( M \) is a matching of \( B \). Conversely, it can be shown by edge set arguments that any matching of \( B \) corresponds to a set of nonzero entries of \( A \), no two of which lie in a common row or column. Therefore, \( \max \{\text{rank}(W); \ W \in W_A\} \) equals the size of the maximum matching of \( B \) and further the size of the minimum vertex cover of \( B \) according to König's theorem. Note that any vertex cover of \( B \) can be equivalently transformed to a head–tail vertex cover of \( G \), by letting \( H \) and \( T \) be the subsets of the vertex cover containing all variables.
in $V'_0$ and of the vertex cover containing all variables in $V'_1$, respectively. Thus, $\max\{\text{rank}(W), W \in W_{\mathcal{A}}\}$ is equal to the size of the minimum head–tail vertex cover.

3) Proof of Theorem 3: We start with the first inequality in Proof of Theorem 3. Let $h_1, \ldots, h_p$ denote the heights where $|V_i| < |\text{ch}(V_i)|$, and $t_1, \ldots, t_q$ the heights where $|V_i| > |\text{ch}(V_i)|$. Let $H = \bigcup_{i=1}^{p} V_{h_i}$ and $T = \bigcup_{i=1}^{q} V_{t_i}$. It is straightforward to see that $(H, T)$ is a head–tail vertex cover. Thus, Proof of Theorem 3 holds according to Theorem 2.

The second inequality can be shown similarly and its proof is omitted. For the third inequality, let $m = \arg\max\{|V_i|: 0 \leq s \leq l(G)|$ and define $H = \bigcup_{i=m} V_i$ and $T = \bigcup_{i<m} V_i$. Then $(H, T)$ is also a head–tail vertex cover and the third inequality follows from Theorem 2, too.

4) Proof of Theorem 4: Notice that Theorem 2 holds for all directed graphs. This theorem then follows by treating the skeleton and the moral graph as directed graphs with loops, that is, an undirected edge $X_i \rightarrow X_j$ is treated as two directed edges $X_i \rightarrow X_j$ and $X_j \rightarrow X_i$.

B. Detailed Empirical Results for Experiment 1 With Linear Gaussian SEMs

Table I reports detailed results including TPRs, FDRs, SHDs, and running time on rank-specified graphs with linear data models where the noises follow standard Gaussian distribution. Here, the true rank is assumed to be known and is used as the rank parameter in NOTEARS-low-rank and GOLEM-low-rank. For both GOLEM and GOLEM-low-rank, we set $30000$ and $100000$ optimization steps in the stochastic first-order method for 100- and 300-node graphs, respectively. We also include (fast) GES, MMHC, and PC. However, PC is too slow since some nodes may have a high in-degree (i.e., hubs) in large, dense, and low-rank graphs. For the same reason, the skeleton may not be correctly estimated by MMHC, which has a similar performance to that of GES. Therefore, we only include the results of GES for comparison. We treat GES favorably by regarding undirected edges as true positives if the true graph has a directed edge in place of the undirected ones.

C. Detailed Results for Experiment 4 With Nonlinear SEMs

Table II reports the detailed SHDs for each method in Section V-D. We also mark in bold the best results from methods with or without low-rank modifications.

---

**References**

[1] J. Pearl, *Causality: Models, Reasoning and Inference*. New York, NY, USA: Cambridge Univ. Press, 2009.

[2] J. Peters, D. Janzing, and B. Schölkopf, *Elements of Causal Inference—Foundations and Learning Algorithms*. Cambridge, MA, USA: MIT Press, 2017.

[3] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, Prediction, and Search*, 2nd ed. Cambridge, MA, USA: MIT Press, 2000.

[4] C. Meek, “Causal inference and causal explanation with background knowledge,” in *Proc. 11th Conf. Uncertainty Artif. Intell.*, 1995, pp. 1–8.

[5] J. Zhang, “On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias,” *Artif. Intell.*, vol. 172, nos. 16–17, pp. 1873–1896, Nov. 2008.

[6] D. M. Chickering, “Learning Bayesian networks is NP-complete,” in *Learning From Data*. Cham, Switzerland: Springer, 1996.

[7] D. M. Chickering, “Optimal structure identification with greedy search,” *J. Mach. Learn. Res.*, vol. 3, pp. 507–554, Nov. 2002.

[8] J. Ramsey, M. Glymour, R. Sanchez-Romero, and C. Glymour, “A million variables and more: The fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images,” *Int. J. Data Sci. Anal.*, vol. 3, no. 2, pp. 121–129, Mar. 2017.

[9] X. Zheng, B. Aragam, P. Ravikumar, and E. P. Xing, “DAGs with NO TEARS: Continuous optimization for structure learning,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1–16.

[10] Y. Yu, J. Chen, T. Gao, and M. Yu, “DAG-GNN: DAG structure learning with graph neural networks,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 1–4.

[11] I. Ng, S. Zhu, Z. Chen, and Z. Fang, “A graph autoencoder approach to causal structure learning,” 2019, arXiv:1911.07420.

[12] I. Ng, S. Zhu, Z. Fang, H. Li, Z. Chen, and J. Wang, “Masked gradient-based causal structure learning,” in *Proc. SIAM Int. Conf. Data Mining (SDM)*, 2022, pp. 424–432.

[13] S. Lachapelle, P. Brouillard, T. Deleu, and S. Lacoste-Julien, “Gradient-based neural DAG learning,” in *Proc. Int. Conf. Represent. Learn. (ICLR)*, 2020, pp. 1–16.

[14] X. Zheng, C. Dan, B. Aragam, P. Ravikumar, and E. P. Xing, “Learning sparse nonparametric DAGs,” in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2020, pp. 3414–3425.

[15] N. R. Ke et al., “Learning neural causal models from unknown interventions,” 2019, arXiv:1910.01075.

[16] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, “The max-min hill-climbing Bayesian network structure learning algorithm,” *Mach. Learn.*, vol. 65, no. 1, pp. 31–78, Oct. 2006.

[17] A. Ghoshal and J. Honorio, “Learning identifiable Gaussian Bayesian networks in polynomial time and sample complexity,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 1–16.

[18] A. Ghoshal and J. Honorio, “Learning linear structural equation models in polynomial time and sample complexity,” in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2018, pp. 1466–1475.

[19] V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky, “Latent variable graphical model selection via convex optimization,” in *Proc. 48th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2010, pp. 1610–1613.

[20] C. Smith, F. Wood, and L. Paninski, “Low rank continuous-space graphical models,” in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2012, pp. 1064–1072.
Zhuangyan Fang received the B.S. degree in mathematics from Sichuan University, Chengdu, China, in 2016, and the Ph.D. degree in statistics from Peking University, Beijing, China, in 2021. He is currently a Data Scientist with the Department of Information Technology, Xiaomi, Beijing. During his doctoral studies, his study interests were causality, graphical models, and machine learning. Currently, he is working on explainable prediction models.

Shengyu Zhu received the B.E. degree in electrical engineering from the Beijing Institute of Technology, Beijing, China, in 2010, and the M.S. degree in mathematics and the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2016 and 2017, respectively. He joined Syracuse University in 2012. He was a Principal Researcher with the Huawei Noah’s Ark Laboratory, Hong Kong. His research interests include causality, machine learning, information theory, and statistical signal processing learning.

Jiji Zhang received the B.A. degree in logic from Peking University, Beijing, China, in 2000, and the Ph.D. degree in logic, computation, and methodology from Carnegie Mellon University, Pittsburgh, PA, USA, in 2006. He is currently a Professor with the Department of Philosophy, The Chinese University of Hong Kong, Hong Kong. His main research program spans a range of questions on causation and causal inference, including both epistemological and logical aspects of causal reasoning, and computational and statistical aspects of causal modeling.

Yue Liu received the B.S. and Ph.D. degrees in statistics from Peking University, Beijing, China, in 2014 and 2019, respectively. He is currently an Assistant Professor with the School of Statistics, Renmin University of China, Beijing. His current research interests include causality, graphical models, and trustworthy machine learning.

Zhitang Chen received the bachelor’s degree from the Department of Automation, Sun Yat-sen University, Guangzhou, China, in 2010, and the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2014. He is currently a Researcher with the Huawei Noah’s Ark Laboratory, Hong Kong. His research interests include kernel methods, deep learning, reinforcement learning, and multiagent systems and their applications to computer networks.

Yangbo He received the Ph.D. degree in statistics from Peking University, Beijing, China, in 2004. He is currently an Associate Professor with the School of Mathematical Sciences, Peking University. Since 2004, he has been working on causal discovery, causal inference, and quantitative finance.