Federated Learning of Generalized Linear Causal Networks

Qiaoling Ye, Arash A. Amini and Qing Zhou

Abstract—Causal discovery, the inference of causal relations among variables from data, is a fundamental problem of science. Nowadays, due to an increased awareness of data privacy concerns, there has been a shift towards distributed data collection, processing and storage. To meet the pressing need for distributed causal discovery, we propose a novel federated DAG learning method called *distributed annealing on regularized likelihood score* (DARLS) to learn a causal graph from data stored on multiple clients. DARLS simulates an annealing process to search over the space of topological sorts, where the optimal graphical structure compatible with a sort is found by distributed optimization. This distributed optimization relies on multiple rounds of communication between local clients and a central server to estimate the graphical structure. We establish its convergence to the solution obtained by an oracle with access to all the data. To the best of our knowledge, DARLS is the first distributed method for learning causal graphs with such finite-sample oracle guarantees. To establish the consistency of DARLS, we also derive new identifiability results for causal graphs parameterized by generalized linear models, which could be of independent interest. Through extensive simulation studies and a real-world application, we show that DARLS outperforms existing federated learning methods and is comparable to oracle methods on pooled data, demonstrating its great advantages in estimating causal networks from distributed data.

Index Terms—Causal graphs, federated learning, generalized linear models, simulated annealing, topological sorts.

1 Introduction

INFERRING causal relations among variables is a fundamental problem in many applications such as computational biology, medical science, social study, etc. It is tightly connected to various areas in statistics, including randomized experiments [1, 2], retrospective counterfactual reasoning [3], potential outcomes [4, 5], and probabilistic graphical models [6, 7]. A critical yet challenging step in causal inference via graphical models is the identification of cause-effect relations from data, namely the causal discovery problem, typically formulated as structure learning of a causal graph or its Markov equivalence class. As a primary causal model, causal graphs have been widely used in epidemiology [8, 9], pathophysiology [10], economics [11], and risk analysis [12] among many other domains. A causal graph is usually represented by a *directed acyclic graph* (DAG), where edges encode causal effects among variables (nodes). The probability density of a set of random variables $\{X_1, \ldots, X_p\}$ in a causal DAG $\mathcal{G}_0$ factorizes as

$$p(x_1, \ldots, x_p) = \prod_{j=1}^{p} p(x_j \mid \mathrm{PA}_j = pa_j), \tag{1}$$

where $\mathrm{PA}_j \subset \{X_1, \ldots, X_p\} \setminus \{X_j\}$ is the parent set of X_j with pa_j being its value. When the parents of X_j are set to pa_j by experiment, the conditional distribution $[X_j \mid \mathrm{PA}_j = pa_j]$ is also interpreted as the intervention distribution of X_j i.e. $[X_j \mid do(\mathrm{PA}_j = pa_j)]$ [13]. Since randomized experiments are not always feasible or available, various approaches have been put forward to learn causal DAGs from observational data; see [14, 15, 16] for recent reviews.

Department of Statistics and Data Science, University of California, Los Angeles, CA 90095, USA. Email: yeqiaoling@g.ucla.edu, aaamini@stat.ucla.edu, zhou@stat.ucla.edu (corresponding author).

1.1 Federated learning

In this work, we focus on the task of federated DAG learning, i.e., learning causal relations encoded by DAGs from distributed data, with a particular interest in non-Gaussian cases which encompass a wide variety of data types. Moreover, distributed learning of Gaussian linear DAG models can be achieved through simple updates of its sufficient statistics (Remark 1). Distributed data storage has been used for privacy protection when managing the copious amount of data generated everyday by government agencies, research institutions, medical centers, technology companies, etc [17]. These organizations often collect similar data, or data from the same population, and collaborate in diverse domains such as social, scientific, and business sectors. Notable examples include the exposure notification system by Google and Apple in 2021 for tracking COVID-19 exposure and a privacy-preserving analytics platform for health metrics collection [18], as well as small clinics pooling data from other facilities to achieve statistically significant results due to their limited datasets. Such collaborations necessitate strict privacy disclosures, ensuring that sensitive data held privately by each institution is not shared externally.

Federated learning proves to be an effective tool when preserving data privacy or when merging data from multiple sources is infeasible. This approach empowers local entities to collaboratively learn from decentralized data without the need for direct data sharing. As a result, federated learning not only addresses privacy and logistical concerns, but also facilitates the development of robust and generalized models by learning from a diverse range of data sources.

A straightforward method for federal learning to obtain a global estimate is to average local estimates, a technique known as *one-shot parameter averaging* [19]. This approach, however, falls short of achieving any desired level of accuracy compared to the global estimate based on all the data [20, 21]. To address these limitations, more sophisticated communication-efficient algorithms that adopt multiple communication rounds between local clients and a central server have been developed [22, 23]. Such algorithms are increasingly vital in distributed optimization for multi-agent systems, including electronic power systems, sensor networks, and smart manufacturing [24, 25].

1.2 Contribution of this work

Despite the methodological advances reviewed above, learning causal DAGs from data distributed across multiple local clients is still a challenging task. A major difficulty is how to integrate local information to efficiently estimate a global causal graph as if data across all local clients were accessible to the statistician. There are simple approaches that iterate over local datasets (once) and then combine the local graphs or the local p-values to form a global graph [26, 27, 28]. However, simple aggregation of local estimates, using this single-iteration approach, would not lead to an estimate close to the corresponding global estimate constructed if the combined data were accessible. Hereafter, we call such a global estimate an oracle solution.

Another challenge specific to our problem comes from the acyclicity constraint on DAGs. Obviously, simple average graphs are likely to fail this essential constraint and thus cannot provide meaningful causal interpretations. To overcome these difficulties, we propose a score-based learning method that carries out multiple rounds of communication to estimate DAGs from distributed data. Our objective function is equivalent to a regularized log-likelihood of the overall data, which has been shown to be effective in learning both continuous and discrete DAGs [29, 30, 31]. The central server proposes a candidate ordering π, where the score of π is evaluated via distributed optimization over DAGs compatible with π. Then, the candidate sort π is selected by simulated annealing. Because every DAG has at least one ordering, searching over the space of orderings ensures that the acyclicity constraint is always satisfied.

Our method is an example of federated learning, where a central server communicates with distributed local clients to learn a DAG. We show that the convergence rate of our federated estimate to the oracle estimate on the overall data is $O(\log(n)/\sqrt{m})$ for a fixed true DAG, where n is the total sample size across all local clients and m is the smallest local sample size (Theorem 1, Section 4.1). Therefore, even for a finite sample, as long as $\log(n)/\sqrt{m}$ is small, our distributed estimate will be essentially identical to the oracle solution, achieving the ideal efficiency while preserving data privacy. To the best of our knowledge, our approach is the first federated causal discovery method with such a nice theoretical guarantee. While this work was under review, two federated DAG learning methods were published [32, 33]. Neither of the two papers establishes such an oracle estimation guarantee. We will further elaborate on other technical differences in Section 3.4.

Another contribution of our work is the use of *generalized linear models* (GLMs) for local conditional distributions in (1), which brings several advantages to causal structure learning. Our proposed GLM DAG model is a flexible family for various data types beyond linear Gaussian models with equal variance and multi-logit models [31, 34]. The negative log-likelihood functions under commonly used GLMs are convex, which facilitates the optimization task. Furthermore, we show that GLM DAG models are identifiable under mild conditions (Proposition 1, Section 2), while other models, such as multinomial for discrete networks and Gaussian linear DAGs (with heterogeneous variance), are not identifiable in general [35, 36]. Under such identifiability, we establish the ℓ_2-consistency of a global maximizer DAG of our regularized likelihood score (Theorem 2, Section 4.2).

1.3 Organization

The paper is organized as follows. Section 2 defines the generalized linear DAG model and establishes some identifiability results under this model. In Section 3, we set up the optimization problem for learning causal graphs and develop the DARLS algorithm that combines simulated annealing to search over permutation space and a distributed optimization algorithm to optimize the network structure given an ordering. We then establish theoretical results on the convergence of the distributed optimization algorithm and estimation consistency in Section 4. Section 5 consists of simulation studies that compare our method to existing ones using distributed and combined data, test the robustness of DARLS against violations of its underlying model assumptions, and examine the accuracy loss and computational efficiency of our distributed learning algorithm. We also apply the distributed learning methods to ChIP-sequencing data for modeling protein-DNA binding networks in Section 6. The paper is concluded with a discussion in Section 7. All proofs are relegated to the supplementary material.

2 Generalized linear DAG models

Denote by $x^j \in \mathbb{R}^{d_j}$ a realization of variable X_j, where $d_j = 1$ for a numerical X_j and $d_j = r_j - 1$ for a categorical variable X_j with r_j classes, using the one-hot encoding. Let $\beta_{ij} \in \mathbb{R}^{d_i \times d_j}$ denote the parameters associated with the edge $X_i \to X_j$ and $\beta_{ij} = 0$ if $X_i \notin \mathrm{PA}_j$. Put

$$\beta_j := [\beta_{0j}, \beta_{1j}, \dots, \beta_{pj}] \in \mathbb{R}^{(d+1)\times d_j}, \tag{2}$$

$$x := [1, x^1, \dots, x^p] \in \mathbb{R}^{d+1}, \tag{3}$$

where $\beta_{0j} \in \mathbb{R}^{1\times d_j}$ and $d = \sum_{i=1}^{p} d_i$. Here and elsewhere, $[x, y]$ denotes the vertical concatenation of two vectors or matrices x and y. We define a *generalized linear DAG* (GLDAG) by (1) with conditional densities given by GLMs with canonical links, that is,

$$p(x^j \mid pa_j, \beta_j) = c_j(x^j) \exp\left(\langle \beta_j^\top x, x^j\rangle - b_j(\beta_j^\top x)\right), \quad j \in [p] := \{1, \dots, p\} \tag{4}$$

where b_j and c_j are both functions from $\mathbb{R}^{d_j}$ to $\mathbb{R}$. Note that $\beta_j^\top x = \sum_{i\in \mathrm{PA}_j} \beta_{ij}^\top x^i$ only depends on the parent set PA_j. GLDAG models allow for many common distributions via the choice of the log partition-function $b_j(\cdot)$. Examples include the Bernoulli distribution for $b_j(\theta) = \log(1 + e^\theta)$, constant-variance Gaussian for $b_j(\theta) = \theta^2/2$, Poisson for

$b_j(\theta) = \exp(\theta)$, Gamma for $b_j(\theta) = -\log(-\theta)$ and the multinomial for $b_j(\theta) = \log\left(1 + \sum_{l=1}^{d_j} e^{\theta_l}\right)$. Note that in the multinomial case $b_j(\cdot)$ is a multivariate function, operating on a vector $\theta = (\theta_l) \in \mathbb{R}^{d_j}$, in contrast to the other example for which $b_j(\cdot)$ is a scalar function. The Bernoulli and multinomial choices above give rise to logistic and multi-logit regression models for each node.

We collect all the parameters of model (4) in a matrix $\beta \in \mathbb{R}^{(d+1)\times d}$ which is obtained by horizontal concatenation of $\beta_j, j = 1, \ldots, p$, each as defined in (2). We say that a GLDAG (4) is continuous if all the variables are continuous. Recall that in this case, $d_j = 1$ for all $j \in [p]$ and thus β is a $(p+1) \times p$ matrix. We rewrite the log pdf of (4), in the continuous case, as

$$L(x; \beta) = \sum_{j=1}^{p} \left[\log c_j(x^j) + x^j \beta_j^\top x - b_j(\beta_j^\top x) \right], \tag{5}$$

where $\beta_j^\top x \in \mathbb{R}$ and $\beta_{ij} \neq 0$ if and only if $X_i \to X_j$. Next, we define identifiability of DAG models following [37, 38, 39], and show that continuous GLDAGs are identifiable.

Definition 1 (Identifiability)**.** *Suppose we are given a joint distribution $\mathcal{L}(X) = \mathcal{L}(X_1, \ldots, X_p)$ that has been generated from an unknown GLDAG model* (4) *with a graph $\mathcal{G}_0$. If the distribution $\mathcal{L}(X)$ cannot be generated by any GLDAG model with a different graph $\mathcal{G} \neq \mathcal{G}_0$, then we say $\mathcal{G}_0$ is identifiable from $\mathcal{L}(X)$.*

It is well-known that linear Gaussian DAGs (with heterogeneous variance) and multinomial DAGs in general are not identifiable [35, 36]. In contrast, continuous GLDAG models (5) are identifiable under mild assumptions:

Proposition 1. *Suppose the joint distribution $\mathcal{L}(X)$ is defined by the log-pdf $L(x; \beta)$ with a DAG $\mathcal{G}_0$ according to* (5) *such that $\beta_{ij} \neq 0$ if and only if $i \in PA_j$ in $\mathcal{G}_0$. If $L(x; \beta)$ is second-order differentiable with respect to x and, for all j, the first-order derivative of $b_j(\cdot)$ exists and is not constant, then $\mathcal{G}_0$ is identifiable from $\mathcal{L}(X)$.*

Proposition 1 establishes the identifiability of continuous GLDAG models (5), partially justifying our goal as to learn causal graphs. This result expands the class of identifiable DAG models in the literature. A different class of identifiable DAG models is the additive noise model, $X_j = f_j(\text{PA}_j) + \varepsilon_j$, under assumptions of nonlinear f_j [37, 38, 40] or non-Gaussian error [41].

3 Federated DAG learning

In this section, we construct the objective function using distributed data and propose a simulated annealing search combined with an iterative optimization method to learn causal DAG structures. We start with the definition of topological sorts for DAGs. Given a permutation π on $[p]$, we permute a vector $v = (v_1, \ldots, v_p)$ according to π to obtain a relabeled vector $v_\pi = (v_{\pi(1)}, \ldots, v_{\pi(p)})$. A *topological sort* of a DAG is a permutation of nodes such that if $a \in \text{PA}_b$, then a precedes b in the order defined by π, denoted by $a \prec_\pi b$. By definition (1), every DAG has at least one topological sort.

Let $\{x_h\}_{h=1}^n$ be an i.i.d. sample of size n from model (4). We also let x_h^j represent the observed value of the j-th variable (X_j) in the h-th data point. Consider a subset $\mathcal{I} \subset [n]$. The normalized negative log-likelihood of the subsample $\{x_h\}_{h\in\mathcal{I}}$ is given, up to an additive constant, by

$$\ell_{\mathcal{I}}(\beta) := \frac{1}{|\mathcal{I}|} \sum_{h\in\mathcal{I}} \sum_{j=1}^{p} \left[b_j(\beta_j^\top x_h) - \langle \beta_j^\top x_h, x_h^j \rangle \right]. \tag{6}$$

Note that in this notation, $\ell_{[n]}$ denotes the normalized negative log-likelihood of the entire sample of size n.

3.1 Global objective function and annealing

We consider the case that the overall data is stored on K different servers, where each local client $\mathcal{M}_k$ holds its private data $\{x_h\}_{h\in\mathcal{I}_k}$ and communicates with a central server $\mathcal{C}$. Let $n_k = |\mathcal{I}_k|$ be the sample size in $\mathcal{M}_k$ so that $\sum_{k=1}^K n_k = n$. The normalized negative log-likelihood based on the entire data can be decomposed as $\ell_{[n]}(\beta) = \sum_{k=1}^K \frac{n_k}{n} \ell_{\mathcal{I}_k}(\beta)$. Let $\mathcal{P}$ be the set of all permutations on $[p]$ and $\mathcal{D}(\pi) \subset \mathbb{R}^{(d+1)\times d}$ the set of DAGs whose topological sorts are compatible with a permutation $\pi \in \mathcal{P}$. Note that $\mathcal{D}(\pi)$ is a linear subspace of $\mathbb{R}^{(d+1)\times d}$. We ideally would like to estimate β by minimizing a regularized loss function of the form

$$\min_{\pi\in\mathcal{P}} f(\pi), \quad \text{where}$$

$$f(\pi) := \min_{\beta\in\mathcal{D}(\pi)} \sum_{k=1}^{K} \frac{n_k}{n} \ell_{\mathcal{I}_k}(\beta) + \rho(\beta), \tag{7}$$

and $\rho(\cdot)$ is an appropriate regularizer to promote sparsity in β. We call $f(\pi)$ the global objective function since it is defined using all data across local clients.

Recall that $\beta_{ij} \neq 0$ if and only if $i \in \text{PA}_j$. To learn sparse DAGs, we apply group regularization of the form

$$\rho(\beta) = \lambda \sum_{i,j} \rho_g(\beta_{ij}), \tag{8}$$

where $\rho_g(\cdot)$ is a nonnegative and nondecreasing group regularizer and $\lambda > 0$ is a tuning parameter. Restricted to $\mathcal{D}(\pi)$, the regularizer can be further simplfied to $\rho(\beta) = \lambda \sum_j \sum_{i\prec_\pi j} \rho_g(\beta_{ij})$. In this paper, we consider the Group Lasso (i.e., group ℓ_2) penalty with the choice

$$\rho_g(\beta_{ij}) = \|\beta_{ij}\|_F, \tag{9}$$

where $\|\beta_{ij}\|_F$ is the Frobenius norm of matrix β_{ij}. As a convex penalty and a natural extension of Lasso regularization, Group Lasso has demonstrated remarkable performance in grouped variable selection [42].

To search over ($\pi \in \mathcal{P}, \beta \in \mathcal{D}(\pi)$) with distributed data as in (7), we propose the *distributed annealing on regularized likelihood score* (DARLS) algorithm, which applies annealing strategies to search over the permutation space, coupled with a distributed optimization method. Such manner of joint optimization over the topological sort space and the DAG space has demonstrated great effectiveness in structure learning; see for example, [43, 44, 45, 46] and the references thereof.

The main steps of DARLS are outlined in Algorithm 1. At each annealing iteration, a permutation π^+ is proposed based on current $\widehat{\pi}$ (line 5) and is accepted with probability according to simulated annealing given a decreasing temperature

Algorithm 1 Distributed annealing on regularized likelihood score (DARLS).

```
Input: $\{x_h\}_{h=1}^n$ distributed over $K$ clients, $\pi_0$,
       temperature schedule $\{T^{(i)}\}_{i=0}^N$, $\tau$.
Output: $\widehat{\pi}, \widehat{\beta}$.
1: Select tuning parameter $\lambda$ by BIC selection given $\pi_0$.
2: $\widehat{\pi} \leftarrow \pi_0$, compute $(\widehat{\beta}, f(\widehat{\pi}))$ by Algorithm 2.
3: for $i = 0, \dots, N$ do
4:     $T \leftarrow T^{(i)}$.
5:     Central server $\mathcal{C}$ proposes $\pi^+$ by flipping a
       random interval (length up to $\tau$) in $\widehat{\pi}$.
6:     Compute $(\beta^+, f(\pi^+))$ using Algorithm 2.
7:     $\mathcal{C}$ sets $(\widehat{\pi}, \widehat{\beta}, f(\widehat{\pi})) \leftarrow (\pi^+, \beta^+, f(\pi^+))$ with prob.
       $\min\left\{1, \exp\left(-\frac{f(\pi^+)-f(\widehat{\pi})}{T}\right)\right\}$.
8: end for
9: Refine the causal structure implied by $\widehat{\beta}$.
```

Algorithm 2 Distributed optimization to compute the global permutation score.

```
Input: $\pi$, $\beta_\pi^{(0)}$, number of iteration $T$.
Output: $\widehat{\beta}_\pi, f(\pi)$.
1: Server $\mathcal{C}$ broadcasts $\pi$ to local clients $\{\mathcal{M}_k\}_{k=1}^K$.
2: for $t = 0, 1, \dots, T-1$ do
3:     Each $\mathcal{M}_k$ computes $\nabla\ell_{\mathcal{I}_k}(\beta_\pi^{(t)})$ and sends it to $\mathcal{C}$.
4:     $\mathcal{C}$ computes $\nabla\ell_{[n]}(\beta_\pi^{(t)}) = \frac{1}{n}\sum_k n_k \nabla\ell_{\mathcal{I}_k}(\beta_\pi^{(t)})$ and
       broadcasts it to local clients.
5:     Each $\mathcal{M}_k$ computes $\beta_{k,\pi}^{(t+1)} = \varphi_{k,\pi}(\beta_\pi^{(t)})$ via (10)
       and sends it to $\mathcal{C}$.
6:     $\mathcal{C}$ computes $\beta_\pi^{(t+1)} = \frac{1}{n}\sum_k n_k \beta_{k,\pi}^{(t+1)}$ and broadcasts it
       to local clients.
7: end for
8: Each $\mathcal{M}_k$ reports $F_k^{(T)} := n_k F_k(\beta_\pi^{(T)})$ to $\mathcal{C}$, and $\mathcal{C}$ sets $\widehat{\beta}_\pi \leftarrow$
   $\beta_\pi^{(T)}$ and $f(\pi) \leftarrow \frac{1}{n}\sum_k F_k^{(T)}$.
```

schedule. To compute the score of the optimal DAG structure for a given permutation, we use the *distributed optimization* approach outlined in Algorithm 2, the details of which are discussed in Section 3.2 below. This approach allows multiple rounds of communications between local clients and the central server to update and synthesize information. Note that DARLS can be applied to any objective function as long as the gradient w.r.t. β has a closed-form expression. Other steps (line 1 and line 9) are discussed in Section 3.3.

Remark 1. To estimate Gaussian DAGs from multiple independent data sets, we can use first-order methods, such as stochastic gradient or proximal gradient algorithm [46]. Given π^+, this type of approach computes the global estimate β^+ using sample covariances of local data sets. Hence, distributed learning of Gaussian DAGs only requires averaging local sufficient statistics and does not need the distributed optimization (line 6, Algorithm 1), and thus it is not the focus of this work.

3.2 Local objectives and distributed optimization

For any fixed π, we use distributed computing to evaluate $f(\pi)$, as the samples $\mathcal{I}_k, k \in [K]$ are not shared among the local clients. That is, instead of directly working with the objective function in (7), we rely on local versions of it to guide a distributed algorithm that divides the task of computing $f(\pi)$ among the K local client. In particular, we consider the local objective functions

$$f_k(\pi) := \min_{\beta\in\mathcal{D}(\pi)} F_k(\beta), \quad \text{where}$$
$$F_k(\beta) := \ell_{\mathcal{I}_k}(\beta) + \rho(\beta).$$

The global version (7) can be rewritten as $f(\pi) = \min_{\beta\in\mathcal{D}(\pi)} F(\beta)$ where $F(\beta) := \ell_{[n]}(\beta) + \rho(\beta)$. Typically, each of F and F_k is nonsmooth due to the presence of the regularizer ρ, but the difference $h_k := F_k - F = \ell_{\mathcal{I}_k} - \ell_{[n]}$ is often smooth. The gradient of h_k is used to guide iterations in each local client. That is, given the current (global) estimate β, local client $\mathcal{M}_k$ performs the update

$$\begin{aligned}\varphi_{k,\pi}(\beta) &:= \operatorname*{argmin}_{\xi\in\mathcal{D}(\pi)} \big[\widetilde{F}_k(\xi) := F_k(\xi) - \langle\nabla h_k(\beta), \xi\rangle \\ &= F_k(\xi) - \langle\nabla\ell_{\mathcal{I}_k}(\beta) - \nabla\ell_{[n]}(\beta), \xi\rangle\big]. \qquad (10)\end{aligned}$$

The local regularized loss F_k guided by ∇h_k, denoted by $\widetilde{F}_k$, is a first-order approximation to the global regularized loss F, up to an additive constant. Let $\beta_\pi^{(t)}$ be the global estimate of the algorithm at iteration t. At the next iteration, $t+1$, we obtain local estimates $\beta_{k,\pi}^{(t+1)} = \varphi_{k,\pi}(\beta_\pi^{(t)})$ for $k = 1, \dots, K$. These local estimates are then passed to the central server $\mathcal{C}$ to compute the next global estimate by averaging, i.e., $\beta_\pi^{(t+1)} = \sum_{k=1}^K \frac{n_k}{n}\beta_{k,\pi}^{(t+1)}$. The main steps of this distributed optimization method are outlined in Algorithm 2.

The above approach is essentially a version of the DANE algorithm [19, 21, 22, 23]. Note that to calculate local updates $\beta_{k,\pi}^{(t+1)}$ (line 5, Algorithm 2), only the current global estimate $\beta_\pi^{(t)}$ and the global gradient $\nabla\ell_{[n]}(\beta_\pi^{(t)})$ need to be communicated to each local client. In Section 4.1, we show that for a sufficiently large minimum sample size per client, i.e. $\min_k n_k$, the sequence $\{\beta_\pi^{(t)}\}_{t\geq 0}$ thus produced will converge to a global minimizer $\widehat{\beta}_\pi$ of $F(\cdot)$ over $\mathcal{D}(\pi)$.

Another piece in the distributed optimization is to compute the local update (10) (line 5, Algorithm 2), for which we use the proximal gradient algorithm (Algorithm 3). Given a current global estimate $\beta^{(t)}$, optimizing local objective $\widetilde{F}_k(\xi)$ (10) is equivalent to

$$\min_{\xi\in\mathcal{D}(\pi)} \ell_{\mathcal{I}_k}(\xi) - \langle\nabla h_k(\beta^{(t)}), \xi\rangle + \rho(\xi). \qquad (11)$$

Define $\widetilde{\ell}_{\mathcal{I}_k}(\xi) := \ell_{\mathcal{I}_k}(\xi) - \langle\nabla h_k(\beta^{(t)}), \xi\rangle$, a surrogate for the global likelihood $\ell_{[n]}(\xi)$. To solve (10), we use iterative proximal gradient descent. At each iteration, we minimize a quadratic approximation to $\widetilde{\ell}_{\mathcal{I}_k}$ around the current solution ξ, plus a regularization term,

$$\begin{aligned}\xi^+ := \operatorname*{argmin}_{\xi'\in\mathcal{D}(\pi)} \big[&\widetilde{\ell}_{\mathcal{I}_k}(\xi) + \langle\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi), \xi' - \xi\rangle \\ &+ \frac{1}{2s}\|\xi' - \xi\|_F^2 + \rho(\xi')\big], \qquad (12)\end{aligned}$$

where $s > 0$ plays the role of a step size and ξ^+ is our next estimate of the solution. Equivalently, the update (12) can be re-written as

$$\xi^+ = \text{prox}_{s\rho}\left(\xi - s\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi)\right), \tag{13}$$

where

$$\text{prox}_{s\rho}(\beta) := \underset{u}{\text{argmin}}\left(s\rho(u) + \frac{1}{2}\|\beta - u\|_F^2\right)$$

is a proximal operator applied to the scaled function $s\rho(\cdot)$. Equation (13) is known as a *proximal gradient update*, and for our choice of the regularizer given by (8) and (9), has the following closed-form expression:

$$\left(\text{prox}_{s\rho}(\beta)\right)_{ij} = \left(1 - \frac{s}{\|\beta_{ij}\|_F}\right)_+ \beta_{ij}$$

This is often referred to as the *block soft-thresholding* operator. To determine the value of the step size s, we use backward line search, shrinking an initial value s_0 until a proper step size is found (line 7, Algorithm 3). Convergence of Algorithm 3 is guaranteed given the convexity of (11) over $\mathcal{D}(\pi)$ [47, 48]. However, to avoid possible slow convergence, we set a maximum iteration for early stopping (line 2). We refer readers to [49] for more details on the proximal algorithms.

Computational complexity. Let us now give an estimate of the overall computational complexity of DARLS. For simplicity, consider numerical X_j so that $d_j = 1$ for all j. Then, β_j is a vector in $\mathbb{R}^{p+1}$ and $\beta = (\beta_j)_{j\in[p]}$ can be thought of as a vector of dimension $p(p+1) = O(p^2)$. From (5), for a single sample $x \in \mathbb{R}^{p+1}$,

$$\partial L(x,\beta)/\partial\beta_j = \left(x^j - b_j'(\beta_j^T x)\right)x$$

and hence needs $O(p)$ operations to compute. Calculating $\partial L(x,\beta)/\partial\beta$ then needs $O(p^2)$ operations for a single sample, and hence $\nabla\ell_{\mathcal{I}}(\beta)$ has computational complexity $O(|\mathcal{I}|p^2)$. For subsequent calculations, note that $\nabla\ell_{\mathcal{I}}(\beta)$ is itself an $O(p^2)$-dimensional vector. For simplicity, assume that all the K local clients have the same number of local samples, namely $m := n/K$. Then, the complexity of step 3 of Algorithm 2 is, $O(mp^2)$, where we are treating parallel computations as one. Let M be the maximum number of iterations in Algorithm 3. Since computing $\text{prox}_{s\rho}(\beta)$ also needs $O(p^2)$ operations, the complexity of step 5 of Algorithm 2 (that is, the entire Algorithm 3) is $O(Mmp^2)$. The complexity of steps 4 and 6 of Algorithm 2 are $O(Kp^2)$. The overall complexity of Algorithm 2 is thus, $O((Mmp^2 + Kp^2)T)$. This gives the overall complexity of $O\big(p^2\big(M(n/K) + K\big)NT\big)$ for DARLS.

3.3 Tuning parameters and structure estimation

Given an initial permutation π_0, we use BIC grid search to select tuning parameter λ that is used in the group Lasso penalty (8) (line 1, Algorithm 1). To construct the grid, we select 20 equally spaced points $\lambda^{(i)}$, on the log scale, from the interval $[0.01, 0.1]$, where $\lambda = 0.1$ is sufficiently large to produce an empty graph in our test. We select the tuning parameter $\lambda^{(i)}$ that minimizes the BIC score, $\text{BIC}(i) = 2\ell_{[n]}(\widehat{\beta}^{(i)}) + (\log n)\mathcal{N}(\widehat{\beta}^{(i)})$, where $\widehat{\beta}^{(i)} \in \mathcal{D}(\pi_0)$ is the minimizer of $F(\beta)$ with penalty parameter $\lambda^{(i)}$ over $\mathcal{D}(\pi_0)$, computed by Algorithm 2, and $\mathcal{N}(\widehat{\beta}^{(i)})$ is the number

Algorithm 3 Use the proximal gradient algorithm to compute local permutation scores.

```
    Input: $\{x_h\}_{h\in\mathcal{I}_k}$, $\pi$, $\beta^{(t-1)} \in \mathcal{D}(\pi)$, $\nabla h_k(\beta^{(t-1)})$, $s_0 > 0$,
           $\kappa \in (0,1)$, max-iter, tol.
    Output: $\beta^{(t)}_{k,\pi}$.
 1: iter $\leftarrow 0$, err $\leftarrow \infty$, $\xi \leftarrow \beta^{(t-1)}$.
 2: while iter $<$ max-iter and err $>$ tol do
 3:     $\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi) \leftarrow \nabla\ell_{\mathcal{I}_k}(\xi) - \nabla h_k(\xi)$
 4:     $s \leftarrow s_0/\|\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi)\|_F$.
 5:     repeat
 6:         $\xi^+ \leftarrow \text{prox}_{s\rho}(\xi - s\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi))$.
 7:         break if $\widetilde{\ell}_{\mathcal{I}_k}(\xi^+) \le \widetilde{\ell}_{\mathcal{I}_k}(\xi) + \langle\nabla\widetilde{\ell}_{\mathcal{I}_k}(\xi), \xi^+ - \xi\rangle$
                      $+ \frac{1}{2s}\|\xi^+ - \xi\|_F^2$.
 8:         $s \leftarrow \kappa s$.
 9:     err $\leftarrow d(\xi^+, \xi)$ where $d(x,y) := \frac{\|x-y\|_F}{\max\{1,\|y\|_F\}}$.
10:     $\xi \leftarrow \xi^+$ and iter $\leftarrow$ iter $+1$.
11: end while
12: $\beta^{(t)}_{k,\pi} \leftarrow \xi$.
```

of free parameters in $\widehat{\beta}^{(i)}$. Note that our tuning parameter selection is done before estimating π and β in Algorithm 1, which is different from the common practice that selects λ after obtaining a solution path. Our strategy greatly reduces the computational cost and works well in practice as demonstrated in our prior work [46].

An estimated GLDAG parameter $\widehat{\beta}$ is provided at the end of the DARLS annealing search, from which we can estimate a causal structure (line 9, Algorithm 1). Let W be a $p \times p$ weighted adjacency matrix of a DAG with weights $W_{ij} := \|\widehat{\beta}_{ij}\|_F$. The use of a group Lasso regularizer helps to produce a sparse estimated DAG, but false positive edges would present in general. Hence, we further refine estimated structures by setting W_{ij} to zero if $|W_{ij}| < \alpha \max_{ij}|W_{ij}|$. One can adjust the value of α to achieve a desired sparsity level, especially when having prior knowledge. In our simulation tests, we fix $\alpha = 0.1$ to remove edges whose weights are relatively small compared to others.

3.4 Comparison to other federated learning methods

In this section, we elucidate the distinctions between our approach and two recent federated methods, FedDAG [33] and NOTEARS-ADMM [32]. First and foremost, our work stands out as the only method with a crucial theoretical guarantee, ensuring convergence of the federated estimate to the oracle estimate based on all local data (see Section 4). Second, the imposition of the acyclicity constraint in our work diverges from the approaches by the other methods. In our algorithm, we rely on order-based search by the central server, while the other two methods use a continuous algebraic constraint [50]. However, recent work [51] shows that the algebraic constraint cannot be satisfied exactly and therefore, post-processing such as thresholding is needed to obtain an estimated DAG. Third, the associated distributed optimization is also very different. In our algorithm, the local client and the central sever communicate the gradient information. FedDAG [32] adopts a strategy where the central server averages over proxy adjacency matrices from local clients and broadcasts the average back to local clients in each round of communication. In NOTEARS-ADMM [33], the model parameters (e.g. weighted adjacency matrix) of local clients and the central server are exchanged via iterative updating rules in the alternative direction method of multipliers (ADMM). Lastly, our work is focused on generalized linear models for various classes of variables, while the other two papers are focused on continuous variables in linear and nonlinear Gaussian cases.

4 Theoretical guarantees

In this section, we study the convergence of the distributed optimization (Algorithm 2) to the oracle solution and establish the consistency of the global minimizer of (7). As our method is primarily motivated by applications involving a large amount of distributed data, we develop theoretical results under the setting that n is large and the number of variables p stays fixed. Our focus in the analysis will be on the non-Gaussian case. In the Gaussian setting, the sample covariance matrix is a sufficient statistic and the local client can communicate their versions to the central server in one round, which can then form the full matrix and compute the global DAG estimate. In other words, there is no need for an elaborate distributed algorithm such as Algorithm 1 in the Gaussian case.

4.1 Oracle guarantees

Recall the local iteration functions $\varphi_{k,\pi}$ defined in (10). The overall iteration function for the distributed algorithm can be written as $\Phi_\pi(\cdot) := \sum_k \frac{n_k}{n}\varphi_{k,\pi}(\cdot)$ (line 6, Algorithm 2). Let $\Sigma := \mathbb{E}[x_h x_h^\top]$ be the population second-moment matrix of the model and $\lambda_{\min}(\Sigma)$ its minimal eigenvalue. For a matrix β, let $\mathbb{B}_F(\beta; r)$ denote the Frobenius ball of radius r centered at β. We consider the case of numerical variables, i.e. $d_j = 1$ for all j, which includes continuous and binary discrete random variables. The following theorem provides convergence guarantees on the distributed optimization algorithm represented by Φ_π for any fixed π. Let $\widehat{\beta}_\pi$ be any global minimizer of the global objective function, i.e.,

$$\widehat{\beta}_\pi \in \operatorname*{argmin}_{\xi\in\mathcal{D}(\pi)} \ell_{[n]}(\xi) + \rho(\xi), \tag{14}$$

where $\rho(\xi)$ is a convex regularizer. In the distributed data setting, $\widehat{\beta}_\pi$ is an oracle solution with access to all data across multiple local clients. Let $\Omega := \bigcup_\pi \mathcal{D}(\pi) \subset \mathbb{R}^{(p+1)\times p}$ be the parameter space of GLDAGs. We recall that for $\theta \in \Omega$, θ_j denotes the jth column and that $\{x_h\}$ is an i.i.d. sample from a GLDAG model (4).

Theorem 1 (Convergence to the oracle)**.** *Assume that the coordinates of x_h are T-bounded, that is, $|x_h^j| \le T$ for all $h \in [n]$ and $j \in [p]$. Let $\theta \in \Omega$ be any GLDAG parameter and $r > 0$, and set $R_1^* = \max_j \|\theta_j\|_1$ and $r_p := 2r\sqrt{p}$. Let $\underline{b}_p = \inf_{|t|\le T(r_p+R_1^*)} b''(t)$, and assume that $b''(\cdot)$ is $\overline{b}_p$-Lipschitz on $[-Tr_p, Tr_p]$. Define*

$$\zeta_n := \left(T^3 \frac{\psi\big(\overline{b}_p(r + \frac{R_1^*}{\sqrt{p}})\big) + b''(0)}{\underline{b}_p \lambda_{\min}(\Sigma)} \right) \frac{p^{3/2}\log(np)}{\sqrt{m}},$$

where $\psi(x) := \max\{x, \sqrt{x}\}$ and $m := \min_k |\mathcal{I}_k|$. Assume further that $np \ge \max\{K+1, 3\}$. There exist constants $c_1, C_1, C > 0$ such that if $C_1 T^2\sqrt{p^2\log(np)/m} \le \lambda_{\min}(\Sigma)$, then with probability at least $1 - 3(np)^{-c_1} - \mathbb{P}(\|\widehat{\beta}_\pi - \theta\|_F > r)$,

$$\|\Phi_\pi(\beta) - \widehat{\beta}_\pi\|_F \le C\zeta_n \|\beta - \widehat{\beta}_\pi\|_F,$$
$$\text{for all } \beta \in \mathbb{B}_F(\widehat{\beta}_\pi, r).$$

Theorem 1 applies to any $\theta \in \Omega$. It is natural to take θ to be β_π^*, the minimizer of the population loss defined as

$$\beta_\pi^* := \operatorname*{argmin}_{\xi\in\mathcal{D}(\pi)} \mathbb{E}[\ell_{[n]}(\xi)]. \tag{15}$$

Since $\widehat{\beta}_\pi$ is a consistent estimate of β_π^* for any π (Theorem 2, Section 4.2), then $\mathbb{P}(\|\widehat{\beta}_\pi - \beta_\pi^*\|_F > r)$ goes to zero as n grows. Thus, with high probability, the iteration operator $\Phi_\pi(\cdot)$ will be a contraction: the sequence $\{\beta_\pi^{(t)}\}_{t\ge 0}$ produced by the distributed algorithm converges geometrically to the oracle estimator $\widehat{\beta}_\pi$ if $C\zeta_n < 1$. For fixed p, and for sufficiently large r such that $\beta_\pi^{(0)} \in \mathbb{B}_F(\widehat{\beta}_\pi, r)$, one can always satisfy the condition of $C\zeta_n < 1$ by taking m (the minimum sample size per client) large enough. Hence, Theorem 1 provides a quantitative lower bound on m for the geometric convergence to kick in. Note that the T-boundedness assumption is trivially satisfied for binary and ordinal data, which are the primary focus of this work.

Theorem 1 is proved by establishing the uniform concentration of the Hessian of the GLDAG model (4) around its expectation over certain balls in the parameter space, and then invoking a general convergence result for the DANE algorithm which we derive in the supplementary material (cf. Theorem S2). Establishing such uniform concentration in the GLDAG model is challenging due to the highly dependent and nonlinear relation among $\{x_h^j\}_{j=1}^p$. A technical tool in establishing the concentration of the Hessian is the Ledoux–Talagrand contraction theorem. In order to extend the argument to the multi-logit and generally vector-valued DAG models with $d_j > 1$, one needs a multivariate extension of the contraction theorem which is not available in literature at the moment. This extension is, in principle, possible and we leave it for the future work.

4.2 Consistency

In this section we establish consistency results under the class of models in (4), without the restriction to numerical variables. Let us write $\psi(x;\beta) := -\log p(x \mid \beta)$ for the negative log-likelihood of a single sample x from model (4). We view x, x_h and β as vectors by concatenating the columns when dealing with $\psi(\cdot;\cdot)$, so that $\beta \in \mathbb{R}^D$ for $D = d(d+1)$.

Recall that $F(\beta) = \ell_{[n]}(\beta) + \lambda_n \sum_{i,j} \|\beta_{ij}\|_F$ is the global regularized negative log-likelihood and Ω is the GLDAG parameter space. The optimization problem (7) is equivalent to $\min_{\beta\in\Omega} F(\beta)$. Let us denote by $\widehat{\beta} \in \Omega$ a global minimizer of $F(\beta)$ and $\beta^* \in \Omega$ the true parameter with the true DAG $\mathcal{G}^*$. For any β, let us consider the (cross) Fisher information matrix

$$I(\beta;\beta^*) := \mathbb{E}_{\beta^*}\nabla^2\psi(x;\beta).$$

We note that $\psi(x;\cdot)$ is a convex function for exponential families, and hence $I(\beta;\beta^*)$ is always positive semi-definite. To establish the consistency of $\widehat{\beta}$ as well as that of $\widehat{\beta}_\pi$, used in Theorem 1, for any fixed π, we make the following assumptions:

(A1) The true DAG $\mathcal{G}^*$ is identifiable.

(A2) For every π, there exists a neighborhood of β^*_π, denoted by $\text{nb}(\beta^*_\pi)$ and functions M_{jkl} such that $\left|\frac{\partial^3}{\partial\beta_j\partial\beta_k\partial\beta_l}\psi(x;\beta)\right| \leq M_{jkl}(x)$ for all $\beta \in \text{nb}(\beta^*_\pi)$, almost surely, and $\mathbb{E}_{\beta^*}[M_{jkl}(x)] < \infty$ for all j, k and l.

(A3) For every π, we have

$$\inf_{u \in D(\pi),\, \|u\|=1} \langle u, I(\beta^*_\pi; \beta^*)u\rangle > 0.$$

In (A2), it is impliclty assumed that $\psi(x;\cdot)$ is finite in $\text{nb}(\beta^*_\pi)$ almost surely.

Before stating our theoretical results, we define $\Pi^* := \{\pi : \beta^*_\pi = \beta^*\}$ which is exactly the set of permutations consistent with β^* or the topological sorts of $\mathcal{G}^*$, and in particular, it is nonempty. To see this, we first note that for any π that is consistent with β^*, we have $\beta^* \in \mathcal{D}(\pi)$. A KL divergence argument then shows that β^* is the unique solution of the optimization problem defining β^*_π. That is, any π consistent with β^* belongs to Π^*. Conversely, if $\pi \in \Pi^*$, then $\beta^* = \beta^*_\pi \in \mathcal{D}(\pi)$, and hence π is consistent with β^*. With this observation, we establish the desired consistency results in the following theorem.

Theorem 2. *Assume (A1)–(A3) and $\sqrt{n}\lambda_n = \mathcal{O}_p(1)$. Then,*

(a) For every π, $F(\cdot)$ has a unqiue minimizer $\widehat{\beta}_\pi$ over $\mathcal{D}(\pi)$ and

$$\sup_{\pi \in \mathcal{P}} \|\widehat{\beta}_\pi - \beta^*_\pi\|_F = \mathcal{O}_p(n^{-1/2}).$$

(b) $F(\cdot)$ has a unique minimizer $\widehat{\beta}$ over Ω (the space of DAGs) and

$$\|\widehat{\beta} - \beta^*\|_F = \mathcal{O}_p(n^{-1/2}).$$

(c) With probability converging to one as $n \to \infty$, $\widehat{\beta} = \widehat{\beta}_{\widehat{\pi}}$ for some (sequence of) $\widehat{\pi} \in \Pi^*$.

Theorem 2 confirms that the Group Lasso regularized estimator $\widehat{\beta}$, defined as a global minimzier of F, is $\sqrt{n}$-consistent, and it will identify a correct topological sort $\widehat{\pi} \in \Pi^*$ in the large-sample limit. Moreover, the theorem also establishes the uniform consistency of restricted miminizers $\widehat{\beta}_\pi$ for all π, the oracle estimators in Theorem 1. Assumption (A1) holds under mild conditions according to Proposition 1, Assumption (A2) is a standard regularity condition, and Assumption (A3) is related to the non-singularity of the second moment matrix $\Sigma = \mathbb{E}_{\beta^*}(xx^\top)$. For example, consider the case $d_j = 1$ for all j and assume that the elements of x are T-bounded and let $R_j = \max_\pi \|[\beta^*_\pi]_j\|_1$, viewing β^*_π as a matrix with jth column $[\beta^*_\pi]_j$. Then if $\inf_{|t| \leq TR_j} b''_j(t) > 0$ for all j, non-singularity of Σ is sufficient for (A3) to hold.

5 Results on Simulated Data

Denote by s_0 the number of edges in a graph on p nodes. We downloaded the following `networks` (p, s_0) from the Bayesian networks repository [52] to simulate data: `Asia` (8, 8), `Sachs` (11, 17), `Child` (20, 25), `Insurance` (27, 52), `Alarm` (37, 46), `Hailfinder` (56, 66) and `Hepar2` (70, 123). We generated data under the GLDAG model (4) and other common DAG models, in Section 5.3 and Section 5.4 respectively, where the latter is to examine the robustness of our method against violation of its model assumptions.

5.1 Methods

We compared the DARLS algorithm to the following DAG structure learning methods: the standard greedy hill climbing (HC) algorithm [53], the Peter-Clark (PC) algorithm [54], the max-min hill-climbing (MMHC) algorithm [55], the fast greedy equivalence search (FGES) [56, 57, 58], the NOTEARS algorithm [50], and the DAG-GNN method [59]. Among these methods, PC is a constraint-based method and MMHC is a hybrid method. The other three methods are score-based, where HC searches over DAGs, FGES searches over the equivalence classes, NOTEARS uses continuous optimization to estimate DAG structures, and DAG-GNN applies graph neural networks architecture to learn DAGs.

Following the practice for DAG learning on distributed data as in [26, 27, 28], we combined local estimates generated by a competing method to obtain a global graph estimate. Denote the dataset on a local client by $D_k = \{x_h\}_{h \in \mathcal{I}_k}$, $k \in [K]$. We applied a competing method on each local dataset D_k to obtain a completed partially directed acyclic graph (CPDAG) A_k, and then constructed a global graph using $\{A_k\}_{k=1}^K$. We used CPDAGs here because all the competing methods were developed under non-idenfitiable DAG models. Among the five competing methods, only PC and FGES output a CPDAG, and thus we converted estimated DAGs from the other methods to CPDAGs to obtain A_k. Given $\{A_k\}_1^K$, we counted occurrences of the three possible orientations between each pair (i, j): $i \to j$, $i \leftarrow j$, or $i - j$ (undirected). We then ranked orientations across all node pairs in the descending order of their counts, and sequentially added these edge orientations to an empty graph, as long as they would not introduce a directed cycle (a cycle consisting of all directed edges). By the end of this process, we had a partially directed graph. Lastly, we applied Meek's rules [56, 60] to maximally orient undirected edges, and hence constructed a global CPDAG estimate.

Remark 2. The HC global estimates had too many edges using this approach, because its local CPDAGs lacked consensus, resulting in a large number of candidate edges and much higher FP edges. To solve this problem, we did not add any edge between a node pair (i, j) if the majority of the local graphs $\{A_k\}$ had no edges between them. In this way, global graphs estimated by HC became reasonably sparse compared to global estimates by the other methods.

Moreover, we provide numerical results comparing DARLS and NOTEARS-ADMM [32] on binary data in supplementary material Section S2, because NOTEARS-ADMM was not designed for categorical data with more than two levels. The numerical results demonstrate that DARLS outperforms NOTEARS-ADMM, consistently achieving much lower SHD across various data-generating scenarios.

We implemented the DARLS algorithm in MATLAB and used the following `packages` to run competing methods: `bnlearn` [61] for the MMHC and HC algorithms, `pcalg` [62] for the PC algorithms and `rcausal` [57] for the FGES. The NOTEARS and DAG-GNN methods were run with their online Python code [63, 64]. Competing methods were applied to each local dataset using a 2016 MacBook Pro (2.9 GHz Intel Core i5, 16 GB memory). Since DARLS is designed for distributed computing, it was run on a computer cluster.

In this study, the DARLS algorithm (Algorithm 1) was initialized with a random permutation. According to the landscape of the objective function, we set the initial annealing temperature to $5 \cdot 10^{-2}$ and gradually decreased it to $5 \cdot 10^{-5}$ in a geometric fashion over a total of 10^3 iterations. Note that since the log-likelihood has been normalized by the sample size, as in (6), the range of the objective function is quite small. For the PC algorithm, a significance level of 0.01 was used to generate graphs with desired sparsity. FGES was applied with a significance level of 0.1, which was the default value. For MMHC and HC methods, the maximum number of parents for a node was set to three. For NOTEARS, we used the default ℓ_2 loss and the default threshold value. For DAG-GNN, we set its threshold as 0.15 and used other default parameter values, where the default value of the threshold is 0.3 and it is used to refine the final DAG structure, similar to the parameter α in our post-processing step (Session 3.3). In our numerical pilot experiments using all the default values of DAG-GNN, it always output empty graphs. Thus, we manually tuned its input parameters one by one, and this method only generated non-empty structures when reducing the threshold value.

5.2 Accuracy metrics

Given estimates generated by the above methods, we use a few metrics to evaluate their structural accuracy. To standardize the performance metrics, we transform an estimated DAG into its CPDAG when the true DAG is not identifiable, before calculating the following metrics.

Let P, TP, FP, M, R be the number of estimated edges, true positive edges, false positive edges, missing edges and reversed edges, respectively. More specifically, P is the number of edges in the estimated graph, FP is the number of edges in the estimated graph skeleton but not in the true skeleton, and M counts the number of edges in the true skeleton but not in the skeleton of the estimated graph. TP reports the number of consistent edges between the estimated DAG/CPDAG and the true DAG/CPDAG, where a consistent edge must have the same orientation between the two nodes. There are two possible orientations for an edge in a DAG and three in a CPDAG. Lastly, the number of reversed edges R = P − TP − FP. We then define structural Hamming distance, SHD = R + FP + M, as a combined metric. A method has higher structure learning accuracy if it achieves a lower SHD.

5.3 GLDAG data

Logistic GLDAGs model (4) with $b_j(\beta_j^\top x) = \log(1 + \exp(\beta_j^\top x))$ for all $j \in [p]$, was used to generate binary data $X_j \in \{0, 1\}$, where the coefficient parameters $\{\beta_{ij}\}$ were uniformly sampled from $[-1.5, -0.8] \cup [0.8, 1.5]$. We simulated 20 datasets for each network under two settings, $n = 100p, K = 10$ and $n = 10,000, K = 20$, where a total of n observations were randomly assigned to K local clients. Since GLDAGs are identifiable, we compare an estimated DAG by DARLS to the true DAG when calculating the SHD. For all other methods, we compare estimated and true CPDAGs since they do not assume an identifiable DAG.

TABLE 1: DARLS against others on distributed logistic data.

Network (p)	DARLS	MMHC	FGES	HC	PC	GNN
`Asia` (8)	**4.6**	6.7	9.0	7.2	6.7	8.4
`Sachs` (11)	**10.0**	10.0	19.1	13.3	13.7	20.6
`Child` (20)	**10.4**	22.1	28.8	18.8	23.1	29.4
`Insurance` (27)	**21.6**	38.2	42.9	41.0	39.6	49.9
`Alarm` (37)	**18.8**	38.5	37.1	35.0	33.8	-
`Hailfinder` (56)	**27.5**	69.2	56.0	51.2	-	-
`Hepar2` (70)	**51.7**	89.0	66.2	62.8	-	-
`Asia` (8)	**3.5**	5.2	8.2	4.7	5.5	6.8
`Sachs` (11)	**8.3**	11.1	19.4	10.2	11.8	14.7
`Child` (20)	**8.2**	19.2	23.0	11.7	20.8	21.5
`Insurance` (27)	**20.4**	34.0	39.9	31.6	37.5	47.4
`Alarm` (37)	**19.1**	43.0	41.4	31.5	40.5	-
`Hailfinder` (56)	**27.7**	91.7	72.5	50.4	-	-
`Hepar2` (70)	**48.8**	118.5	86.5	73.5	-	-

The upper panel pertains to scenarios with $n = 100p$ and $K = 10$, while the lower panel illustrates cases using $n = 10,000$ and $K = 20$. The minimum average SHD of each network is highlighted in bold.

Table 1 reports the average performance metrics SHD across 20 datasets for each of the seven graphs, using six methods. A detailed breakdown, including averages of TP, FP, R, and M, along with the standard deviations of SHD, is provided in Table S1 in the supplement. Since NOTEARS estimates were too sparse to be competitive, using the default settings, we decreased the penalty tuning parameter to 10^{-4} from the suggested value of 10^{-1}. We only report its results for two small graphs, `Asia` and `Sachs` in the supplementary material. Its SHD falls within the median performance spectrum compared to the other methods. DAG-GNN (GNN in Table 1) was time-demanding in computing graphs, taking more than one hour for each data set when $n = 10,000$, while other competing methods took at most 5 minutes. Thus, we only provide its results for the first four graphs that are relatively small. PC also had difficulty generating estimates within a reasonable time limit (30 minutes per estimation) for the last two networks, `Hailfinder` and `Hepar2`, and hence it was removed from the comparisons on these two graphs.

Table 1 shows that in both cases $n = 100p$ and $n = 10,000$, DARLS consistently achieved the lowest SHD among all methods for every network, demonstrating higher accuracy in estimating graphical structures. The relative efficacy of DARLS remains robust, showing no signs of diminishing as the graph size increases. Across the four largest graphs, `Insurance`, `Alarm`, `Hailfinder` and `Hepar2`, DARLS consistently achieved about 40% decrease in SHD relative to the second-best method for $n = 10,000$. Additional numerical studies on the performance of DARLS with growing graph sizes from 76 to 223 are provided in the supplementary material Section S4, showing comparable or even more substantial improvement. DARLS identified more TP edges than the other methods in almost every case. The refinement step via thresholding $\widehat{\beta}$ also helped to reduce SHD by cutting down FP edges. A key difference from the competing methods is the federated learning feature of DARLS, which coordinates structure learning across all local clients. The numerical comparison shows this is much more accurate than a simple consensus by vote or average over individually learned graphs from local clients. Due to lack of coordination, the local estimates may represent a set of locally optimal structures, having substantially different graphical structures. A consensus constructed from a set of such graphs may not be close to the true structure. This issue is in general more severe when the graph is large.

To examine the loss of accuracy in estimating network structures on distributed data, we computed the oracle solutions of each method assuming full access to all the K local datasets (combined data). For brevity in reporting the results, we chose the best performance on each network among the five competing methods, called the best competing method, to compare with DARLS. Figure 1 shows the performance, in terms of SHDs, of DARLS and the best competing method on distributed and combined data.

First, we observe that, applied to either distributed or combined data, DARLS achieves similar SHD values. The SHD difference between the distributed and combined data of DARLS is much smaller than that of the best competing method, demonstrating that DARLS is more effective in using distributed data. Second, consistent with the results in Table 1, DARLS always outperformed significantly the best competing method applied on distributed data (best-distributed). Furthermore, DARLS on distributed data shows a substantial overlap in the distribution of SHD with the best competing method on combined data (best-oracle), which is either HC or FGES for $n = 100p$ and HC, MMHC or FGES for $n = 10,000$. Such overlaps indicate the highly competitive performance of our distributed learning algorithm. The variability in SHD of DARLS is in general smaller than that of the best competing method, showing a higher consistency across different datasets.

To quantify the accuracy of distributed optimization using Algorithm 2, we computed permutation scores $f(\pi)$ (7) under various values of $K \in \{1, 2, 5, 10\}$ for a fixed π. For each value of K, we fixed the tuning parameter λ, permutation π and all internal computation parameters to ensure the only K varied in the calculation of $f(\pi)$. Let $f^{(K)}$ be the value of $f(\pi)$ computed by K local clients, and $\Delta f^{(K)} := f^{(K)} - f^{(1)}$ be the relative increase in the loss $f^{(K)}$. Figure 2a shows $\{\Delta f^{(K)} : K = 2, 5, 10\}$ across all the networks. The values of $\Delta f^{(K)}$ is in the order of 10^{-13}, verifying $f(\pi)$ is essentially identical using either the overall ($K = 1$) or distributed data ($K \geq 2$). Since the number of iterations is fixed, $\Delta f^{(K)}$ increases with the network size.

We also examined computation time of finding $f^{(K)}$ for $K \in [20]$ using the 20 `Insurance` datasets. In each test, the same data were split and distributed to different number K of local clients to solve the optimization problem (7), with all other parameters fixed. Figure 2b shows the computation time versus K. As expected when distributing a complicated task, computing $f(\pi)$ requires less time if using more clients. However, the reduction in computation time reaches approximately a stable level after $K = 10$, indicating a trade-off between the gains from parallel computation and the communication overhead. Also, the smallest local sample size m decreases when K is large, and Theorem 1 shows that would slow down the convergence of Algorithm 2.

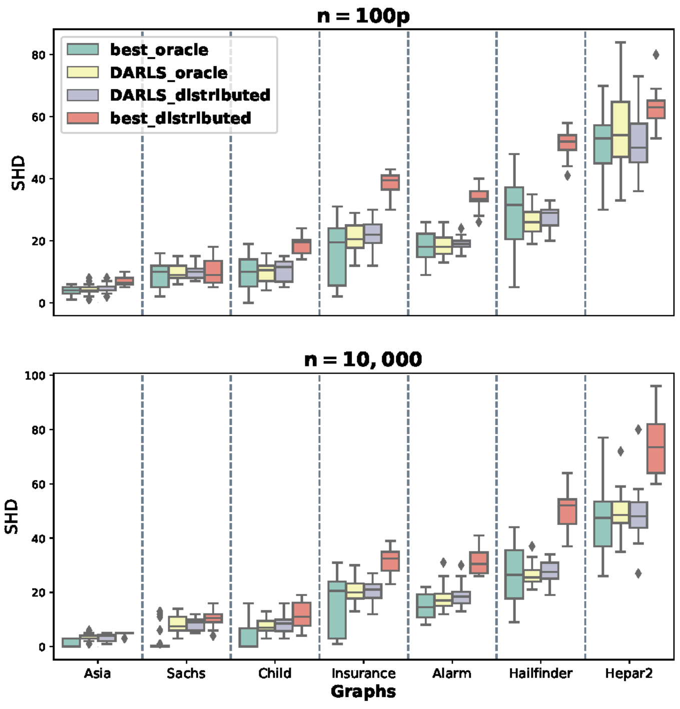


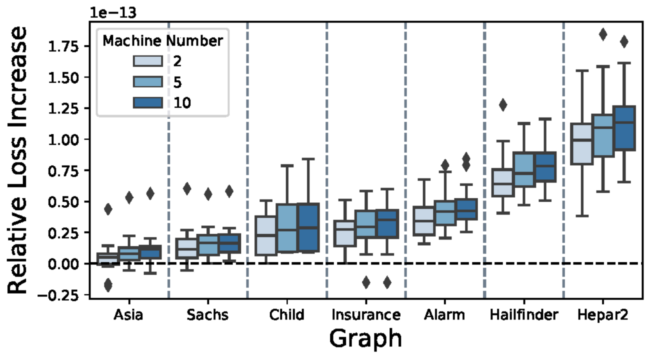
Fig. 1: SHD comparison using combined (oracle) or distributed data. For each graph, the box plots from left to right report the results for the best competing method on combined data (best-oracle), DARLS on combined data (DARLS-oracle), DARLS on distributed data and the best competing method on distributed data (best-distributed).

Federated learning primarily aims to enable collaborative learning across multiple local clients, rather than to speed up processes by distributing data. However, Figure 2b suggests that data distribution could, in fact, reduce computing time for DARLS. Specifically, our findings indicate that for DARLS, the runtime to obtain the oracle estimate with combined data (i.e., $K = 1$) is about four times longer than when $K = 20$.

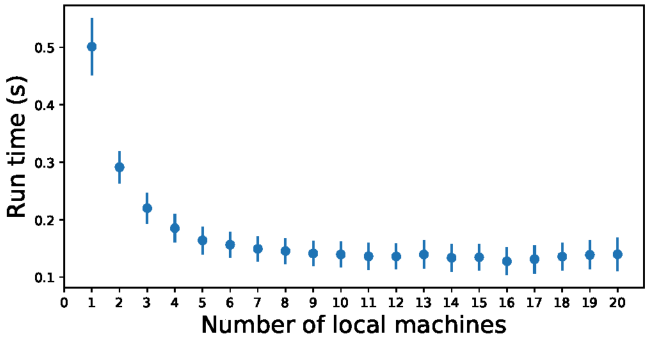
5.4 Other data generation models

To test the robustness of DARLS against violations of its model assumptions, we also generated data from different DAG models. Particularly, we used threshold-Gaussian and multinomial DAGs to generate discrete data, and then compared DARLS with the other methods.

For the threshold-Gaussian DAG model, we first generated continuous variables $Y_1, \ldots, Y_p$ using Gaussian structural equation models $Y_j = B_j^\top \mathrm{PA}_j + \epsilon_j$ where ϵ_j is a Gaussian noise from $\mathcal{N}(0, 1)$ and each coefficient parameter in $B_j \in \mathbb{R}^{s_j}$ was sampled uniformly from $[-1.2, -0.8] \bigcup [0.8, 1.2]$, where $s_j = |\mathrm{PA}_j|$. Then, we thresheld these continuous values to generate binary data $X_j = \mathcal{I}(Y_j > c_j)$, where c_j is the sample mean of Y_j. We used two-component mixture Gaussian to simulate continuous data for root variables in Y_j, each drawn from $\mathcal{N}(1, 1)$ and $\mathcal{N}(-1, 1)$ with an equal probability. Under this design, most of the Y_j's were bi-modal in distribution, which greatly increased the robustness in thresholding. Such conversion between continuous and categorical variables has been used for modeling BNs in prior work [65, 66, 67], but



(a) Relative loss distribution.



(b) Computation time distribution.

Fig. 2: Accuracy and computation time comparison for $f(\pi)$. computation time comparison, with mean and standard deviations plotted, is performed over 20 datasets generated from `Insurance`.

the threshold-Gaussian DAG model is not the underlying model for any of six methods in our comparison.

We also simulated multinomial data from contingency tables provided in the BN repository [52]. We made a few modifications to the contingency tables to ensure that (1) the number of states per variable was at most three, and (2) the marginal probability of any state was at least 0.1 for every variable by merging states. Due to the high structure complexity of `Hailfinder` and `Hepar2`, the original contingency tables of a few nodes were used without modifications, resulting in marginal probability less than 0.1 for some states. We note that the multinomial DAG model is the underlying model for the majority of the competing methods, including HC, PC, FGES and MMHC. Therefore, comparisons on these data would also test if the GLDAG model can approximate the multinomial model reasonably well.

Table 2 reports the structural estimation accuracy of each method on data generated by the threshold-Gaussian and the mulitnomial DAG models, with $n = 10,000$ and $K = 20$. More detailed metrics, including TP, FP, R, M and the SD of SHD are provided in Table S2 of the supplement. When the underlying model is a threshold-Gaussian DAG, DARLS achieved the lowest SHD for 4 networks, i.e. `Asia`, `Sachs`, `Child` and `Insurance`. For data simulated from multinomial models, DARLS still could estimate the network with the lowest SHD for `Sachs` and `Child`. For most of the other cases, DARLS was only worse than HC, but better than or at least comparable to the other methods.

TABLE 2: DARLS against others when its model assumption is violated.

`Network` (p)	DARLS	MMHC	FGES	HC	PC
`Asia` (8)	**2.8**	9.1	11.1	4.5	12.4
`Sachs` (11)	**11.5**	16.6	17.8	12.9	17.9
`Child` (20)	**18.6**	24.5	31.4	20.2	49.0
`Insurance` (27)	**53.8**	57.2	65.1	56.1	82.8
`Alarm` (37)	39.8	48.9	59.5	**36.9**	75.2
`Hailfinder` (56)	61.3	59.4	80.3	**42.5**	-
`Hepar2` (70)	135.4	144.8	157.0	**118.2**	-
`Asia` (8)	1.1	3.8	7.3	**0.2**	5.8
`Sachs` (11)	**6.2**	14.0	15.5	10.3	8.3
`Child` (20)	**7.5**	13.3	14.4	10.2	19.6
`Insurance` (27)	39.8	36.3	49.2	**30.2**	45.6
`Alarm` (37)	34.0	23.6	35.0	**21.2**	36.2
`Hailfinder` (56)	52.9	56.4	55.6	**44.0**	-
`Hepar2` (70)	108.1	139.4	140.7	**96.8**	-

The upper panel illustrates cases where data are generated using threshold Gaussian models, while the lower panel corresponds to cases using multinomial models. In each network, the minimum average SHD is highlighted in bold.

Note that we specifically tuned the procedure for HC to combine local estimates and build a global graph, due to its lack of consensus among local estimates (Remark 2). It is encouraging to see that DARLS uniformly outperformed FGES, a consistent score-based method under the multinomial DAG model [56], highlighting the effectiveness of DARLS in using distributed data. This also suggests that the GLDAG (4) can be a pretty good approximation to the commonly used multinomial DAG model for discrete data. This study confirms that DARLS indeed performs relatively well on data generated from different DAG models, which is important for its practical use. This is further demonstrated by the application to a real dataset in next section.

Remark 3. The source code of DAG-GNN does not demonstrate how it handles categorical data with three or more categories, and hence we could not apply it directly to the multinomial data sets. For brevity, we relegate its results on the threshold-Gaussian case, which is binary, to the Supplemental Material. The SHD values of DAG-GNN are close to, or slightly higher, than the values of MMHC.

6 Real data application

In this section, we apply our method to the ChIP-Seq data generated by [68]. The dataset contains the DNA binding sites of 12 transcription factors (TFs) in mouse embryonic stem cells: *Smad1*, *Stat3*, *Sox2*, *Pou5f1*, *Nanog*, *Esrrb*, *Tcfcp2l1*, *Klf4*, *Zfx*, *E2f1*, *Myc*, and *Mycn*. For each TF, an association strength score, a weighted sum of the corresponding ChIP-Seq signal strength, was calculated for each of the 18,936 genes [69]. Roughly speaking, this score can be understood as a measure of the binding strength of a TF to a gene. Following the same preprocessing in [70], the genes with zero association scores were removed from our analysis. Accordingly, our observed data matrix, of size $n \times p = 8462 \times 12$, contains the association scores of 12 TFs over 8,462 genes. We aim to build a causal network that reveals how these 12 TFs affect each other's binding to genes. The associate scores of a TF are typically bimodal, and they were discretized before network estimation; see Figure S1 in the supplemental material for illustration of the discretization.

We distributed this dataset across $K = 20$ local clients, and hence each local client contains around 400 samples. Then we applied DARLS, HC, MMHC, PC and FGES to the distributed data to learn the protein-DNA binding network. NOTEARS and DAG-GNN were excluded in this comparison because their performance was not competitive as demonstrated in the simulation studies. Local estimates of a competing method were combined to construct a global graph as we did in Section 5. To ease the comparison, we controlled the sparsity of estimated networks such that every method produced two graphs using the distributed data, with roughly $\widehat{s}_0 = 17$ and 29 edges, except FGES which had difficulty generating output close to 17 edges. We also applied each method on the combined data (i.e, $K = 1$) with the same parameters in Section 5. In this case, each estimated graph had around $\widehat{s}_0 = 30$ edges. The only exception is PC, whose estimate had 21 edges even when its significance level had been reduced to 10^{-10}. Key parameters of each method are reported in Section S5 in the supplemental material.

Since the true network structure is unknown, test data likelihood under multinomial DAG models in ten-fold cross validation is used to assess the accuracy of estimated networks. Denote by $\widetilde{g}$ and g the likelihood values using training and test datasets, respectively, under multinomial DAG models (see supplemental material Section S5 for calculation of $\widetilde{g}$ and g). We also compute $\text{BIC} = -2\log\widetilde{g} + \log(\widetilde{n})\mathcal{N}(\widehat{\mathcal{G}})$ for model comparison, where $\widetilde{n}$ is the training sample size and $\mathcal{N}(\widehat{\mathcal{G}})$ is the number of multinomial parameters for estimated graph $\widehat{\mathcal{G}}$. We choose some benchmarks to ease comparison. Denote by g_{B} and BIC_{B} the highest test data likelihood and the lowest BIC value, respectively. Define the log-likelihood difference $\Delta(\log g) = \log g - \log g_{\text{B}}$ and the BIC difference $\Delta\text{BIC} = \text{BIC} - \text{BIC}_{\text{B}}$. Note that since $\log g$ is test data log-likelihood while BIC is calculated with training data, the magnitude of ΔBIC is much larger than $\Delta(\log g)$. We also compute the value of $\exp\{-\Delta\text{BIC}/(2\widetilde{n})\}$ as an approximation to the normalized marginal likelihood ratio (NLR) $(P(\widetilde{X} \mid \widehat{\mathcal{G}})/P(\widetilde{X} \mid \widehat{\mathcal{G}}_B))^{1/\widetilde{n}}$, where $\widetilde{X}$ denotes training data, between estimated DAGs $\widehat{\mathcal{G}}$ by a competing method and $\widehat{\mathcal{G}}_B$ by the BIC benchmark.

Table 3 summarizes $\widehat{s}_0$, $\Delta(\log g)$, ΔBIC and NLR of each method under three comparison settings, namely sparse, moderate and oracle. The first two settings report results with different degree of sparsity in estimated graphs using distributed data over $K = 20$ clients, and the last one shows results of the corresponding oracle solutions on the combined data (i.e., $K = 1$). In both sparse and moderate settings, DARLS achieves the highest test data likelihood and the smallest BIC, outperforming all the other methods by a substantial margin, which again demonstrates its effectiveness in DAG learning with distributed data. Oracle methods, except PC, have comparable test data likelihood values, all higher than their corresponding results on distributed data. Comparing the likelihood between combined and distributed data for each method, we see that DARLS shows the smallest difference. In other words, among all the methods, DARLS has the smallest loss when applied to distributed data as compared to its oracle result on the combined data.

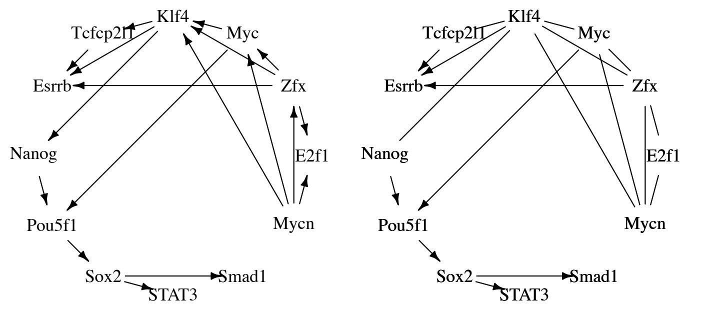


Fig. 3: DAG (left) and the converted CPDAG (right) learned by DARLS.

It is worth mentioning that the likelihood for each method is calculated under the multinomial DAG model, instead of the GLDAG model. Thus, the superior performance of the distributed learning by DARLS on this real-world data suggests that our proposed GLDAG model is a good approximation to the underlying data generation mechanism.

To gain more scientific insights, we show in Figure 3 the sparser DAG ($\widehat{s}_0 = 17$ in Table 3) and its converted CPDAG, learned by DARLS from the full dataset ($n = 8,462$) distributed over $K = 20$ local clients. An interesting observation is the directed path *Nanog*$\rightarrow$*Pou5f1*$\rightarrow$*Sox2* in the estimated CPDAG, among the three core regulators in the gene regulatory network in mouse embryonic stem cells [68, 71]. It is well-known that many genes are co-regulated by *Pou5f1*, *Sox2* and *Nanog*. The estimated path suggests that *Nanog* binding would cause the binding of *Pou5f1*, which then may cause *Sox2* binding. This provides new clue for how the three TFs work together to co-regulate downstream genes. Data analysis in [68], the original work that generated the ChIP-Seq data, suggests that there are two clusters of TFs that tend to co-bind: The first group consists of *Nanog*, *Sox2*, *Oct4*, *Smad1* and *STAT3*, while the second group includes *Mycn*, *Myc*, *Zfx* and *E2f1*. These two groups are clearly recovered in the estimated CPDAG, which contains a dense undirected subgraph on the second group of TFs and a fully directed subgraph on the first group. Moreover, the directed edge *Myc*$\rightarrow$*Pou5f1* indicates that the second group might be in the causal upstream of the first group, a novel hypothesis for potential experimental investigation.

7 Discussion

In this paper, we develop the DARLS algorithm that incorporates a distributed optimization method in simulated annealing to learn causal graphs from distributed data. Based on simulation studies and a real data application, we have shown that DARLS is highly competitive even when its model assumptions are violated. In its distributed optimization given an ordering, DARLS learns a causal graph by optimizing a convex penalized likelihood. In practice, one may consider concave penalties [30, 46] to improve accuracy when learning DAGs, although there may be lack of theoretical guarantees for convergence of distributed learning with a concave penalty. This is certainly a promising further development for our method.

TABLE 3: Comparison on the ChIP-Seq dataset. Best performance of each case is in bold.

Method	Sparse				Moderate				Oracle			
	$\widehat{s}_0$	$\Delta(\log g)$	ΔBIC	NLR	$\widehat{s}_0$	$\Delta(\log g)$	ΔBIC	NLR	$\widehat{s}_0$	$\Delta(\log g)$	ΔBIC	NLR
DARLS	16.5	$\mathbf{-136.9}$	**1051.6**	0.501	27.5	$\mathbf{-22.4}$	**297.7**	0.822	31.0	−3.7	540.2	0.701
MMHC	17.0	−149.6	2470.0	0.198	29.5	−44.4	1062.4	0.498	30.0	−1.7	**0.0**	1.000
FGES	11.5	−164.3	2478.5	0.196	28.0	−39.6	1026.6	0.510	34.0	−2.7	250.9	0.848
HC	17.0	−151.9	2510.1	0.192	29.0	−36.8	1174.9	0.462	30.0	**0.0**	7.9	0.995
PC	17.0	−156.3	2510.1	0.192	29.0	−36.9	942.6	0.538	21.0	−77.1	1047.8	0.503

Our proposed GLDAG model includes a family of flexible distributions besides linear Gaussian models (with equal variance), and thus can be applied to different types of data. It is also possible to generalize the framework of GLDAGs (4) to model nonlinear causal relations among variables. Consider scalar variables $X_j \in \mathbb{R}$ for simplicity. For each edge $X_i \rightarrow X_j$, we associate a nonlinear function $f_{ij}(x^i)$. Then the $\beta_j^\top x$ in (4) is replaced by $\sum_{i \in \mathrm{PA}_j} f_{ij}(x^i)$, leading to a generalized additive model for $[X_j \mid \mathrm{PA}_j]$. Such generalization is expected to approximate real causal relations with higher accuracy. We have established that continuous GLDAGs are identifiable, justifying their use in causal discovery, and it is left as future work to study the identifiability of general GLDAGs.

The primary focus of this paper is on big distributed data, with large n but moderate p. Under this setting, we established the convergence of the solution obtained by distributed optimization to a global minimizer of the loss (i.e. the oracle solution) and the consistency of the global minimizer as an estimator of the true DAG parameter. However, generalizing the convergence and consistency results to allow diverging p is theoretically interesting and left as future work.

Acknowledgment

This work was supported by NSF grants DMS-1952929, DMS-2305631 and DMS-1945667.

References

[1] D. B. Rubin, "Estimating Causal Effects from Large Data Sets Using Propensity Scores," *Annals of Internal Medicine*, vol. 127, pp. 757–763, 1997.

[2] G. W. Basse, A. Feller, and P. Toulis, "Randomization Tests of Causal Effects under Interference," *Biometrika*, vol. 106, pp. 487–494, 2019.

[3] S. L. Morgan and C. Winship, *Counterfactuals and Causal Inference*. Cambridge University Press, 2015.

[4] A. P. David, "Causal Inference without Counterfactuals," *Journal of the American Statistical Association*, vol. 95, no. 450, pp. 407–424, 2000.

[5] D. B. Rubin, "Causal Inference Using Potential Outcomes," *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 322–331, 2011.

[6] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*. Springer-Verlag, New York, 1993.

[7] S. L. Lauritzen, *Graphical Models*. Clarendon Press, 1996.

[8] M. M. Glymour, "Using Causal Diagrams to Understand Common Problems in Social Epidemiology," in *Methods in Social Epidemiology*. John Wiley and Sons, 2006, pp. 393–428.

[9] P. W. G. Tennant, E. J. Murray, K. F. Arnold, L. Berrie, M. P. Fox, S. C. Gadd, W. J. Harrison, C. Keeble, L. R. Ranker, J. Textor, G. D. Tomova, M. S. Gilthorpe, and G. T. H. Ellison, "Use of Directed Acyclic Graphs (DAGs) to Identify Confounders in Applied Health Research: Review and Recommendations," *International Journal of Epidemiology*, vol. 50, no. 2, pp. 620–632, 2021.

[10] X. Shen, S. Ma, P. Vemuri, and et al., "Challenges and Opportunities with Causal Discovery Algorithms: Application to Alzheimer's Pathophysiology," *Scientific Reports*, vol. 10, no. 1, p. 2975, 2020.

[11] G. W. Imbens, "Nonparametric Estimation of Average Treatment Effects under Exogeneity: A Review," *The Review of Economics and Statistics*, vol. 86, no. 1, pp. 4–29, 2004.

[12] H. Seiti, A. Makui, A. Hafezalkotob, M. Khalaj, and I. A. Hameed, "R.Graph: A New Risk-based Causal Reasoning and Its Application to COVID-19 Risk Analysis," *Process Safety and Environmental Protection*, vol. 159, pp. 585–604, 2022.

[13] J. Pearl, *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

[14] M. Drton and M. H. Maathuis, "Structure Learning in Graphical Modeling," *Annual Review of Statistics and Its Application*, vol. 4, no. 1, pp. 365–393, 2017.

[15] C. Heinze-Deml, M. H. Maathuis, and N. Meinshausen, "Causal Structure Learning," *Annual Review of Statistics and Its Application*, vol. 5, pp. 371–391, 2018.

[16] R. Guo, L. Cheng, J. Li, P. R. Hahn, and H. Liu, "A Survey of Learning Causality with Data: Problems and Methods," *ACM Computing Surveys*, vol. 53, no. 4, pp. 1–37, 2020.

[17] A. Mehmood, I. Natgunanathan, Y. Xiong, G. Hua, and S. Guo, "Protection of Big Data Privacy," *IEEE Access*, vol. 4, pp. 1821–1834, 2016.

[18] Apple and Google, "Explosure Notification Privacy-preserving Analytics (enpa) white paper," Apple and Google, Tech. Rep., 2021.

[19] Y. Zhang, J. C. Duchi, and M. J. Wainwright, "Communication-Efficient Algorithms for Statistical Optimization," *Journal of Machine Learning Research*, vol. 14, pp. 3321–3363, 2013.

[20] M. A. Zinkevich, M. Weimer, A. Smola, and L. Li, "Parallelized Stochastic Gradient Descent," in *Advances in Neural Information Processing Systems*, vol. 23, 2010,

pp. 2595–2603.

[21] O. Shamir, N. Srebro, and T. Zhang, "Communication-Efficient Distributed Optimization using an Approximate Newton-type Method," *Proceedings of the 31st International Conference on International Conference on Machine Learnings*, vol. 32, no. 2, pp. 1000–1008, 2014.

[22] M. I. Jordan, J. D. Lee, and Y. Yang, "Communication-Efficient Distributed Statistical Inference," *Journal of the American Statistical Association*, vol. 114, no. 526, pp. 668–681, 2018.

[23] J. Fan, Y. Guo, and K. Wang, "Communication-efficient accurate statistical estimation," *arXiv:1906.04870*, 2019.

[24] D. K. Molzahn, F. Dörfler, H. Sandberg, S. H. Low, S. Chakrabarti, R. Baldick, and J. Lavaei, "A Survey of Distributed Optimization and Control Algorithms for Electric Power Systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941–2962, 2017.

[25] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A Survey of Distributed Optimization," *Annual Reviews in Control*, vol. 47, pp. 278–305, 2019.

[26] Y. Na and J. Yang, "Distributed Bayesian Network Structure Learning," *IEEE International Symposium on Industrial Electronics*, pp. 1607–1611, 2010.

[27] K. X. Gou, G. X. Jun, and Z. Zhao, "Learning Bayesian Network Structure from Distributed Homogeneous Data," *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, vol. 3, pp. 250–254, 2007.

[28] Y. Tang, J. Wang, M. Nguyen, and I. Altintas, "PEnBayes: A Multi-layered Ensemble Approach for Learning Bayesian Network Structure from Big Data," *Sensors*, vol. 19, no. 20, p. 4400, 2019.

[29] F. Fu and Q. Zhou, "Learning Sparse Causal Gaussian Networks with Experimental Intervention: Regularization and Coordinate Descent," *Journal of the American Statistical Association*, vol. 108, pp. 288–300, 2013.

[30] B. Aragam and Q. Zhou, "Concave Penalized Estimation of Sparse Gaussian Bayesian Networks," *Journal of Machine Learning Research*, vol. 16, pp. 2273–2328, 2015.

[31] J. Gu, F. Fu, and Q. Zhou, "Penalized Estimation of Directed Acyclic Graphs from Discrete Data," *Statistics and Computing*, vol. 29, pp. 161–176, 2019.

[32] E. Gao, J. Chen, L. Shen, T. Liu, M. Gong, and H. Bondell, "FedDAG: Federated DAG Structure Learning," *Transactions of Machine Learning Research*, 2023.

[33] I. Ng and K. Zhang, "Towards federated Bayesian Network Structure Learning with Continuous Optimization," *International Conference on Artificial Intelligence and Statistics*, 2022.

[34] B. Aragam, J. Gu, and Q. Zhou, "Learning Large-Scale Bayesian Networks with the sparsebn Package," *Journal of Statistical Software*, vol. 91, no. 11, pp. 1–38, 2019.

[35] J. Pearl, "Causal Diagrams for Empirical Research," *Biometrika*, vol. 82, no. 4, pp. 669–710, 1995.

[36] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data," *Machine Learning*, vol. 20, pp. 197–243, 1995.

[37] P. O. Hoyer, D. Janzing, J. Mooij, J. Peters, and B. Schölkopf, "Nonlinear Causal Discovery with Additive Noise Models," *Preceedings of 21st International Conference on Neural Information Processing Systems*, pp. 689–696, 2008.

[38] J. Peters, J. M. Mooij, D. Janzing, and B. Schölkopf, "Causal Discovery with Continuous Additive Noise Models," *Journal of Machine Learning Research*, vol. 15, pp. 2009–2053, 2014.

[39] J. Peters and P. Bühlmann, "Identifiability of Gaussian Structural Models with Equal Error Variances," *Biometrika*, vol. 101, no. 1, pp. 219–228, 2014.

[40] P. Bühlmann, J. Peters, and J. Earnest, "Cam: Causal Additive Models, High-Dimensional Order Search and Penalized Regression," *The Annals of Statistics*, vol. 42, pp. 2526–2556, 2014.

[41] S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen, "A Linear Non-Gaussian Acyclic Model for Causal Discovery," *Journal of Machine Learning Research*, vol. 7, no. 72, pp. 2003–2030, 2006.

[42] M. Yuan and Y. Lin, "Model Selection and Estimation in Regression with Grouped Variables," *Journal of Royal Statistical Society, Series B*, vol. 68, no. 1, pp. 49–67, 2007.

[43] P. Larrañaga, M. Poza, Y. Yurramendi, R. H. Murga, and C. M. H. Kuijpers, "Structure Learning of Bayesian Networks by Genetic Algorithms: A Performance Analysis of Control Parameters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 9, pp. 912–926, 1996.

[44] N. Friedman and D. Koller, "Being Bayesian about Network Structure. A Bayesian Approach to Structure Discovery in Bayesian Networks," *Machine Learning*, vol. 50, pp. 95–125, 2003.

[45] M. Scanagatta, C. P. de Campos, G. Corani, and M. Zaffalon, "Learning Bayesian Networks with Thousands of Variables," in *Advances in Neural Information Processing Systems*, pp. 1864–1872, 2015.

[46] Q. Ye, A. A. Amini, and Q. Zhou, "Optimizing Regularized Cholesky Score for Order-Based Learning of Bayesian Networks," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 43, pp. 3555–3572, 2021.

[47] A. Beck and M. Teboulle, "A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems," *SIAM Journal on Imaging Scciences*, vol. 2, no. 1, pp. 183–202, 2009.

[48] Y. Nesterov, "Gradient Methods for Minimizing Composite Functions," *Mathematical Programming, Series B*, vol. 140, no. 125–161, 2013.

[49] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 123–231, 2013.

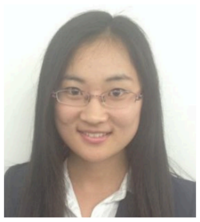
[50] X. Zheng, B. Aragam, P. Ravikumar, and E. P. Xing, "Dags with NO TEARS: Continuous Optimization for Structure Learning," in *Advances in Neural Information Processing Systems*, 2018.

[51] D. Wei, T. Gao, and Y. Yu, "DAGs with No Fears: A Closer Look at Continuous Optimization for Learning Bayesian Networks," *34th Conference on Neural Information Processing Systems*, 2020.

[52] M. Scutari, "Bayesian network repository," http://www.bnlearn.com/bnrepository/, 2007, accessed: 2020-05-01.

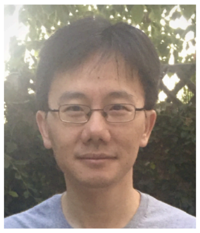
[53] J. A. Gámez, J. L. Mateo, and J. M. Puerta, "Learning

Bayesian Networks by Hill Climbing: Efficient Methods Based on Progressive Restriction of the Neighborhood," *Data Mining and Knowledge Discovery*, vol. 22, pp. 106–148, 2011.
[54] P. Spirtes and C. Glymour, "An Algorithm for Fast Recovery of Sparse Causal Graphs," *Social Science Computer Review*, vol. 9, no. 1, pp. 62–72, 1991.
[55] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm," *Machine Learning*, vol. 65, no. 1, pp. 31–78, 2006.
[56] D. M. Chickering, "Optimal Structure Identification with Greedy Search," *Journal of Machine Learning Research*, vol. 3, pp. 507–554, 2002.
[57] 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," *International Journal of Data Science and Analytics*, vol. 3, pp. 121–129, 2017.
[58] N. Ramanan and S. Natarajan, "Causal Learning from Predictive Modeling for Observational Data," *Frontiers in Big Data*, vol. 3, p. 535976, 2020.
[59] Y. Yu, J. Chen, T. Gao, and M. Yu, "DAG-GNN: DAG Structure Learning with Graph Neural Networks," *Preceedings of th 36th International Conference on Machine Learning*, vol. 97, pp. 7154–7163, 2019.
[60] C. Meek, "Strong Completeness and Faithfulness in Bayesian Networks," *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pp. 411–418, 1995.
[61] M. Scutari, "Learning Bayesian Networks with the bnlearn R Package," *Journal of Statistical Software*, vol. 35, no. 3, pp. 1–22, 2010.
[62] M. Kalisch, M. Mächler, D. Colombo, M. H. Maathuis, and P. Bühlmann, "Causal Inference Using Graphical Models with the R Package pcalg," *Journal of Statistical Software*, vol. 47, no. 11, pp. 1–26, 2012.
[63] X. Zheng, "Dags with no tears," https://github.com/xunzheng/notears, 2018, accessed: 2021-09-01.
[64] Y. Yu, J. Chen, T. Gao, and M. Yu, "Dag-gnn," https://github.com/fishmoon1234/DAG-GNN, 2019, accessed: 2022-07-01.
[65] N. Lee and J.-M. Kim, "Conversion of Categorical Variables into Numerical Variables via Bayesian Network Classifiers for Binary Classifications," *Computational Statistics Harold & Maude Data Analysis*, vol. 54, no. 5, pp. 1247–1265, 2010.
[66] Q. Cai, J. Kang, and T. Yu, "Bayesian Network Marker Selection via the Thresholded Graph Laplacian Gaussian Prior," *Bayesian Analysis*, vol. 15, pp. 79–102, 2020.
[67] Y. Song, X. Zhou, J. Kang, M. T. Aung, M. Zhang, W. Zhao, B. L. Needham, S. L. R. Kardia, Y. Liu, J. D. Meeker, J. A. Smith, and B. Mukherjee, "Bayesian Sparse Mediation Analysis with Targeted Penalization of Natural Indirect Effects," *Journal of the Royal Statistical Society. Series C, Applied statistics*, vol. 70, no. 5, pp. 1391–1412, 2021.
[68] X. Chen, H. Xu, P. Yuan, F. Fang, M. Huss, V. B. Vega, E. Wong, Y. L. Orlov, W. Zhang, J. Jiang, Y.-H. Loh, H. C. Yeo, Z. X. Yeo, V. Narang, K. R. Govindarajan, B. Leong, A. Shahab, Y. Ruan, G. Bourque, W.-K. Sung, N. D. Clarke, C.-L. Wei, and H.-H. Ng, "Integration of External Signaling Pathways with the Core Transcriptional Network in Embryonic Stem Cells," *Cell*, vol. 133, pp. 1106–17, 2008.
[69] Z. Ouyang, Q. Zhou, and W. H. Wong, "Chip-Seq of Transcription Factors Predicts Absolute and Differential Gene Expression in Embryonic Stem Cells," *Preceedings of the National Academic of Science of the United State of America*, vol. 106, no. 51, pp. 21 521–6, 2009.
[70] B. Wang and Q. Zhou, "Causal Network Learning with Non-intertible Functional Relationships," *Computational Statistics and Data Analysis*, vol. 156, p. 107141, 2021.
[71] Q. Zhou, H. Chipperfield, D. A. Melton, and W. H. Wong, "A Gene Regulatory Network in Mouse Embryonic Stem Cells," *Preceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 42, pp. 16 438–16 443, 2007.



Qiaoling Ye received her Ph.D in Statistics from UCLA in 2021. Her research interests include causal inference, community detection, and graph-based recommender system. She is currently a data scientist at Supplyframe, focusing on user behavior analysis, modeling, and online recommender systems.

Arash A. Amini received his Ph.D. in electrical engineering from University of California, Berkeley, in 2011. He is currently an Associate Professor of Statistics at UCLA. His research interests lie in high-dimensional statistics, network models, unsupervised and semisupervised learning, clustering and community detection, causal graphical models, kernel methods, representation learning, optimization and quantitative fiance.



Qing Zhou received his Ph.D. in Statistics from Harvard University in 2006. He is currently Professor of Statistics at UCLA. His research interests include causal inference, graphical models, machine learning, high-dimensional statistics, Monte Carlo methods, and bioinformatics. He received an NSF Career Award in 2011.