Learning Sparse Fixed-Structure Gaussian Bayesian Networks

Arnab Bhattacharyya [1] Davin Choo [1] Rishikesh Gajjala [2]
Sutanu Gayen [1] Yuhao Wang [1]

[1]National University of Singapore

[2]Indian Institute of Science, Bangalore

Table of Contents

1 Background

2 High-level approach

3 What's new: coefficients recovery
- Estimators based on least squares
- Estimator based on Cauchy random variables

4 Hardness results

5 Experiments

Bayesian Network

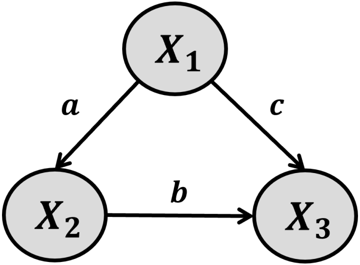
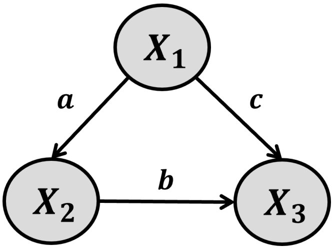


Figure: Bayesian Network

1. The DAG $G = (V, E)$
2. $V = \{X_1, \dots, X_n\}$
3. $(X_j, X_i) \in E$ whenever $X_j \to X_i$
4. For variable X_i with parent indices $\pi_i \subseteq [n]$
5. By Bayesian rule: $\mathcal{P}(X_1, \dots, X_n) = \prod_{i=1}^{n} \Pr_{\mathcal{P}}(X_i \mid \pi_i)$

Eg. $P(X_1, X_2, X_3) = P(X_1)P(X_2|X_1)p(X_3|X_1, X_2)$

Structural Equation Models with Gaussian Noise

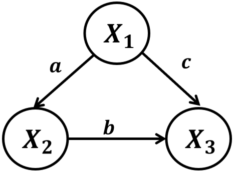


$$
\begin{aligned}
X_1 &= \eta_1, \\
X_2 &= aX_1 + \eta_2, \\
X_3 &= bX_2 + cX_3 + \eta_3 \\
\eta_i &\sim \mathcal{N}(0, \sigma_i^2)
\end{aligned}
$$

1. $\widehat{M}$ empirical covariance matrix of $X_1, \ldots, X_p$ with Cholesky decomposition $\widehat{M} = \widehat{L}\widehat{L}^\top$
2. $(X_1, \ldots, X_p) \sim N(0, M)$ is distributed as a multivariate Gaussian;
3. Structural equation model: $X_i = \eta_i + \sum_{j \in \pi_i} a_{i \leftarrow j} X_j, \ \eta_i \sim N(0, \sigma_i^2)$.

Objective

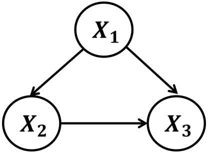
Goal: Parameter learning on a given graph structure



Ground truth with parameters $[a, b, c]$ and $[\sigma_1, \sigma_2, \sigma_3]$

X_1	X_2	X_3
$X_1^{(1)}$	$X_2^{(1)}$	$X_3^{(1)}$
...	...	...

Draw sample (possibly contaminated)



$+$ Given graph structure

Objective:

$$\underbrace{\text{Recover } [a, b, c]}_{\textbf{Parameter estimation}} \quad \underbrace{[\sigma_1, \sigma_2, \sigma_3]}_{\textbf{Variance recovery}} \quad \text{inducing } \hat{P} \text{ such that } \underbrace{d_{TV}(P, \hat{P}) \leq \epsilon}_{\textbf{Distance measure}};$$

Table of Contents

1 Background

2 High-level approach

3 What's new: coefficients recovery
- Estimators based on least squares
- Estimator based on Cauchy random variables

4 Hardness results

5 Experiments

The distance measure

- **Total variational (TV) distance:**
  $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) = \sup_{A \in \mathbb{R}^n} |\mathcal{P}(A) - \mathcal{Q}(A)| = \frac{1}{2} \int_{\mathbb{R}^n} |\mathcal{P}(x) - \mathcal{Q}(x)| \, dx.$
- **Kullback–Leibler (KL) divergence:**
  $\mathrm{d_{KL}}(\mathcal{P}, \mathcal{Q}) = \int_{A \in \mathbb{R}^n} \mathcal{P}(A) \log \left(\frac{\mathcal{P}(A)}{\mathcal{Q}(A)} \right) dA.$

Fact (Pinsker's inequality)

For distributions $\mathcal{P}$ and $\mathcal{Q}$, $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) \le \sqrt{\mathrm{d_{KL}}(\mathcal{P}, \mathcal{Q})/2}$.

If $s(\varepsilon)$ samples are needed to ensure $\underline{\mathrm{d_{KL}}(\mathcal{P}, \mathcal{Q}) \le \varepsilon}$
$\implies s(\varepsilon^2)$ samples are needed to ensure $\underline{\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) \le \varepsilon}$.

Decomposing KL divergence:

Decompose KL divergence into n terms:

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \sum_{i=1}^{n} \mathrm{d}_{\mathrm{CP}}(\alpha_i^*, \widehat{\alpha}_i)$$

$\implies$ Estimate parameters for each variable independently

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \sum_{i=1}^{n} \mathrm{d}_{\mathrm{CP}}(\alpha_i^*, \widehat{\alpha}_i) = \sum_{i=1}^{n} \ln\left(\frac{\widehat{\sigma}_i}{\sigma_i}\right) + \frac{\sigma_i^2 - \widehat{\sigma}_i^2}{2\widehat{\sigma}_i^2} + \frac{\Delta_i^\top M_i \Delta_i}{2\widehat{\sigma}_i^2}$$

- $\alpha_i^* = (A_i, \sigma_i)$: coefficients and variance associated with X_i
- $\widehat{\alpha}_i = (\widehat{A}_i, \widehat{\sigma}_i)$: estimates for α_i^*
- M_i: covariance matrix associated with X_i
- $\Delta_i = \widehat{A}_i - A_i$

Two-phased recovery approach

- **First phase:** Estimate the **coefficients** $\widehat{A}_1, \ldots, \widehat{A}_n$ of the Bayesian network
- **Second phase:** Recover **variances** $\widehat{\sigma}_y^2$ using empirical variances conditioned on our recovered coefficients (y refers to any arbitrary variable index).
- We are running the same algorithm for **each node**.

Coefficient estimators overview

High-level approach

- **There's no explicit sample complexity bound** known for Node-wise least square

Our contributions:

Algorithm	Remarks
`LeastSquares`	$\widetilde{\mathcal{O}}(nd/\varepsilon^2)$ samples within TV distance ε.
`BatchAvgLeastSquares`	Distributed-friendly generalization.
`CauchyEstTree`	$\widetilde{\mathcal{O}}(nd/\varepsilon^2)$ samples within TV distance ε.
`CauchyEst`*	Robust when data are contaminated

- All algorithms run in $poly(n, d, \log(\delta^{-1}))$ time with success probability at least $1 - \delta$.

$*$ No theoretical guarantees but empirically robust against contaminated samples

Table of Contents

1 Background

2 High-level approach

3 What's new: coefficients recovery
- Estimators based on least squares
- Estimator based on Cauchy random variables

4 Hardness results

5 Experiments

Vanilla least square

LeastSquares: using linear least squares.

$$\underbrace{\begin{pmatrix} X_1^{(1)} & \cdots & X_p^{(1)} \\ \vdots & \ddots & \vdots \\ X_1^{(m_1)} & \cdots & X_p^{(m_1)} \end{pmatrix}}_{X \in \mathbb{R}^{m_1 \times p}} \cdot \underbrace{\begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix}}_{A \in \mathbb{R}^p} = \underbrace{\begin{pmatrix} Y^{(1)} \\ \vdots \\ Y^{(m_1)} \end{pmatrix}}_{B \in \mathbb{R}^{m_1}}$$

Problem: $X\widehat{A} = B$

Solution:

$$\widehat{A} = (X^\top X)^{-1} X^\top B$$

No known analysis for the explicit **sample complexity bound**

Conclusion:: $\mathcal{O}(nd_{avg}\varepsilon^{-2} \cdot \log(n\delta^{-1}))$ samples within TV distance ε.

Batch average least squares

`BatchAvgLeastSquares`: any interpolation between **"batch size"** and **"number of batches"** as long as the **total number of samples** used is sufficiently large.

$$\text{batch 1} \begin{pmatrix} X_1^{(1)} & \cdots & X_p^{(1)} \\ \vdots & \ddots & \vdots \\ X_1^{(k)} & \cdots & X_p^{(k)} \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix} = \begin{pmatrix} Y^{(1)} \\ \vdots \\ Y^{(k)} \end{pmatrix}$$

... ...

$$\text{batch b} \underbrace{\begin{pmatrix} X_1^{((b-1)k+1)} & \cdots & X_p^{((b-1)k+1))} \\ \vdots & \ddots & \vdots \\ X_1^{(m_1)} & \cdots & X_p^{(m_1)} \end{pmatrix}}_{X \in \mathbb{R}^{k \times p}} \cdot \underbrace{\begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix}}_{A \in \mathbb{R}^p} = \underbrace{\begin{pmatrix} Y^{(((b-1)k+1))} \\ \vdots \\ Y^{(m_1)} \end{pmatrix}}_{b \in \mathbb{R}^k}$$

Notation: $\widehat{M}$ empirical covariance matrix of $X_1, \ldots, X_p$ with Cholesky decomposition $\widehat{M} = \widehat{L}\widehat{L}^\top$

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \sum_{i=1}^{n} \mathrm{d}_{\mathrm{CP}}(\alpha_i^*, \widehat{\alpha}_i) = \sum_{i=1}^{n} \ln\left(\frac{\widehat{\sigma}_i}{\sigma_i}\right) + \frac{\sigma_i^2 - \widehat{\sigma}_i^2}{2\widehat{\sigma}_i^2} + \underbrace{\frac{\Delta_i^\top M_i \Delta_i}{2\widehat{\sigma}_i^2}}$$

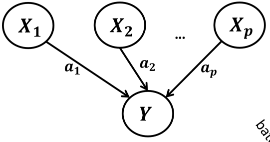
$$\Longrightarrow \left|\Delta^\top M \Delta\right| = \left|\Delta^\top L L^\top \Delta\right| = \left\|L^\top \Delta\right\|^2$$

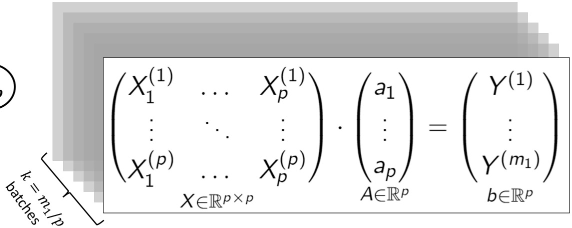
CauchyEstTree

Consider a batch estimate $\widetilde{A}$ and define $\Delta = \widetilde{A} - A$. If the Bayesian network is a polytree, then $\Delta_i = (\widetilde{A} - A)_i \sim \frac{\sigma_y}{\sigma_i} \cdot \mathrm{Cauchy}(0, 1)$ for all $i \in [n]$.

The study of **Cauchy random variables**

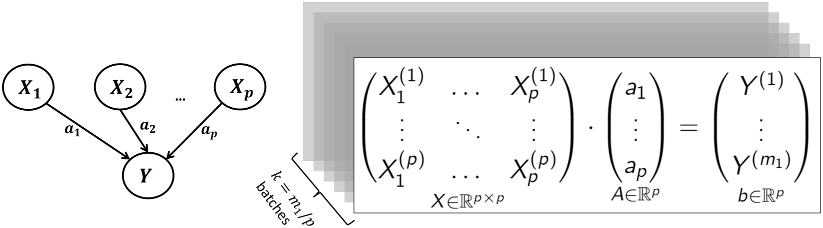
CauchyEstTree: Algorithm





- Compute $\widetilde{A} = [\widehat{a}_{y\leftarrow 1}, \dots, \widehat{a}_{y\leftarrow p}]^\top$ as any solution to $X\widetilde{A} = \left[Y^{(1)}, \dots, Y^{(p)}\right]^\top$

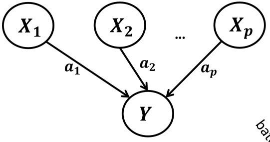
CauchyEstTree: Algorithm

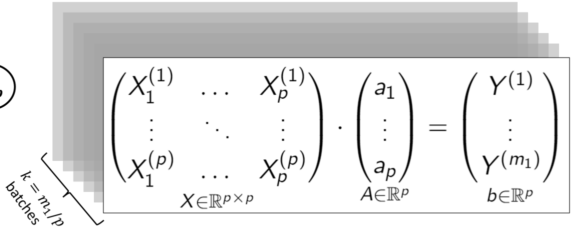


$$\begin{pmatrix} X_1^{(1)} & \dots & X_p^{(1)} \\ \vdots & \ddots & \vdots \\ X_1^{(p)} & \dots & X_p^{(p)} \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix} = \begin{pmatrix} Y^{(1)} \\ \vdots \\ Y^{(m_1)} \end{pmatrix}$$

- Compute $\widetilde{A} = [\widehat{a}_{y\leftarrow 1}, \dots, \widehat{a}_{y\leftarrow p}]^\top$ as any solution to $X\widetilde{A} = \left[Y^{(1)}, \dots, Y^{(p)}\right]^\top$
- $\mathtt{MED}_i$ as the median of $(\widehat{L}^\top \widetilde{A}^{(1)})_i, \dots, (\widehat{L}^\top \widetilde{A}^{(\lfloor m/p \rfloor)})_i$

CauchyEstTree: Algorithm





- Compute $\widetilde{A} = [\widehat{a}_{y\leftarrow 1}, \dots, \widehat{a}_{y\leftarrow p}]^\top$ as any solution to $X\widetilde{A} = \left[Y^{(1)}, \dots, Y^{(p)}\right]^\top$
- $\mathtt{MED}_i$ as the median of $(\widehat{L}^\top \widetilde{A}^{(1)})_i, \dots, (\widehat{L}^\top \widetilde{A}^{(\lfloor m/p \rfloor)})_i$
- $\widehat{A}^\top = (\widehat{L}^\top)^{-1} [\mathtt{MED}_1, \dots, \mathtt{MED}_n]^\top$

Cauchy-based estimators

Q: Why take median instead of mean?

A: The variance of a Cauchy variable is unbounded

Conclusion: $\mathcal{O}(nd_{avg}d\varepsilon^{-1} \cdot \log(n\delta^{-1}))$ samples within TV distance ε.

Generalization to random DAGs $\Longrightarrow$ `CauchyEst` estimator.

Limitation: No guarantees when DAG is not a tree.

Table of Contents

1 Background

2 High-level approach

3 What's new: coefficients recovery
- Estimators based on least squares
- Estimator based on Cauchy random variables

4 Hardness results

5 Experiments

Hardness results

Learning Gaussian product distributions

Given samples from a n-fold Gaussian product distribution P, learning a $\widehat{P}$ such that in $\mathrm{d_{TV}}(P, \widehat{P}) = O(\varepsilon)$ with success probability $2/3$ needs $\Omega(n\varepsilon^{-2})$ samples in general.

Learning Gaussian Bayesian networks

For any $0 < \varepsilon < 1$ and n, d such that $d \leq n/2$, there exists a DAG G over $[n]$ of in-degree d such that learning a Gaussian Bayesian network $\widehat{P}$ on G such that $\mathrm{d_{TV}}(P, \widehat{P}) \leq \varepsilon$ with success probability $2/3$ needs $\Omega(nd\varepsilon^{-2})$ samples in general.

Table of Contents

1 Background

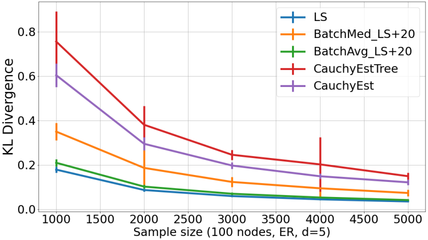
2 High-level approach

3 What's new: coefficients recovery
- Estimators based on least squares
- Estimator based on Cauchy random variables

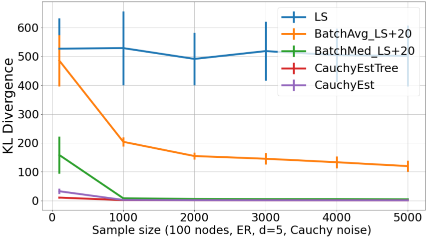
4 Hardness results

5 Experiments

Experiment results



Algorithms evaluated on ER graph with $d = 5$ on *uncontaminated* data



Algorithms evaluated on ER graph with $d = 5$ on *contaminated* data