Probabilistic Graphical Models & Probabilistic AI

Ben Lengerich

Lecture 10: Structure Learning

February 27, 2025

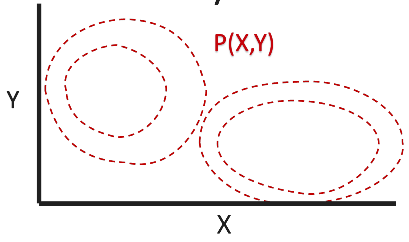
Reading: See course homepage

Logistics

- **Quiz** graded, up on Canvas
  - Office hours to review.
  - **Average:** 89%
  - **Min:** 59%
  - **Max:** 100%
- **HW3** due **Mar 1**
- **Next week:**
  - **Project Proposal** due **Mar 7**
  - **HW4** due **Mar 8**
- **Later:**
  - **HW5** due **Mar 15**
  - **Midterm – Mar 20**

A Brief Recap of our Roadmap

Probability Distribution



Observations

X	Y
0	0
0.1	1
0.2	1

Prior Knowledge

Parametric Family P_θ

Regularizer / Prior

Disc / Gen Models - Lecture 3

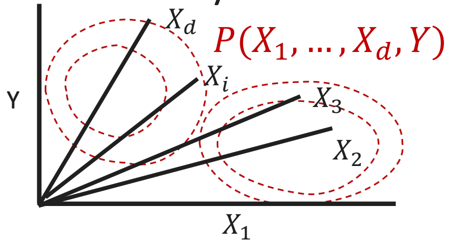
Estimation

$\hat{\theta}$

MLE / MAP - Lecture 2

A Brief Recap of our Roadmap

Probability Distribution



Observations

X	Y
0	0
0.1	1
0.2	1

Local Structure ← Prior Knowledge

Graphs - Lectures 4,5

Parametric Family P_θ

Regularizer / Prior

Disc / Gen Models - Lecture 3

Estimation → $\hat{\theta}$

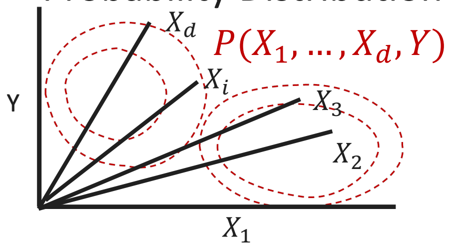
MLE / MAP - Lecture 2

Inference → $P_{\hat{\theta}}(\cdot)$

Exact - Lecture 6

A Brief Recap of our Roadmap

Probability Distribution



Observations

X	Y
0	0
0.1	1
0.2	1

Structure Learning - Lectures 10, 11

Local Structure

Architecture Design, Lectures 16-23

Prior Knowledge

Graphs - Lectures 4,5

Parametric Family P_θ

Regularizer / Prior

Disc / Gen Models - Lecture 3

Estimation

MLE / MAP - Lecture 2

Learning on Graphs - Lectures 7 - 9, 12-13

$\hat{\theta}$

Inference

Exact - Lecture 6

Approximate - Lecture 14

$P_{\hat{\theta}}(\cdot)$

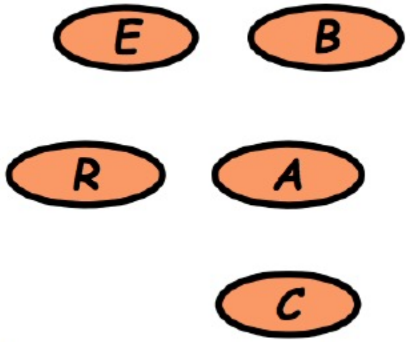
Today

- Structure Learning
  - Learning Tree BNs
  - Learning Pairwise MRFs

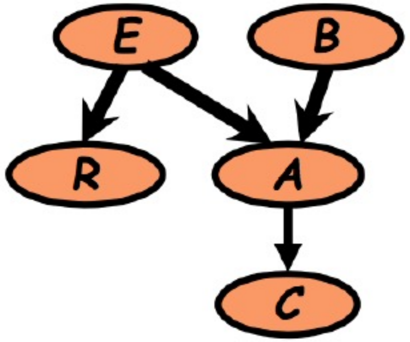
Structure Learning

Learning in Graphical Models

- Goal: Given a set of independent samples (**assignments** to random variables), find the **best** Bayesian Network (both DAG and CPDs)



(B,E,A,C,R) = (T,F,F,T,F)
(B,E,A,C,R) = (T,F,T,T,F)
…
(B,E,A,C,R) = (F,T,T,T,F)



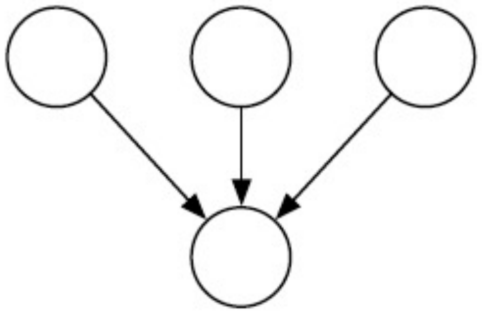
Structure learning

E	B	P(A | E,B)	
e	$\underline{b}$	0.9	0.1
$\underline{e}$	b	0.2	0.8
$\underline{e}$	$\underline{b}$	0.9	0.1
e	b	0.01	0.99

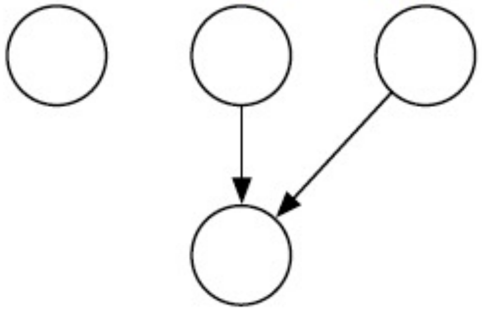
Parameter learning

Why aim for accurate structure?

True Structure

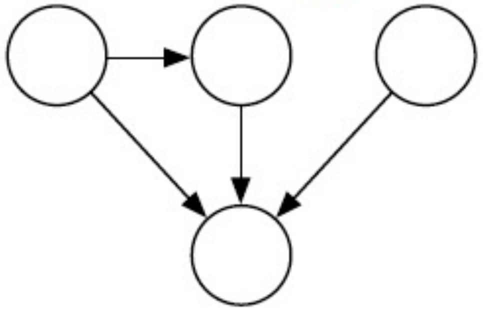


Missing Edge



- Cannot be compensated for by fitting parameters
- Wrong assumptions about domain structure

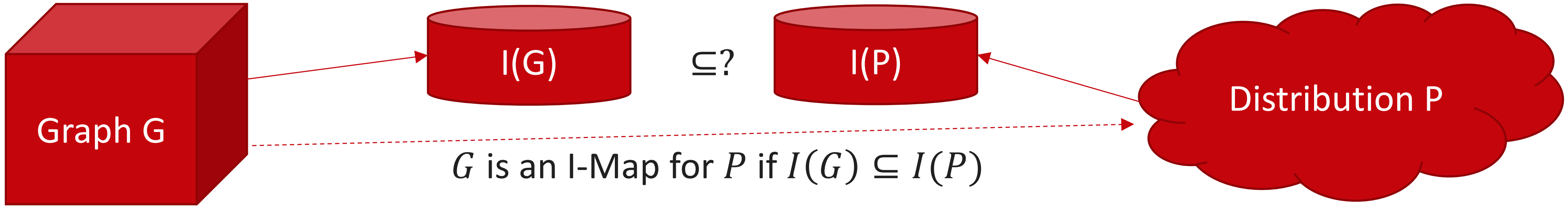
More Edges



- Increases the number of parameters to be estimated
- Wrong assumptions about domain structure

Recall: I-Maps

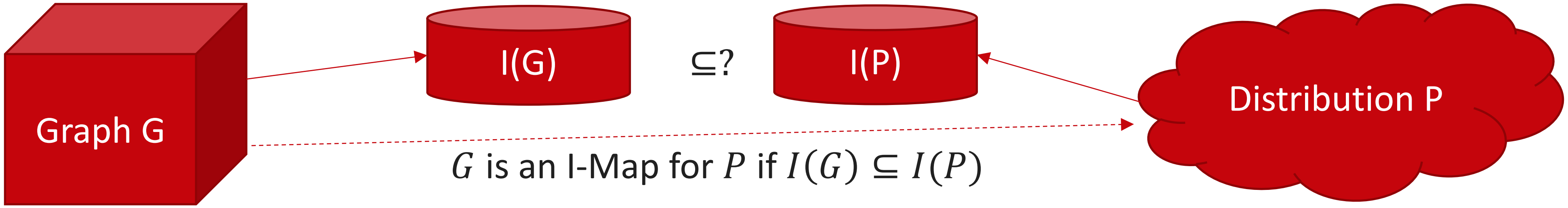
- Independence set: Let P be a distribution over X. We define $I(P)$ to be the set of independences $(X \perp Y \mid Z)$ that hold in P.
- I-Map: Let G be any graph object with an associated independence set $I(G)$. We say that G is an **I-map** for an independence set I if $I(G) \subseteq I$.
- I-Map of Distribution: We say G is an I-map for P if G is an I-map for $I(P)$, when we use $I(G)$ as the associated independence set.



Why does the graph get special privileges?

The I-Map view of Structure Learning

- We are looking for a graph G such that $I(G) \subseteq I(P)$
- This gets easier the looser we permit the inequality to be.
  - Trivial: fully-connected. No structure learning needed.
  - Hard: Perfect I-Map (no extra edges in graph).



G is an I-Map for P if $I(G) \subseteq I(P)$

Why does the graph get special privileges?

Information Theoretic Interpretation

$$
\begin{aligned}
\ell(\theta_G, G; D) &= \log p(D \mid \theta_G, G) \\
&= \log \prod_n \left(\prod_i p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}, \theta_{i|\pi_i(G)}) \right) \\
&= \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{n,\pi_i(G)}, \theta_{i|\pi_i(G)}) \right) \\
&= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \frac{count(x_i, \mathbf{x}_{\pi_i(G)})}{M} \log p(x_i \mid \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)}) \right) \\
&= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log p(x_i \mid \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)}) \right) \\
&= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log \frac{\hat{p}(x_i, \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)})}{\hat{p}(\mathbf{x}_{\pi_i(G)})} \frac{\hat{p}(x_i)}{\hat{p}(x_i)} \right) \\
&= M \sum_i \left(\sum_{x_i, \mathbf{x}_{\pi_i(G)}} \hat{p}(x_i, \mathbf{x}_{\pi_i(G)}) \log \frac{\hat{p}(x_i, \mathbf{x}_{\pi_i(G)}, \theta_{i|\pi_i(G)})}{\hat{p}(\mathbf{x}_{\pi_i(G)}) \hat{p}(x_i)} \right) - M \sum_i \left(\sum_{x_i} \hat{p}(x_i) \log \hat{p}(x_i) \right) \\
&= M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M \sum_i \hat{H}(x_i)
\end{aligned}
$$

Information Theoretic Interpretation

$$\ell(\theta_G, G; D) = \log p(D \mid \theta_G, G)$$
$$= M\sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M\sum_i \hat{H}(x_i)$$

Mutual information between x_i and its parents

Entropy of x_i

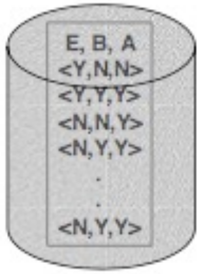
- As we match x_i and parents better, the mutual information increases.
- Problems?
- Adding edges always helps!

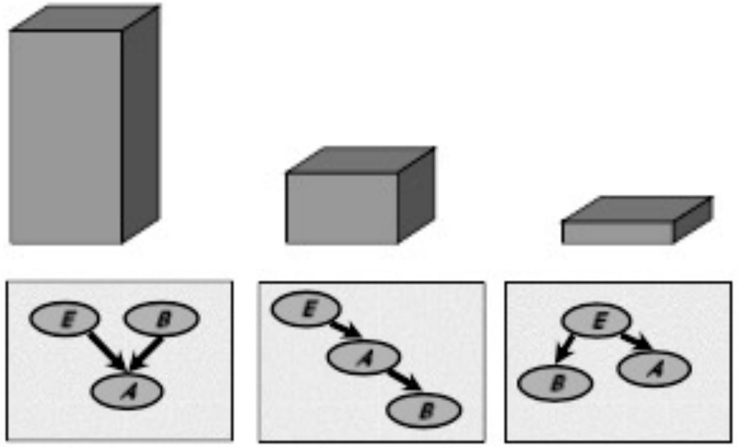
Different approaches to structure learning

- Two main problems:
  - **Likelihood** is **maximized for fully-connected graph**, so we don't want to just maximize likelihood alone.
  - Finding optimal BN structure is an **NP-hard** problem if allowed to be non-tree.
- Many heuristics but no "guarantees" of returning the perfect structure.
- Can get some guarantees if we make assumptions:
  - For tree BNs: Chow-Liu algorithm
  - For pairwise MRFs: Covariance selection, neighborhood-selection

Score-based Learning

- Define a scoring function that evaluates how well a structure matches the data:





- Search for a structure that maximizes the score

Bayesian Score

- Let's take a Bayesian approach
  - Place a distribution over our "uncertain" elements (G and θ)

Marginal likelihood

Prior over structures

$$P(G \mid D) = \frac{P(D \mid G)P(G)}{P(D)}$$

Marginal probability of Data

P(D) does not depend on the network

- Bayesian score for G

$$Score_B(G : D) = \log P(D \mid G) + \log P(G)$$

Bayesian Score cont'd

- Bayesian score for G

$$Score_B(G:D) = \log P(D|G) + \log P(G)$$

- Our choice of prior $P(G)$ has implications.
- Example: Let the edges have Dirichlet priors. Then as the number of configurations $M \to \infty$,

$$\log P(D|G) = l(\hat{\theta}_G : D) - \frac{\log M}{2} Dim(G) + O(1)$$

Dim(G): number of independent parameters in G

Tradeoff between fit to vs. data and complexity

Bayesian Information Criterion (BIC)

- Bayesian score gives Bayesian Information Criterion:

$$Score_{BIC}(G:D) = l(\hat{\theta}_G : D) - \frac{\log M}{2} Dim(G)$$

Structure Learning of Tree BNs

Tree BNs

- Let’s assume at most one parent per variable
- Why trees?
  - Sparse
  - No V-structures →
    - Can solve the optimization problem with a greedy algorithm

Chow-Liu tree learning algorithm

- Start by calculating Mutual Information between every pair of variables X_i and X_j

$$\hat{p}(X_i, X_j) = \frac{count(x_i, x_j)}{M}$$

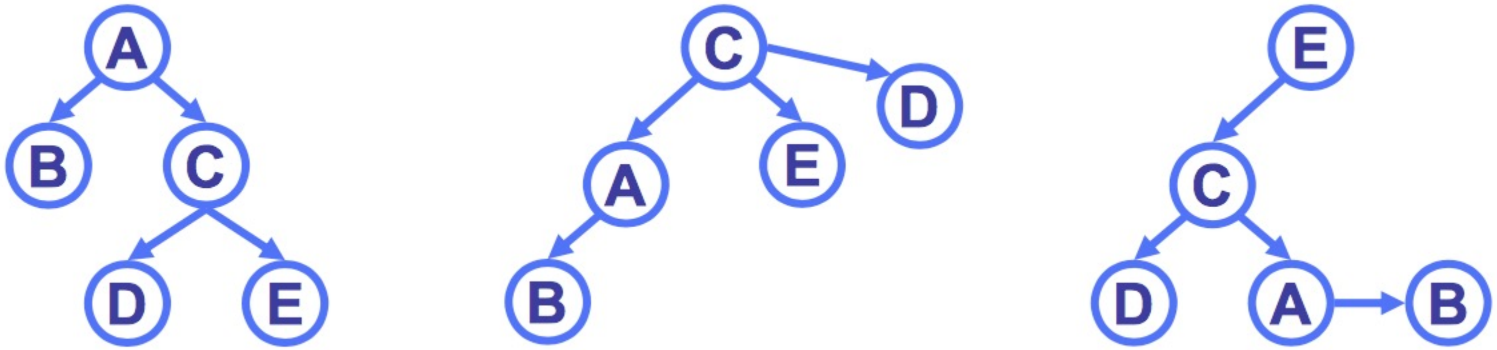
$$\hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)}$$

- Compute maximum weight spanning tree (Kruskal)
- Guarantees to maximize objective function:

$$\begin{aligned} \ell(\theta_G, G; D) &= \log \hat{p}(D \mid \theta_G, G) \\ &= M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \end{aligned} \quad \Rightarrow \quad \boxed{C(G) = M \sum_i \hat{I}(x_i, \mathbf{x}_{\pi_i(G)})}$$

Chow-Liu tree learning algorithm: directionality

- How to pick direction of edges?
- Pick any node as root, do BFS to define directions

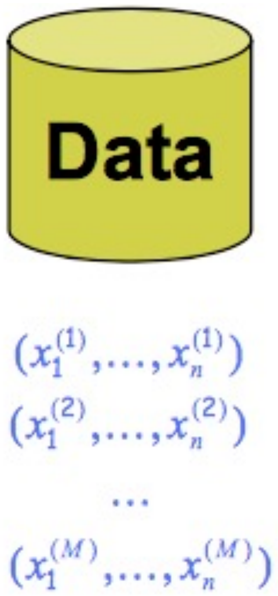


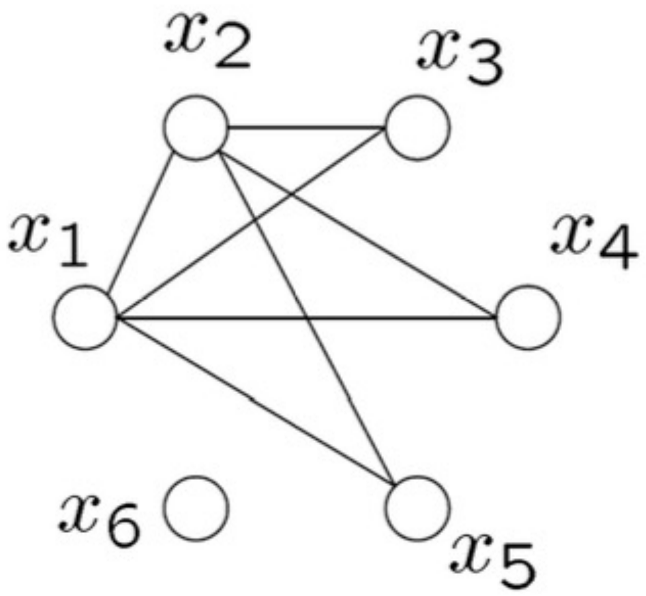
$$C(G) = I(A,B) + I(A,C) + I(C,D) + I(C,E)$$

- Can't tell the difference between competing root nodes

Structure Learning of UGMs

Learning Undirected Graphical Models





Pairwise MRFs

- Pairwise MRF:

$$P(X) \propto \prod_i \psi_i(X_i) \prod_{i,j} \psi_{i,j}(X_i, X_j)$$

- Gaussian Graphical Model:
  - Let $\psi_i(X_i) = \exp(\theta_i X_i)$, $\psi_{i,j}(X_i, X_j) = \exp(\theta_{ij} X_i X_j)$
  - Then:

$$\mathrm{P}(\mathrm{X} \mid \theta) \propto \exp\left(\sum_i \theta_i X_i + \sum_{i,j} \theta_{ij} X_i X_j\right)$$

Gaussian Graphical Model

- Gaussian Graphical Model:
  - Let $\psi_i(X_i) = \exp(\theta_i X_i)$, $\psi_{i,j}(X_i, X_j) = \exp(\theta_{ij} X_i X_j)$
  - Then:

$$\mathrm{P}(\mathrm{X} \mid \theta) \propto \exp\left(\sum_i \theta_i X_i + \sum_{i,j} \theta_{ij} X_i X_j\right)$$

- This is a Multivariate Gaussian density:

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp[-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)]$$

- for $\mu = 0$ and $\theta = \Sigma^{-1} = Q$.

The covariance and precision matrices

- Covariance matrix Σ

$$\Sigma_{i,j} = 0 \quad \Rightarrow \quad X_i \perp X_j \quad \text{or} \quad p(X_i, X_j) = p(X_i)p(X_j)$$

  - What is the graphical model interpretation?

Marginal independence / Correlation graph

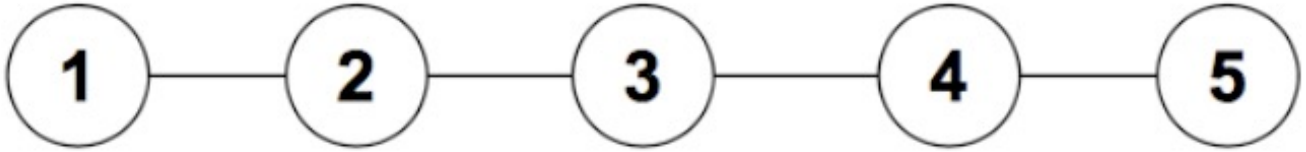
- Precision matrix $Q = \Sigma^{-1}$

$$Q_{i,j} = 0 \quad \Rightarrow \quad X_i \perp X_j | \mathbf{X}_{-ij} \quad \text{or} \quad p(X_i, X_j | \mathbf{X}_{-ij}) = p(X_i | \mathbf{X}_{-ij}) p(X_j | \mathbf{X}_{-ij})$$

  - What is the graphical model interpretation?

Conditional independence / Markov graph

Precision vs. Covariance



$$\Sigma^{-1} = \begin{pmatrix} 1 & 6 & 0 & 0 & 0 \\ 6 & 2 & 7 & 0 & 0 \\ 0 & 7 & 3 & 8 & 0 \\ 0 & 0 & 8 & 4 & 9 \\ 0 & 0 & 0 & 9 & 5 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 0.10 & 0.15 & -0.13 & -0.08 & 0.15 \\ 0.15 & -0.03 & 0.02 & 0.01 & -0.03 \\ -0.13 & 0.02 & 0.10 & 0.07 & -0.12 \\ -0.08 & 0.01 & 0.07 & -0.04 & 0.07 \\ 0.15 & -0.03 & -0.12 & 0.07 & 0.08 \end{pmatrix}$$

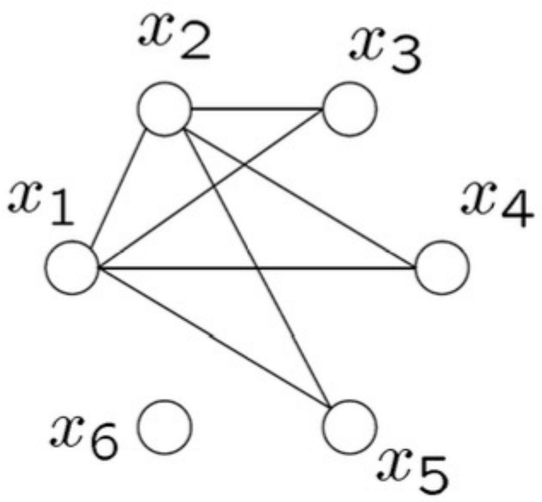
$$\Sigma^{-1}_{15} = 0 \Leftrightarrow X_1 \perp X_5 \,\middle|\, X_{nbrs(1) \text{ or } nbrs(5)}$$

$$\not\Rightarrow$$

$$X_1 \perp X_5 \Leftrightarrow \Sigma_{15} = 0$$

Example

$$Q = \begin{pmatrix} * & * & * & * & * & 0 \\ * & * & * & * & * & 0 \\ * & * & * & 0 & 0 & 0 \\ * & * & 0 & * & 0 & 0 \\ * & * & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & 0 & * \end{pmatrix}$$



If we can estimate a sample covariance, then we can estimate $Q = \hat{\Sigma}^{-1}$

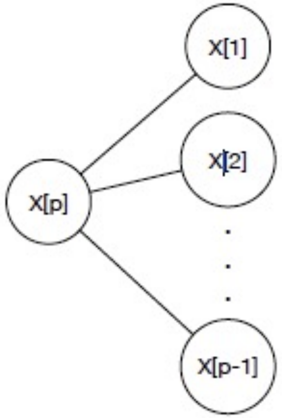
What if the number of dimensions > number of data points?

Recall Lasso

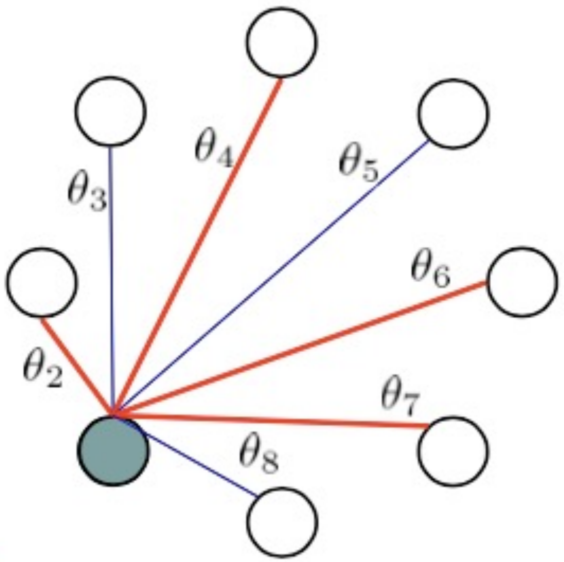
$$\hat{\theta}_i = \arg\min_{\theta_i} l(\theta_i) + \lambda_1 \| \theta_i \|_1$$

$$\text{where } l(\theta_i) = \log P(y_i | \mathbf{x}_i, \theta_i).$$

Let's apply Lasso for each graph regression:



Graph Regression



Neighborhood selection

Lasso:

$$\hat{\theta} = \arg\min_{\theta} \sum_{t=1}^{T} l(\theta) + \lambda_1 \| \theta \|_1$$

Gives graph structure.

Can we do graph regression for BNs?

Yes! Instead of raw Lasso, use matrix exponential to regularize toward DAG structure:

Theorem 1. *A matrix $W \in \mathbb{R}^{d \times d}$ is a DAG if and only if*

$$h(W) = \operatorname{tr}\left(e^{W \circ W}\right) - d = 0, \tag{5}$$

where $\circ$ is the Hadamard product and e^A is the matrix exponential of A. Moreover, $h(W)$ has a simple gradient

$$\nabla h(W) = \left(e^{W \circ W}\right)^T \circ 2W, \tag{6}$$

and satisfies all of the desiderata (a)-(d).

DAGs with NO TEARS:
Continuous Optimization for Structure Learning

Xun Zheng[1], **Bryon Aragam**[1], **Pradeep Ravikumar**[1], **Eric P. Xing**[1,2]
[1]Carnegie Mellon University [2]Petuum Inc.
{xunzheng,naragam,pradeepr,epxing}@cs.cmu.edu

Abstract

Estimating the structure of directed acyclic graphs (DAGs, also known as Bayesian networks) is a challenging problem since the search space of DAGs is combinatorial and scales superexponentially with the number of nodes. Existing approaches rely on various local heuristics for enforcing the acyclicity constraint. In this paper, we introduce a fundamentally different strategy: we formulate the structure learning problem as a purely *continuous* optimization problem over real matrices that avoids this combinatorial constraint entirely. This is achieved by a novel characterization of acyclicity that is not only smooth but also exact. The resulting problem can be efficiently solved by standard numerical algorithms, which also makes implementation effortless. The proposed method outperforms existing ones, without imposing any structural assumptions on the graph such as bounded treewidth or in-degree.

1 Introduction

Learning directed acyclic graphs (DAGs) from data is an NP-hard problem [8, 11], owing mainly to the combinatorial acyclicity constraint that is difficult to enforce efficiently. At the same time, DAGs are popular models in practice, with applications in biology [33], genetics [49], machine learning [22], and causal inference [42]. For this reason, the development of new methods for learning DAGs remains a central challenge in machine learning and statistics.

In this paper, we propose a new approach for score-based learning of DAGs by converting the traditional *combinatorial* optimization problem (left) into a *continuous* program (right):

$$\begin{array}{ll} \min\limits_{W \in \mathbb{R}^{d \times d}} & F(W) \\ \text{subject to} & \mathsf{G}(W) \in \mathsf{DAGs} \end{array} \iff \begin{array}{ll} \min\limits_{W \in \mathbb{R}^{d \times d}} & F(W) \\ \text{subject to} & h(W) = 0, \end{array} \tag{1}$$

where $\mathsf{G}(W)$ is the d-node graph induced by the weighted adjacency matrix W, $F : \mathbb{R}^{d \times d} \to \mathbb{R}$ is a score function (see Section 2.1 for details), and our key technical device $h : \mathbb{R}^{d \times d} \to \mathbb{R}$ is a smooth function over real matrices, whose level set at zero exactly characterizes acyclic graphs. Although the two problems are equivalent, the continuous program on the right eliminates the need for specialized algorithms that are tailored to search over the combinatorial space of DAGs. Instead, we are able to leverage standard numerical algorithms for constrained problems, which makes implementation particularly easy, not requiring any knowledge about graphical models. This is similar in spirit to the situation for undirected graphical models, in which the formulation of a continuous log-det program [4] sparked a series of remarkable advances in structure learning for undirected graphs (Section 2.2). Unlike undirected models, which can be reduced to a convex program, however, the program (1) is *nonconvex*. Nonetheless, as we will show, even naïve solutions to this program yield state-of-the-art results for learning DAGs.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

[Zheng 2018]

Questions?

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON