*Overview

Lec 1-4 • Graphical Models & Markov Properties

• Inference Problems : given $P_G(x)$ find $\left[\begin{array}{l} P(x_1) \\ \arg\max_x P(x) \\ \text{compute } Z. \end{array}\right.$

Lec 5-11 - Belief Propagation

Lec 12-14 - Variational methods

Lec 15-16 - Gibbs sampling

Lec 17-19 • Learning Graphical models : given samples $x^{(1)}, x^{(2)}, \dots, x^{(N)} \in \mathcal{X}^n$

- Learn the structure of the graph

* Structural learning.

- $X \in \mathcal{X}^n$ Random Vector.
- directed graphical model : $|E| = \binom{n}{2} = \frac{n(n-1)}{2}$

  # of possible DAGs $\leq 3^{|E|} = 3^{\frac{n(n-1)}{2}}$

- Given $X^{(1)}, \dots, X^{(N)}$ independent samples from unknown $P(x)$.
  - How do we score each graph?
  - How do we find the graph with highest score?

- There are 2 ways to approach such statistical problems

Frequentist

- Assume graph G and conditionals $P = \{P(X_i | X_{\pi_i})\}_{i=1}^{n}$ are deterministic but unknown

- Maximum Likelihood (ML) estimation finds (G, P) that maximizes log likelihood

$$\max_{G,P} \underbrace{\sum_{j=1}^{N} \log \mathbb{P}_{G,P}(X^{(j)})}_{SCORE(G,P)}$$

Bayesian

- Assume (G, P) is randomly drawn from some known prior distribution $Q_{G,P}$

- Maximum a Posteriori (MAP) estimation, which

$$\max_{G,P} \mathbb{P}(G,P \mid X^{(1)}, X^{(2)}, \dots, X^{(N)})$$

* Frequentist's approach to structural learning

$$\hat{G} = \arg\max_{G} \max_{P} \frac{1}{N} \sum_{i=1}^{N} \log P_{G,P}(X^{(i)})$$

*Simple Case with $n=2$. , $X=(X_1,X_2) \in \{0,1\}^2$, $\hat{P}_{12}(x_1,x_2)=$

samples $(0,0), (0,1), (1,1), (0.0)$

sufficient statistics = empirical distribution: $\hat{p}_1(x_1) = \begin{cases} 3/4 & , x_1=0 \\ 1/4 & , x_1=1 \end{cases}$, $\hat{p}_2(x_2) = \begin{cases} 1/2 & , x_2=0 \\ 1/2 & , x_2=1 \end{cases}$

case 1: for $G_1 =$ ① ②

the maximum likelihood estimate of $P_1(x_1), P_2(x_2)$ are

$$\max_{P_1} \boxed{\frac{1}{N}\sum_{j=1}^{N} \log P_1(x_1^{(j)})} + \max_{P_2} \frac{1}{N}\sum_{j=1}^{N} \log P_2(x_2^{(j)})$$

$\mathbb{I}(x_1^{(j)}=0)$

definition of empirical distribution → $= \max_{P_1=[-]} \boxed{\hat{P}_1(0)} \boxed{\log P_1(0)} + \hat{P}_1(1) \log P_1(1)$

$$= \max_{P_1} \sum_{x_1} \hat{P}_1(x_1) \log P_1(x_1)$$

adding/subtracting $H(\hat{P}_1)$ → $= \max_{P_1} \underbrace{\sum_{x_1} \hat{P}_1(x_1) \log \hat{P}_1(x_1)}_{-H_{\hat{P}_1}(X_1)} + \underbrace{\sum_{x_1} \hat{P}_1(x_1) \log \frac{P_1(x_1)}{\hat{P}_1(x_1)}}_{-D_{KL}(\hat{P}_1 || P_1)}$

is maximized when $\boxed{P_1 = \hat{P}_1}$ *empirical distribution is the maximum likelihood.

for G_1, (① ②) $\max_{P_1 \times P_2} \frac{1}{N}\sum_{j=1}^{N} \log P_1(x_1^{(j)}) P_2(x_2^{(j)})$ ← max achieved with $\left\{ \begin{matrix} P_1 = \hat{P}_1 \\ P_2 = \hat{P}_2 \end{matrix} \right\}$

$$\text{SCORE}(G_1) = \underbrace{\sum_{x_1} \hat{P}_1(x_1) \cdot \log \hat{P}_1(x_1)}_{-H(\hat{P}_1)} + \underbrace{\sum_{x_2} \hat{P}_2(x_2) \log \hat{P}_2(x_2)}_{-H(\hat{P}_2)}$$

Case 2: $G_2 =$ ① → ②

$$\max_{P_{12}(x_1,x_2)} \frac{1}{N}\sum_{j=1}^{N} \log P_{12}(x^{(j)}) = -H(\hat{P}_{12}) \underbrace{-D_{KL}(\hat{P}_{12} || P_{12})}_{\substack{\text{maximum} \\ \text{achieved} \\ \text{with } \hat{P}_{12} = P_{12}}}$$

$$\text{SCORE}(G_2) = -H(\hat{P}_{12})$$

$$\mathcal{L}(① \quad ②) = -H(\hat{P}_1) - H(\hat{P}_2)$$

$$\mathcal{L}(① \to ②) = -H(\hat{P}_{12})$$

$$H(\hat{P}_{12}) \leq H(\underbrace{\hat{P}_1 \cdot \hat{P}_2}_{\text{independt}})$$

· Remark 1: $-H(\hat{P}_{12}) \geq -H(\hat{P}_1) - H(\hat{P}_2)$

$G_2 > G_1$

hence, blindly chosing a more likely model results in overfitting.

↑ even if the true distribution was independent, we always chose dependent model.

• Remark 2: depending on the sample size N and the target false positive rate β, decision is made by

$$\underset{\uparrow \atop \text{depart } X_1, X_2}{\mathcal{L}(① \to ②)} - \underset{\uparrow \atop X_1 \perp\!\!\!\perp X_2}{\mathcal{L}(① \quad ②)} = H(\hat{P}_{12}) - H(\hat{P}_1) - H(\hat{P}_2) \triangleq \boxed{I_{\hat{P}_{12}}(X_1; X_2)}$$

output ①→② if $I_{\hat{P}_{12}}(X_1; X_2) > \frac{t_\beta}{N}$

① ② otherwise.

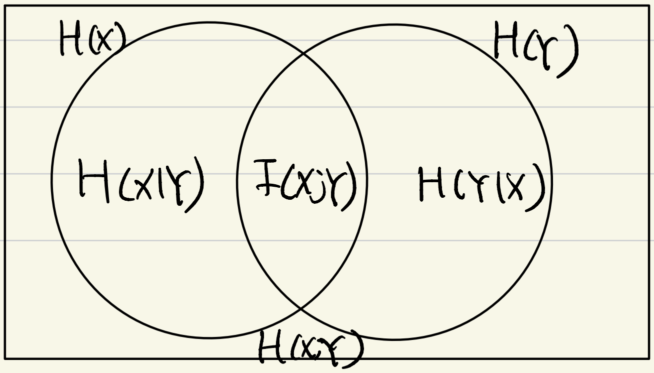
* We need to restrict the model class OR control false discovery rate

Refresh notations:

$$I(X;Y) \triangleq \sum_{XY} P(X,Y) \log \frac{P(X,Y)}{P(X) \cdot P(Y)}$$

$$H(X) \triangleq \sum_{X} -P(X) \log P(X)$$

$$H(Y|X) \triangleq \sum_{XY} -P(X,Y) \log P(Y|X)$$



$$H(Y|X) = H(Y) - I(X;Y)$$

$$H(X) + H(Y) = H(X,Y) + I(X;Y)$$

*Maximum Likelihood Approach for a DAG.

$$G^* = \arg\max_{G} \underbrace{\max_{\{P_i(X_i|X_{\pi_i})\}} \frac{1}{N}\sum_{j=1}^{N} \log \prod_{i=1}^{n} P_i(X_i|X_{\pi_i})}_{\text{the maximum is achieved at}}$$

$$P_i(X_i|X_{\pi_i}) = \hat{P}_i(X_i|X_{\pi_i})$$

↑ the empirical distribution

SCORE(G)

sufficient statistics.

$$\Rightarrow \frac{1}{N}\sum_{j=1}^{N} \log \prod_{i=1}^{n} \hat{P}_i(X_i|X_{\pi_i})$$

choice G

$$= \sum_{i=1}^{n} \frac{1}{N}\sum_{j=1}^{N} \log \hat{P}(X_i|X_{\pi_i})$$

def. empirical distribution →

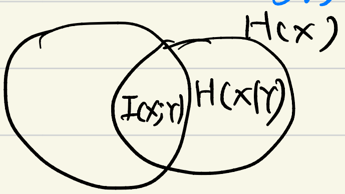
$$= \sum_{i=1}^{n} \left\{ \sum_{X_i, X_{\pi_i}} \hat{P}_i(X_i, X_{\pi_i}) \cdot \log \hat{P}_i(X_i|X_{\pi_i}) \right\}$$

$$= \sum_{i=1}^{n} \left\{ - H_{\hat{P}}(X_i|X_{\pi_i}) \right\}$$

$H(X)$
$= H(X|Y) + I(X;Y)$ →

$$= \sum_{i=1}^{n} \left\{ \underset{\uparrow}{I_{\hat{P}}(X_i ; X_{\pi_i})} - \underbrace{H_{\hat{P}}(X_i)}_{\text{Does not depend on } G.} \right\}$$

find G from a family of graphs that maximize this term



*Remark:
- this gives a "score" = likelihood for any given DAG G.
- We can now search over a class of graphs to find the best one.
- $I_{\hat{P}}(X_i ; X_{\pi_i}) \geq I_{\hat{P}}(X_i ; X_{\pi_i'})$ if $\pi_i \supset \pi_i'$ and hence denser graphs are preferred (and overfitted)
- So we need an appropriate class of graphs to search over.

*Chow-Liu algorithm: searches over all trees, efficiently.

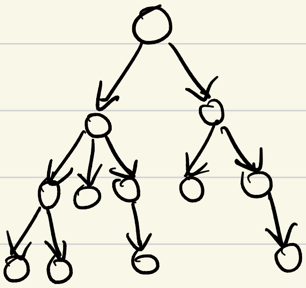
Step 1: Create a complete undirected graph over $V=\{1, \cdots, n\}$ with edge weights $I_{\hat{p}}(X_i, X_j) = w_{ij}$

Step 2: Use Kruskal's algorithm, for example, to find the max-weight spanning tree.

Claim: Chow-Liu algorithm finds the optimal tree that maximizes

$$\max_{G \in \text{Tree}} \sum_{i=1}^{n} I_{\hat{p}}(X_i ; \underset{\text{single node.}}{\underset{\|}{X_{\pi_i}}})$$

proof: each node only has one parent.



for general graphs, $n!$ orderings make it intractable.

*Another impractical approach for Undirected graph learning.

Consider learning an Ising Model $(G, \underset{\{\theta_i\}_{i\in V},\ \{\theta_{ij}\}_{(i,j)\in E}}{\theta})$, $\mathcal{X} = \{\pm 1\}$

from samples $\{\underset{\mathcal{X}^n}{x^{(1)}}, x^{(2)}, \dots, x^{(N)}\} = D$

the likelihood is

$$P_{(G,\theta)}(D) = \prod_{l=1}^{N} P_{(G,\theta)}(x^{(l)})$$

$$= \prod_{l=1}^{N} \frac{1}{Z_G(\theta)} \prod_{i\in V} e^{\theta_i \cdot x_i^{(l)}} \cdot \prod_{(i,j)\in E} e^{\underbrace{\theta_{ij} x_i^{(l)} x_j^{(l)}}}$$

$$= \exp\left\{ -N \log Z_G(\theta) + \sum_{i\in V} N \cdot \underset{\frac{1}{N}\sum_{l=1}^{N} x_i^{(l)}}{\underset{\triangleq}{\hat{M}_{ii}}}\, \theta_i + \underbrace{\sum_{(i,j)\in E}}_{G} N \cdot \underset{\frac{1}{N}\sum_{l=1}^{N} x_i^{(l)} x_j^{(l)}}{\underset{\triangleq}{\hat{M}_{ij}}}\, \theta_{ij} \right\}$$

the log-likelihood is

$$\min \quad L(G,\theta,D) = -\frac{1}{N} \log P_{(G,\theta)}(D) \qquad \langle M, \theta \rangle = \sum_{ij} M_{ij} \theta_{ij}$$

$$= \underset{\text{log-partition function}}{\Phi(\theta)} - \langle \hat{M}, \theta \rangle$$

$\log Z_G(\theta)$

$$n\left\{\underbrace{\begin{bmatrix} \hat{M}_{11} & \hat{M}_{12} & \cdots \\ & \hat{M}_{22} & \\ & & \end{bmatrix}}_{n}\right. \qquad \begin{bmatrix} \theta_1 & \theta_{12} & \cdot & 0 \cdots 0 \\ \vdots & \theta_2 & & 0 \\ \theta & 0 & \ddots & \\ 0 & 0 & & \ddots \end{bmatrix}$$

non-zero for $(i,j) \in E$ & (i,i).

Remark: this is strictly convex in θ,

but log-partition function requires inference.

* learning is easier when inference is easier.

but in general this is computationally intractable, even if the graph is sparse, requiring $O(|\mathcal{X}|^n)$ computations.

but continuing our (theoretical) investigation,
we want to apply this method to learn the structure of the graph as follows

$$\underset{G}{\text{minimize}}\ \underset{\theta}{\text{minimize}}\ \mathcal{L}(G, \theta, D)$$

As we did previously, we need to restrict our search to a class of "simple" graphs, as otherwise dense graphs always win. A natural condition is $|E| \leq m$.

$$\underset{\theta}{\text{minimize}}\ \mathcal{L}(K_n, \theta, D) \qquad \text{s.t.} \quad \|\theta\|_0 \leq m$$

, where K_n is the complete graph

$\|\theta\|_{L_0} = \sum_{i,j} \mathbb{I}(\theta_{ij} \neq 0)$

As $\|\theta\|_0$ constraint is intractable, people have proposed

$$\underset{\theta}{\text{minimize}}\ \underbrace{\overset{\downarrow}{\Phi}(\theta) - \langle \hat{\mu}, \theta \rangle}_{\mathcal{L}(K_n, \theta, D)} + \lambda \cdot \|\theta\|_1.$$

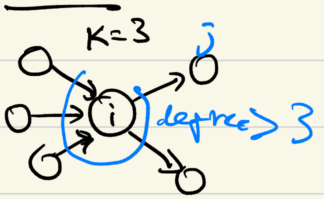
where $\|\theta\|_1 = \sum_{i,j} |\theta_{ij}|$

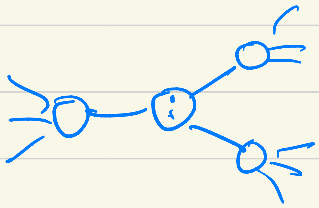
A different approach: Local Independence Test for undirected graphical models.

tries to take advantage of sparsity of the learned graph. to make learning faster than $O(|\mathcal{X}|^n)$.

```
Alg 1: Local Independence Test (samples $\{X^{(\ell)}\}_{\ell=1}^{N}$, neighborhood size $K$)
  - $E = \phi$
  - For each $i \in V$
  -     For each $S \subseteq V \setminus \{i\}$  s.t. $|S| \leq K$        $O(\binom{n}{K}) = O(n^K)$
  -         compute $SCORE(S,i) = H_{\hat{p}}(X_i | X_S)$
  -     Set $S^* \leftarrow \arg\min_S SCORE(S,i)$
  -     $E \leftarrow E \cup \{(i,j)\}_{j \in S^*}$

  - Prune the resulting graph.
```





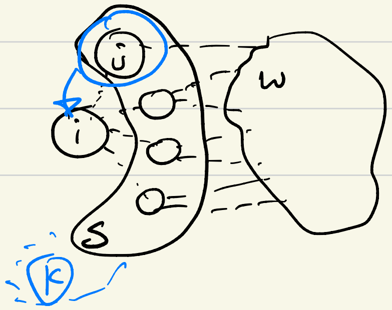
Remark 1: $X_i \perp\!\!\!\perp X_{rest} | X_S \iff$

$$H(X_i | X_S \cup X_{rest}) = H(X_i | X_S) \leq H(X_i | X_T) \quad \forall T \subset S.$$

Remark 2: Other scores have been proposed.

$$\underline{S}^* = \arg\max \{ |S| : \varepsilon < SCORE(S,i) \}$$

$$SCORE(S,i) = \min_{\substack{W \subseteq V \setminus S \\ j \in S}} \max_{\substack{x_i, x_W \\ x_S, x_j = a}} \Big| \hat{P}\{X_i = x_i | X_W = x_W, X_S = x_S\} - \hat{P}\{X_i = x_i | X_W = x_W, X_{S \setminus j} = x_{S \setminus j}, X_j = a\} \Big|$$



Remark: SCORE(S,i) will be small if it includes any non-significant node.

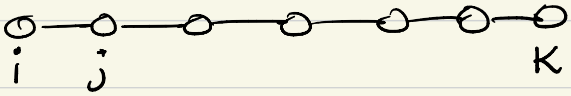
These approaches are still of more theoretical interest, as the run-time is $O(n^{k+1})$. $\ll O(|\mathcal{X}|^n)$

Here is a practical algorithm.

```
Alg 2: Thresholding ( samples $\{x^{(l)}\}_{l=1}^{N}$ , threshold $\tau$ )
  - Compute the empirical correlation $\{\hat{M}_{ij}\}_{(i,j)\in V\times V}$
  - For each $(i,j)\in V\times V$        $O(n^2)$
  -     If $|\hat{M}_{ij}| \geq \tau$ ,  set $(i,j)\in E$
```

where $\hat{M}_{ij} = \frac{1}{N}\sum_{l=1}^{N}\left(x_i^{(l)} - \bar{x}_i\right)\left(x_j^{(l)} - \bar{x}_j\right)$, $\bar{x}_i = \frac{1}{N}\sum_{l=1}^{N} x_i^{(l)}$

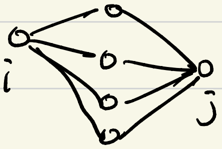
Remark: a heuristic based on the fact that two nodes faraway in the graph might be less correlated.



$\hat{M}_{iK} \ll \hat{M}_{ij}$

in general, this can fail if.

True graph



learned graph