CS839: Probabilistic Graphical Models

Lecture 15: Review

Theo Rekatsinas
Representing Multivariate Distributions

• If $X_i$’s are **conditionally independent** (as described by a PGM), the joint can be factored to a product of simpler terms, e.g.,

\[
P(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = P(X_1)P(X_2)P(X_3|X_1)P(X_4|X_2)P(X_5|X_2)P(X_6|X_3, X_4)P(X_7|X_6)P(X_8|X_5, X_6)
\]

• If $X_i$’s are **independent**: $P(X_i|\cdot) = P(X_i)$

\[
P(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = P(X_1)P(X_2)P(X_3)P(X_4)P(X_5)P(X_6)P(X_7)P(X_8)
\]
Local structure

• Common parent
  • Fixing B **decouples** A and C

• Cascade
  • Knowing B **decouples** A and C

• V-structure
  • Knowing C **couples** A and B (knowing one variable explains the contribution of the other to a common child event)

Conditional independences

\[ I(A \perp C | B) \]

\[ I(C \perp A | B) \]

Three foundational building blocks (compact language) for creating complex BNs
I-maps

• **Def:** Let $P$ be a distribution over $X$. We define $I(P)$ to be the set of independence assertions of the form $(X \perp Y | Z)$ that hold in $P$ (despite how we set the parameter-values).

• **Def:** Let $K$ be any graph object associated with a set of independencies $I(K)$. We say that $K$ is an I-map for a set of independencies $I$ if $I(K) \subseteq I$

• We say that graph $G$ is an I-map for $P$ if $G$ is an I-map for $I(P)$, where we use $I(G)$ as the set of independencies associated.
I-maps

• For F to be an I-map of P, it is necessary that G does not mislead us regarding independencies in P:
  • Any independence that G asserts must also hold in P. Conversely, P may have additional independencies that are not reflected in G

• Example

\[
\begin{array}{ccc|c}
X & Y & P(X, Y) \\
\hline
x^0 & y^0 & 0.08 \\
x^0 & y^1 & 0.32 \\
x^1 & y^0 & 0.12 \\
x^1 & y^1 & 0.48 \\
\end{array}
\]

\[
\begin{array}{ccc|c}
X & Y & P(X, Y) \\
\hline
x^0 & y^0 & 0.4 \\
x^0 & y^1 & 0.3 \\
x^1 & y^0 & 0.2 \\
x^1 & y^1 & 0.1 \\
\end{array}
\]
From I(G) to local Markov assumptions of BNs

• A BN with structure G is a directed acyclic graph whose nodes represent random variables $X_1, \ldots, X_n$.

• **Local Markov assumptions** (one way to specify independencies: )

• Def:

   Let $Pa_{X_i}$ denote the parents of $X_i$ in $G$ and $NonDescendants_{X_i}$ denote the variables in the graph that are not descendants of $X_i$. Then $G$ encodes the following set of **local conditional independence assumptions** $I_l(G)$:

   $$I_l(G) : \{X_i \perp NonDescendants_{X_i} | Pa_{X_i} : \forall I\}$$

   In other words, each node $X_i$ is independent of its non-descendants given its parents.
Global Markov properties of BN

- X is d-separated (directed-separated) from Z given Y if we cannot send a ball from any node in X to any node in Z using the “Bayes-ball” algorithm.

\[ I(G) = \left\{ X \perp Z \mid Y : dsep_G(X; Z \mid Y) \right\} \]
Perfect Maps (P-Maps)

- A DAG $G$ is a **perfect map** (P-map) for a distribution $P$ if $I(P) = I(G)$

- **Thm**: Not every distribution has a perfect map as DAG.
- **Proof**: Suppose we have a model where $A \perp C \mid \{B, D\}$ and $B \perp D \mid \{A, C\}$

*Can we represent this with a BN?* Cannot be represented by any BN
Section 1

Representation

• **Def:** an **undirected graphical model** represents a distribution \( P(X_1, \ldots, X_n) \) defined by an undirected graph \( H \), and a set of positive **potential functions** \( \psi \) associated with the cliques of \( H \), s.t.

\[
P(X_1, \ldots, X_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(X_c)
\]

where \( Z \) is known as the partition function:

\[
Z = \sum_{X_1, \ldots, X_n} \prod_{c \in C} \psi_c(X_c)
\]

• A.K.A. Markov Random Fields, Markov networks

• The potential function can be understood as a “score” of the joint configuration
Global Markov Independencies

Let $H$ be an undirected graph:

- $B$ separates $A$ and $C$ if every path from a node in $A$ to a node in $C$ passes through a node in $B$: $sep_H(A; C|B)$

A probability distribution satisfies the **global Markov property** if for any disjoint $A$, $B$, $C$, such that $B$ separates $A$ and $C$, $A$ is independent of $C$ given $B$: $I(H) = \{A \perp C|B : sep_H(A; C|B)\}$
Local Markov Independencies

• For each node $X_i \in V$ there is a unique Markov blanket of $X_i$ denoted $MB_{X_i}$ which is the set of neighbors of $X_i$ in the graph (those that share an edge with $X_i$).

• Def:

The local Markov independencies associated with $H$ is:

$$I(H) : \{X_i \perp V \setminus \{X_i\} \mid MB_{X_i} : \forall i\}$$

In other words, $X_i$ is independent of the rest of the notes in the graph given its direct neighbors.
Example MRF – subcliques

\[ P''(x_1, x_2, x_3, x_4) = \frac{1}{Z} \prod_{ij} \psi_{ij}(x_{ij}) \]
\[ = \frac{1}{Z} \psi_{12}(x_{12}) \psi_{14}(x_{14}) \psi_{23}(x_{23}) \psi_{24}(x_{24}) \psi_{34}(x_{34}) \]

- We can represent \( P(X_1, X_2, X_3, X_4) \) as 5 2D tables instead of one 4D table.
- Pair MRFs, a popular simple special case
- \( I(P') = I(P'')? \)
- \( D(P') \supset D(P'')? \)
Conditional Models

• Model conditional probability $P(\text{label sequence } Y \mid \text{observation sequence } X)$ rather than joint probability $P(X,Y)$
  • Specify the probability of possible label sequences given an observation sequence

• Allow arbitrary, non-independent features on the observation sequence $X$

• The probability of a transition between labels may depend on past and future observations

• Relax strong independent assumption in generative models
Conditional Distribution

- CRFs use the observation-dependent normalization $Z(x)$ for the conditional distributions

$$P_{\theta}(y|x) = \frac{1}{Z(x)} \exp \left( \sum_{e \in E,k} \lambda_k f_k(e, y|e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|v, x) \right)$$
Probabilistic Inference and Learning

- We discussed compact representations of probability distributions: **Graphical Models (GMs)**
- A GM (fully parameterized) $M$ describes a unique probability distribution $P$
- Tasks we can perform with a GM:
  - **Task 1 (Inference):** What is the probability $P_M(X|Y)$? $Y$: evidence
  - **Task 2 (Learning):** How do we estimate a **plausible** model $M$ from data $D$?
    - Learning refers to obtaining a point estimate of $M$
    - Bayesian’s are looking for $P(M|D)$ which is actually an inference problem.
    - Missing data: to compute a point estimate of $M$ we need to perform inference to impute the missing data.
Likelihood

• Most queries involve evidence
  • Evidence $e$ is an assignment of values to a set $E$ variables in the domain
  • Without loss of generality $E = \{X_{k+1}, ..., X_n\}$

• Example: compute the probability of evidence $e$

$$ P(e) = \sum_{x_1} \cdots \sum_{x_k} P(x_1, \ldots, x_k, e) $$

• A.k.a. compute the likelihood of $e$
Conditional Probability

• Often we are interested in the **conditional probability distribution** of a variable given the evidence:

\[ P(X|e) = \frac{P(X, e)}{P(e)} = \frac{P(X, e)}{\sum_x P(X = x, e)} \]

  • Aka **a posteriori belief** in \( X \), given evidence \( e \)

• Most of the times we only care about a subset \( Y \) of all domain variables \( X=\{Y,Z\} \) and do not care about the remaining \( Z \),

\[ P(Y|e) = \sum_z P(Y, Z = z|e) \]

  • the process of summing out “do not care” variables is called marginalization, and the resulting probability is called a marginal probability
Elimination on Chains

• Consider the following GM:

  • What is the likelihood that E is true?

• Reorder terms

\[
P(e) = \sum_d \sum_c \sum_b \sum_a \sum P(a)P(b|a)P(c|b)P(d|c)P(e|d)
\]

\[
= \sum_d \sum_c \sum_b \sum \sum P(c|b)P(d|c)P(e|d) \sum_a P(a)P(b|a)
\]
Elimination on Chains

• Consider the following GM:

\[
P(e) = \sum_d \sum_c \sum_b P(c|b) P(d|c) P(e|d) \sum_a P(a) P(b|a)
\]

\[
= \sum_d \sum_c \sum_b P(c|b) P(d|c) P(e|d) p(b)
\]

• This summation eliminates one variable from our summation argument at a local cost
Elimination on Chains

• Consider the following GM:

• Perform innermost summation

\[
P(e) = \sum_d \sum_c \sum_b P(c|b)P(d|c)P(e|d)p(b)
\]

\[
= \sum_d \sum_c P(d|c)P(e|d) \sum_b P(c|b)p(b)
\]

\[
= \sum_d \sum_c P(d|c)P(e|d)p(c)
\]
Elimination on Chains

• Consider the following GM:

• Eliminate nodes one-by-one all the way to the end

\[ P(e) = \sum_{d} P(e|d)p(d) \]

• Complexity:
  • For each step we have \( O(|\text{Dom}(X_i)| \times |\text{Dom}(X_{i+1})|) \) operations: \( O(kn^2) \)
  • Compare with naïve \( O(n^k) \)
Undirected Chains

• **Rearranging terms...**

\[
P(e) = \sum_{d} \sum_{c} \sum_{b} \sum_{a} \frac{1}{Z} \phi(b, a) \phi(c, b) \phi(d, c) \phi(e, d)
\]

\[
= \frac{1}{Z} \sum_{d} \sum_{c} \sum_{b} \phi(c, b) \phi(d, c) \phi(e, d) \sum_{a} \phi(b, a)
\]

\[
= \ldots
\]
Variable Elimination Example

Query: \( P(A|h) \)

- Need to eliminate: \( B, C, D, E, F, G, H \)

Initial factors:

\[
P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)
\]

Step 1:

- **Conditioning** on evidence (fix \( H \) to \( h \))

\[
\rho_H(E, F) = P(H = h|E, F)
\]

- Same as a marginalization step:

\[
\rho_H(E, F) = \sum_{h'} P(H = h'|E, F)\delta(h' = h)
\]
Variable Elimination Example

Query: $P(A|h)$

- Need to eliminate: B, C, D, E, F, G, H

Initial factors:

$P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)p_H(E,F)$

Step 2: Eliminate $G$

$$p_G(E) = \sum_g P(G = g|E) = 1$$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)p(H(E,F)$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A) p_H(E,F)$
Variable Elimination Example

Query: $P(A|h)$
• Need to eliminate: B, C, D, E, F, G, H

Initial factors:
$P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P_H(E,F)$

**Step 3: Eliminate F**

$$P_H(E, A) = \sum_f P(F = f | A) P_H(E, F)$$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D) P_F(A,E)$
Variable Elimination Example

Query: $P(A|h)$

- Need to eliminate: B, C, D, E, F, G, H

Initial factors:

$P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)P(E|C,D)p_F(A,E)$

**Step 4: Eliminate E**

$$p_E(A, C, D) = \sum_e P(E = e|C, D)p_F(A, E)$$

$\Rightarrow P(A)P(B)P(C|B)P(D|A)p_E(A,C,D)$
Variable Elimination Example

Query: P(A|h)

• Need to eliminate: B, C, D, E, F, G, H

Initial factors:

P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)

=> P(A)P(B)P(C|B)P(D|A)p_E(A,C,D)

Step 5: Eliminate D

\[ p_D(A, C) = \sum_d P(D = d|A)p_E(A, C, D) \]

=> P(A)P(B)P(C|B) p_D(A,C)
Variable Elimination Example

Query: \( P(A|h) \)

- Need to eliminate: B, C, D, E, F, G, H

Initial factors:
\[
P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)
\]
\[
\Rightarrow P(A)P(B)P(C|B) \quad p_D(A,C)
\]

**Step 6: Eliminate C**

\[
p_C(A, B) = \sum_c P(C = c|B)p_D(A, C)
\]
\[
\Rightarrow P(A)P(B)P(C|B) \quad p_C(A,B)
\]
Variable Elimination Example

Query: \( P(A|h) \)

- Need to eliminate: B, C, D, E, F, G, H

Initial factors:
\[
P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)
\]
\[
\Rightarrow P(A)P(B)p_C(A,B)
\]

**Step 7: Eliminate B**

\[
p_B(A) = \sum_b P(B = b|A)p_C(A, B)
\]
\[
\Rightarrow P(A)p_B(A)
\]
Variable Elimination Example

Query: \( P(A|h) \)

• Need to eliminate: B, C, D, E, F, G, H

Initial factors:

\[
P(A)P(B)P(C|B)P(D|A)P(E|C,D)P(F|A)P(G|E)P(H|E,F)
\]

=> \( P(A)p_B(A) \)

**Step 8: Wrap-up**

\[
P(A, h) = P(A)p_B(A), P(h) = \sum_a P(A = a)p_B(A = a)
\]

\[
P(A|h) = \frac{P(A,h)}{P(h)}
\]
From VE to Message Passing

• Elimination = message passing on a clique tree

\[ m_e(a,c,d) = \sum_e p(e|c,d)m_g(e)m_f(a,e) \]

• What if we want to find the probability of G?
2-pass Algorithm

• A dynamic programming algorithm
Max-Product Algorithm: MAP Probabilities

\[
\max_{\mathbf{x}} p(\mathbf{x}) = \max_{\mathbf{x}_f} (\psi(\mathbf{x}_f) m_{if}(\mathbf{x}_f))
\]

\[
m_{if}(\mathbf{x}_f) = \max_{\mathbf{x}_i} \left( \psi(\mathbf{x}_j) \psi(\mathbf{x}_i, \mathbf{x}_j) \prod_{k \in N(j) - i} m_{kj}(\mathbf{x}_j) \right)
\]
Max-Product Algorithm: MAP States

\[ x_f^* = \arg \max_{x_f} (\psi(x_f) m_{i_f}(x_f)) \]

\[ x_i^* = \arg \max_{x_i} (\psi(x_i) \psi(x_f, x_i) m_{j_i}(x_i)) \]

\[ x_j^* = \arg \max_{x_j} (\psi(x_j) \psi(x_i, x_j) m_{k_j}(x_j) m_{l_j}(x_j)) \]

\[ x_k^* = \arg \max_{x_k} (\psi(x_k) \psi(x_k, x_j^*)) \]

Compute MAP state with a backward pass.
Clique Tree

• A clique tree is an (undirected) tree of cliques

\[
P(X_1, X_2, X_3, X_4, X_5, X_6) \\
= P(X_1)P(X_2)P(X_3 | X_1, X_2)P(X_4 | X_3)P(X_5 | X_3)P(X_6 | X_4, X_5) \\
= P(X_1, X_2, X_3) \frac{P(X_3, X_4, X_5)}{P(X_3)} \frac{P(X_4, X_5, X_6)}{P(X_4, X_5)} \\
= \psi(X_1, X_2, X_3) \frac{\psi(X_3, X_4, X_5)}{\phi(X_3)} \frac{\psi(X_4, X_5, X_6)}{\phi(X_4, X_5)} \\
\]

Now each potential is isomorphic to the \textit{cluster marginal} of the attendant set of variables.
Local Consistency

• We have two ways of obtaining $p(S)$

$$P(S) = \sum_{V \setminus S} \psi(V) \quad P(S) = \sum_{W \setminus S} \psi(W)$$

• The following update-rule ensures this:

- Forward update:  
  $$\phi^*_S = \sum_{V \setminus S} \psi^*_V \quad \psi^*_W = \frac{\phi^*_S}{\phi^*} \psi_W$$

- Backward update  
  $$\phi^{**}_S = \sum_{W \setminus S} \psi^*_W \quad \psi^{**}_V = \frac{\phi^{**}_S}{\phi^{**}} \psi^*_V$$

• Two important identities

$$\sum_{V \setminus S} \psi^*_V = \sum_{W \setminus S} \psi^*_W = \phi^*_S$$  

Local Consistency

$$\frac{\psi^*_V \psi^*_W}{\phi^*_S} = \frac{\psi^{**}_V \psi^{**}_W}{\phi^{**}_S} = \frac{\psi_V \psi_W}{\phi_S}$$  

Invariant Joint
Junction Tree

• A clique tree for a **triangulated** graph is referred to as a **junction tree**

• In junction trees, local consistency implies global consistency. Thus the local message-passing algorithms is (provably) correct.

• **Only** triangulated graphs have the property that their clique trees are junction trees. Thus if we want local algorithms, we **must** triangulate
Sketch of the Junction Tree Algorithm

1. Moralize the graph (trivial)
2. Triangulate the graph (good heuristic exist, but actually NP hard)
3. Build a clique tree (e.g., using a maximum spanning tree algorithm)
4. Propagation of probabilities --- a local message-passing protocol

• Results in marginal probabilities of all cliques --- solves all queries in a single run
• A **generic** exact inference algorithm for any GM
• **Complexity**: exponential in the size of the maximal clique --- a good elimination order often leads to small maximal clique, and hence a good (i.e., thin) JT
MLE for GLIMs with natural response

• Log-likelihood

\[ \ell = \sum_n \log h(y_n) + \sum_n \left( \theta^T x_n y_n - A(\eta_n) \right) \]

\[ \frac{d\ell}{d\theta} = \sum_n \left( x_n y_n - \frac{dA(\eta_n)}{d\eta_n} \frac{d\eta_n}{d\theta} \right) \]

\[ = \sum_n (y_n - \mu_n) x_n \]

\[ = X^T (y - \mu) \]

This is a fixed point function because \( \mu \) is a function of \( \theta \)

• Derivative of log-likelihood

• Learning for canonical GLIMs
  • Stochastic gradient ascent = least mean squares (LMS)

\[ \theta^{t+1} = \theta^t + \rho (y_n - \mu_n^t) x_n \]

where \( \mu_n^t = (\theta^t)^T x_n \) and \( \rho \) is a step size
MLE for general BNs

• If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node

\[
\ell(\theta; D) = \log p(D | \theta) = \log \prod_n \left( \prod_i p(x_{n,i} | x_{n,\pi_i}, \theta_i) \right) = \sum_l \left( \sum_n \log p(x_{n,i} | x_{n,\pi_i}, \theta_i) \right)
\]

• MLE-based parameter estimation of GM reduces to local est. of each GLIM.
Example: multinomial model

- Data: We observed $N$ iid die rolls (K-sided): $D = \{5, 1, K, \ldots 3\}$

$$x_n = [x_{n,1}, x_{n,2}, \cdots, x_{n,K}] \text{ where } x_{n,k} = 0, 1 \sum_{k=1}^{K} x_{n,k} = 1$$

- Model: $X_{n,k} = 1$ with probability $\theta_k$ and $\sum_{k \in \{1, \cdots, K\}} \theta_k = 1$

- Likelihood of an observation:

$$P(x_i) = P(\{x_{n,k} = 1, \text{ where } k \text{ is the index of the n-th roll}\})$$

$$= \theta_k = \theta_{x_{n,1}} \theta_{x_{n,2}} \cdots \theta_{x_{n,k}} = \prod_{k=1}^{K} \theta_{x_{n,k}}$$

- Likelihood of $D$:

$$P(x_1, x_2, \ldots, x_N | \theta) = \prod_{n=1}^{N} P(x_n | \theta) = \prod_{k} \theta_{n,k}$$
MLE for general BNs

- If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

\[
\ell(\theta; D) = \log p(D | \theta) = \log \prod_n \left( \prod_i p(x_{n,i} | x_{n,\pi_i}, \theta_i) \right) = \sum_i \left( \sum_n \log p(x_{n,i} | x_{n,\pi_i}, \theta_i) \right)
\]

- MLE-based parameter estimation of GM reduces to local est. of each GLIM.
Decomposable likelihood of a BN

• Consider the GM:

\[ p(x \mid \theta) = p(x_1 \mid \theta_1)p(x_2 \mid x_1, \theta_2)p(x_3 \mid x_1, \theta_3)p(x_4 \mid x_2, x_3, \theta_4) \]

• This is the same as learning four separate smaller BNs each of which consists of a node and its parents.
MLE for BNs with tabular CPDs

• Each CPD is represented as a table (multinomial) with

\[ \theta_{ijk} \overset{\text{def}}{=} p(X_i = j \mid X_{\pi_i} = k) \]

• In case of multiple parents the CPD is a high-dimensional table

• The sufficient statistics are counts of variable configurations

\[ n_{ijk} \overset{\text{def}}{=} \sum_n x_{n,i}^j x_{n,\pi_i}^k \]

• The log-likelihood is

\[ \ell(\theta; D) = \log \prod_{i,j,k} \theta_{ijk}^{n_{ijk}} = \sum_{i,j,k} n_{ijk} \log \theta_{ijk} \]

• And using a Lagrange multiplier to enforce that conditionals sum up to 1 we have:

\[ \theta_{ijk}^{\text{ML}} = \frac{n_{ijk}}{\sum_{j'} n_{ij'k}} \]
What about parameter priors?

• In a BN we have a collection of local distributions,

\[ p(x_i^k | x_{\pi_i}^j) = \theta_{x_i^k | x_{\pi_i}^j} \]

• How can we define priors over the whole BN?

• We could write \( P(x_1, x_2, \ldots, x_N; G, \theta)P(\theta | \alpha) \)
  • Symbolically the same as before but \( \theta \) is defined over a vector of random variables that follow different distributions.
  • We need \( \theta \) to decompose to use local rules. Otherwise we cannot decompose the likelihood any more.

• We need certain rules on \( \theta \)
  • Complete Model Equivalence
  • \textbf{Global Parameter Independence}
  • \textbf{Local Parameter Independence}
  • Likelihood and Prior Modularity
Log likelihood for Undirected Graphical Models with tabular clique potentials

• Sufficient statistics: for an MRF (V, E) the number of times that a configuration \( \mathbf{x} \) is observed in a dataset \( D \) can be represented as follows.

\[
m(\mathbf{x}) = \sum_n \delta(\mathbf{x}, \mathbf{x}_n) \quad (\text{total count}), \quad \text{and} \quad m(\mathbf{x}_c) = \sum_{\mathbf{x}_{\mathcal{V}\setminus c}} m(\mathbf{x}) \quad (\text{clique count})
\]

• The log-likelihood is

\[
p(D|\theta) = \prod_n \prod_{\mathbf{x}} p(\mathbf{x} | \theta)^{\delta(\mathbf{x}, \mathbf{x}_n)}
\]

\[
\log p(D|\theta) = \sum_n \sum_{\mathbf{x}} \delta(\mathbf{x}, \mathbf{x}_n) \log p(\mathbf{x} | \theta) = \sum_{\mathbf{x}} \sum_n \delta(\mathbf{x}, \mathbf{x}_n) \log p(\mathbf{x} | \theta)
\]

\[
\ell = \sum_{\mathbf{x}} m(\mathbf{x}) \log \left( \frac{1}{Z} \prod_c \psi_c(\mathbf{x}_c) \right)
\]

\[
= \sum_c \sum_{\mathbf{x}_c} m(\mathbf{x}_c) \log \psi_c(\mathbf{x}_c) - N \log Z
\]
MLE for Undirected Graphical Models

• Case 1: The model is **decomposable** (triangulated graph) and all the clique potentials are defined on maximal cliques.
  • The MLE of clique potentials are equal to the empirical marginals (or conditionals) of the corresponding clique.
  • Solve MLE by inspection

• Decomposable models
  • G is decomposable, G is triangulated, G has a junction tree

\[ p(x) = \frac{\prod_c \psi_c(x_c)}{\prod_s \phi_s(x_s)} \]

\[
p_{MLE}(X_1, X_2, X_3) = \frac{\tilde{p}(X_1, X_2)\tilde{p}(X_2, X_3)}{\tilde{p}(X_2)}
\]

\[
p_{MLE}(X_1, X_2) = \sum_{X_3} \tilde{p}(X_1, X_2, X_3) = \tilde{p}(X_1|X_2) \sum_{X_3} \tilde{p}(X_2, X_3) = \tilde{p}(X_1, X_2)
\]

\[
p_{MLE}(X_2, X_3) = \tilde{p}(X_2, X_3)
\]

• Ex.: Chain X1 – X2 – X3

\[ p_{MLE}(X_1, X_2, X_3) = \frac{\tilde{p}(X_1, X_2)\tilde{p}(X_2, X_3)}{\tilde{p}(X_2)} \]
MLE for Undirected Graphical Models

• Case 2: The model is **non-decomposable**, the potentials are defined as non-maximal cliques. We cannot equate MLE of clique potentials to empirical marginals (or conditionals)
  • Iterative potential fitting
  • Generalized Iterative Scaling
Iterative Proportional Fitting (IPF)

- From the log-likelihood:
  \[
  \frac{\partial \ell}{\partial \psi_c(x_c)} = \frac{m(x_c)}{\psi_c(x_c)} - N \frac{p(x_c)}{\psi_c(x_c)}
  \]

- Let’s rewrite in a different way:
  \[
  \frac{m(x_c)}{N \psi_c(x_c)} = \frac{p(x_c)}{\psi_c(x_c)} \quad \text{or} \quad \frac{\tilde{p}(x_c)}{\psi_c(x_c)} = \frac{p(x_c)}{\psi_c(x_c)}
  \]
  - The clique potentials implicitly appear in the model marginal: \( p(x_c) = f(\psi_c(x_c)) \)

- Let’s forget a closed form solution and focus on a fixed-point iteration method
  \[
  \frac{\tilde{p}(x_c)}{\psi_c^{(t+1)}(x_c)} = \frac{p(x_c)}{\psi_c^{(t)}(x_c)} \quad \Rightarrow \quad \psi_c^{(t+1)}(x_c) = \psi_c^{(t)}(x_c) \frac{\tilde{p}(x_c)}{p^{(t)}(x_c)}
  \]
  - Need to run inference for \( p^{(t)}(x_c) \)
Feature-based model

• Joint distribution:
  \[ p(x) = \frac{1}{Z(\theta)} \prod_c \psi_c(x_c) = \frac{1}{Z(\theta)} \exp \left\{ \sum_c \sum_{i \in I_c} \theta_k f_k(x_{c_i}) \right\} \]

• We can use the simplified form
  \[ p(x) = \frac{1}{Z(\theta)} \exp \left\{ \sum_i \theta_i f_i(x_{c_i}) \right\} \]

• The features correspond to the sufficient statistics of our model.

• We need to learn parameters \( \theta_k \)

• What about IPF?
  • Not clear how to use this rule to update the parameters and potentials
MLE of Feature-based Undirected Graphical Models

- We have

\[ \tilde{\ell} (\theta; D) \geq \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \mid \theta^{(t)}) \sum_i f_i(x) \exp (\Delta \theta_i^{(t)}) - \log Z (\theta^{(t)}) + 1 = \Lambda (\theta) \]

- We take the derivative

  - \( p^{(t)}(x) \) is the unnormalized version of \( p(x \mid \theta^{(t)}) \)

- Our updates are:

\[ \theta_i^{(t+1)} = \theta_i^{(t)} + \Delta \theta_i^{(t)} \Rightarrow p^{(t+1)}(x) = p^{(t)}(x) \prod_i e^{\Delta \theta_i^{(t)} f_i(x)} \]
Learning in Graphical Models

• Goal: Given a set of independent samples (assignments to random variables), find the best network

(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)
Information Theoretic Interpretation

\[
\ell(\theta_G, G; D) = \log \hat{p}(D | \theta_G, G) \\
= M \sum_i \left( \sum_{x_i, x_{\pi_i(G)}} \hat{p}(x_i, x_{\pi_i(G)}) \log \hat{p}(x_i | x_{\pi_i(G)}, \theta_{|\pi_i(G)}) \right) \\
= M \sum_i \left( \sum_{x_i, x_{\pi_i(G)}} \hat{p}(x_i, x_{\pi_i(G)}) \log \frac{\hat{p}(x_i, x_{\pi_i(G)}, \theta_{|\pi_i(G)})}{\hat{p}(x_{\pi_i(G)})} \hat{p}(x_i) \right) \\
= M \sum_i \left( \sum_{x_i, x_{\pi_i(G)}} \hat{p}(x_i, x_{\pi_i(G)}) \log \frac{\hat{p}(x_i, x_{\pi_i(G)}, \theta_{|\pi_i(G)})}{\hat{p}(x_{\pi_i(G)}) \hat{p}(x_i)} \right) - M \sum_i \left( \sum_{x_i} \hat{p}(x_i) \log \hat{p}(x_i) \right) \\
= M \sum_i \hat{I}(x_i, x_{\pi_i(G)}) - M \sum_i \hat{H}(x_i)
\]

- Larger dependence of $X_i$ on the parents defined by $G$ => higher score
- Adding edges always helps! $I(X; Y) \leq I(X; \{Y, Z\})$
- Max score when we have a fully connected graph. We are overfitting to our data!
Chow-Liu tree learning algorithm

• **Objective function:**

\[
\mathcal{L}(\theta_G, G; D) = \log \hat{p}(D | \theta_G, G) = M \sum_i \hat{I}(x_i, x_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \Rightarrow \]

\[
C(G) = M \sum_i \hat{I}(x_i, x_{\pi_i(G)})
\]

• **Chow-Liu:**
  • For each pair of variables \(x_i\) and \(x_j\)
    • Compute empirical distribution
      \[
      \hat{p}(X_i, X_j) = \frac{\text{count}(x_i, x_j)}{M}
      \]
  • Compute mutual information
    \[
    \hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)}
    \]
  • Define a graph with nodes \(x_1, \ldots, x_n\)
    • Edge \((i, j)\) gets weight \(\hat{I}(X_i, X_j)\)
Chow-Liu tree learning algorithm

• Objective function:

\[ \ell(\theta_G, G; D) = \log \hat{p}(D \mid \theta_G, G) = M \sum_i \hat{I}(x_i, x_{\pi_i(G)}) - M \sum_i \hat{H}(x_i) \Rightarrow C(G) = M \sum_i \hat{I}(x_i, x_{\pi_i(G)}) \]

• Chow-Liu:
  • Optimal tree BN
    • Compute maximum weight spanning tree
    • Direction in BN: pick any node as root, do BFS to define directions

\[ C(G) = I(A, B) + I(A, C) + I(C, D) + I(C, E) \]
Gaussian Graphical Models

• Multivariate Gaussian density:

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

• Let \( \mu = 0 \) \( Q = \Sigma^{-1} \)

\[
p(x_1, x_2, \ldots, x_p | \mu = 0, Q) = \frac{|Q|^{1/2}}{(2\pi)^{n/2}} \exp\left\{ -\frac{1}{2} \sum_i q_{ii}(x_i)^2 - \sum_{i<j} q_{ij}x_ix_j \right\}
\]

• We can view this as a continuous MRF with potentials defined on every node and edge

\[
P(x_d | \Theta) = \exp\left( \sum_{i \in V} \frac{t_i}{i} x_{d,i} + \sum_{(i,j) \in E} \theta_{ij} x_{d,i} x_{d,j} - A(\Theta) \right)
\]
The covariance and the precision matrices

- Covariance matrix \( \Sigma \)
  \[
  \Sigma_{i,j} = 0 \ \Rightarrow \ \ X_i \perp X_j \text{ or } p(X_i, X_j) = p(X_i)p(X_j)
  \]
  - What is the graphical model interpretation?

- Precision matrix \( \mathcal{Q} = \Sigma^{-1} \)
  \[
  Q_{i,j} = 0 \ \Rightarrow \ \ X_i \perp X_j|\mathbf{X}_{-ij} \text{ or } p(X_i, X_j|\mathbf{X}_{-ij}) = p(X_i|\mathbf{X}_{-ij})p(X_j|\mathbf{X}_{-ij})
  \]
  - What is the graphical model interpretation?
The covariance and the precision matrices

• Covariance matrix $\Sigma$

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

• What is the GM interpretation? Marginally independent/Correlation graph

• Precision matrix $Q = \Sigma^{-1}$

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

• What is the GM interpretation? Conditionally independent/Markov graph

• Why important?
Recall Lasso

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

where \( l(\theta_i) = \log P(y_i | x_i, \theta_i) \).

\[ P(x[1], ..., x[p]) \sim N(0, Q) \]

\[ P(x[i] | x[-i]) \sim N(q^*x[-i]/c, \sigma) \]

Given this we can estimate Q column-by-column using a slightly different estimator. We estimate p vectors.

How can we estimate \( N(q^*x[-i]/c, \sigma) \)? Linear regression corresponds to MLE of this model.
Gaussian Mixture Models (GMMs)

• Consider a mixture of $K$ Gaussian components:
  
  • $Z$ is a latent class indicator vector
  
  \[
p(z_n) = \text{multi}(z_n : \pi) = \prod_k (\pi_k)^{z_{nk}}
\]

  • $X$ is a conditional Gaussian variable with a class-specific mean/covariance
  
  \[
p(x_n \mid z_n = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2}|\Sigma_k|^{1/2}} \exp\left\{ -\frac{1}{2} (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) \right\}
\]

• Likelihood

\[
p(x_n \mid \mu, \Sigma) = \sum_k p(z_n = 1 \mid \pi) p(x_n \mid z_n = 1, \mu, \Sigma) = \sum_{z_n} \prod_k (\pi_k)^{z_{nk}} N(x_n \mid \mu_k, \Sigma_k)^{z_{nk}} = \sum_k \pi_k N(x_n \mid \mu_k, \Sigma_k)
\]
Why is learning with latent variables harder?

• In fully observed iid settings, the log likelihood decomposes into a sum of local terms (at least for Bayes Nets)

\[
\ell_c(\theta; D) = \log p(x, z | \theta) = \log p(z | \theta_z) + \log p(x | z, \theta_x)
\]

• With latent variables, all parameters become coupled via marginalization
Gaussian Mixture Models (GMMs)

• Consider a mixture of K Gaussian components:

  \[
  p(x_n|\mu, \Sigma) = \sum_k p(z^k = 1|\pi)p(x_n|z^k = 1, \mu, \Sigma)
  = \sum_{z_n} \prod_k \left(\pi_k(z_n)^x \cdot N(x_n: \mu_k, \Sigma_k)^z \right) = \sum_k \pi_k N(x_n|\mu_k, \Sigma_k)
  \]

• The expected complete log-likelihood

  \[
  \langle \ell_c(\Theta; x, z) \rangle = \sum_n \langle \log p(z_n|\pi) \rangle_{p(z|x)} + \sum_n \langle \log p(x_n|z_n, \mu, \Sigma) \rangle_{p(z|x)}
  = \sum_n \sum_k \langle z_n^k \rangle \log \pi_k - \frac{1}{2} \sum_n \sum_k \langle z_n^k \rangle \left( (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k) + \log |\Sigma_k| + C \right)
  \]
E-step

• We maximize $<l_c(\theta)>$ iteratively using the following iterative procedure:
  
  • Expectation step: Compute the expected value of the sufficient statistics of the hidden variables (i.e., $z$) given current estimate of the parameters (i.e., $\pi$ and $\mu$)

  \[
  \tau_n^{k(t)} = \left< z_n^k \right>_{q(t)} = p(z_n^k = 1 | x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi^{(t)}_k N(x_n, | \mu^{(t)}_k, \Sigma^{(t)}_k)}{\sum_i \pi^{(t)}_i N(x_n, | \mu^{(t)}_i, \Sigma^{(t)}_i)}
  \]

  • We are doing inference
M-step

• We maximize $\langle l_c(\theta) \rangle$ iteratively using the following iterative procedure:
  
  • Maximization step: Compute the parameters under the current results of the expected value of the hidden variables

\[
\pi_k^* = \arg \max \langle l_c(\theta) \rangle, \quad \Rightarrow \quad \frac{\partial}{\partial \pi_i} \langle l_c(\theta) \rangle = 0, \forall k, \quad \text{s.t.} \sum_k \pi_k = 1 \\
\Rightarrow \quad \pi_k^* = \frac{\sum_n \langle z_n \rangle}{\sum_n \langle n_k \rangle} = \frac{\sum_n \tau_n^{(t)}}{\sum_n \tau_n^{(t)}} = \langle n_k \rangle/\langle N \rangle
\]

\[
\mu_k^* = \arg \max \langle l(\theta) \rangle, \quad \Rightarrow \quad \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{(t)} x_n}{\sum_n \tau_n^{(t)}}
\]

\[
\Sigma_k^* = \arg \max \langle l(\theta) \rangle, \quad \Rightarrow \quad \Sigma_k^{(t+1)} = \frac{\sum_n \tau_n^{(t)}(x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T}{\sum_n \tau_n^{(t)}}
\]

• Isomorphic to MLE except that hidden variables are replaced by their expectations

Fact:
\[
\frac{\partial \log |A^{-1}|}{\partial A^{-1}} = A^T \\
\frac{\partial x^T A x}{\partial A} = x x^T
\]
The EM algorithm for mixtures of Gaussians is like a soft-version of the K-means algorithm:

**K-means**
- In the K-means “E-step” we do hard assignment:
  \[ z_n^{(t)} = \arg \max_k (x_n - \mu_k^{(t)} )^T \Sigma_k^{-1}(x_n - \mu_k^{(t)}) \]
- In the K-means “M-step” we update the means as the weighted sum of the data, but now the weights are 0 or 1:
  \[ \mu_k^{(t+1)} = \frac{\sum_n \delta(z_n^{(t)}, k) x_n}{\sum_n \delta(z_n^{(t)}, k)} \]

**EM**
- E-step
  \[ \tau_n^{k(t)} = \langle z_n^k \rangle_{q(t)} \]
  \[ = p(z_n^k = 1 | x, \mu, \Sigma) = \frac{\pi_k^{(t)} N(x_n | \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n | \mu_i^{(t)}, \Sigma_i^{(t)})} \]
- M-step
  \[ \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}} \]
K-Means vs EM

• The EM algorithm for mixtures of Gaussians is like a soft-version of the K-means algorithm

  ● **K-means**
    - In the K-means “E-step” we do hard assignment:
      \[ z_n^{(t)} = \text{arg max}_k (x_n - \mu_k^{(t)})^T \Sigma_k^{-1}(x_n - \mu_k^{(t)}) \]
    - In the K-means “M-step” we update the means as the weighted sum of the data, but now the weights are 0 or 1:
      \[ \mu_k^{(t+1)} = \frac{\sum_n \delta (z_n^{(t)}, k) x_n}{\sum_n \delta (z_n^{(t)}, k)} \]

  ● **EM**
    - **E-step**
      \[ \tau_n^{k(t)} = \langle z_n^k \rangle_q^{(t)} \]
      \[ = p(z_n^k = 1 | x, \mu^{(t)}, \Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n | \mu_k^{(t)}, \Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n | \mu_i^{(t)}, \Sigma_i^{(t)})} \]
    - **M-step**
      \[ \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}} \]
Variational Principle

• The dual function takes the form

\[ A^*(\mu) = \begin{cases} -H(p_{\theta(\mu)}) & \text{if } \mu \in \mathcal{M}^\circ \\ +\infty & \text{if } \mu \notin \overline{\mathcal{M}}. \end{cases} \]

• The log partition function has the variational form

\[ A(\theta) = \sup_{\mu \in \mathcal{M}} \{ \theta^T \mu - A^*(\mu) \} \]

• For all \( \theta \) the above optimization problem is attained uniquely at \( \mu(\theta) \) that satisfies

\[ \mu(\theta) = \mathbb{E}_\theta [\phi(X)] \]
Marginal Polytope

• For any distribution $p(x)$ and a set of sufficient statistics $\phi(x)$ define a vector of mean parameters

$$\mu_i = \mathbb{E}_p[\phi_i(X)] = \int \phi_i(x)p(x)\,dx$$

• $p(x)$ is not necessarily an exponential family

• The set of all realizable mean parameters is a convex set

$$\mathcal{M} := \{\mu \in \mathbb{R}^d \mid \exists \ p \text{ s.t. } \mathbb{E}_p[\phi(X)] = \mu\}.$$  

• For discrete exp. families this is called marginal polytope.
Convex Polytope

• Convex hull representation

\[ \mathcal{M} = \left\{ \mu \in \mathbb{R}^d \mid \sum_{x \in X^m} \phi(x) p(x) = \mu, \text{ for some } p(x) \geq 0, \sum_{x \in X^m} p(x) = 1 \right\} \]

\[ \triangleq \text{conv} \{ \phi(x), x \in X^m \} \]

• Half-plane representation
  • Minkowski-Weyl Theorem: any non-empty convex polytope can be characterized by a finite collection of linear inequality constraints

\[ \mathcal{M} = \left\{ \mu \in \mathbb{R}^d \mid a_j^T \mu \geq b_j, \forall j \in \mathcal{J} \right\}, \]

where \(|\mathcal{J}|\) is finite.
Example: Two-node Ising Model

- The distribution
  - Sufficient statistics

\[ p(x; \theta) \propto \exp\{\theta_1 x_1 + \theta_2 x_2 + \theta_{12} x_{12}\} \]

\[ \phi(x) = \{x_1, x_2, x_{12}\} \]

- The marginal polytope is characterized by

\[
\begin{align*}
\mu_1 & \geq \mu_{12} \\
\mu_2 & \geq \mu_{12} \\
\mu_{12} & \geq 0 \\
1 + \mu_{12} & \geq \mu_1 + \mu_2
\end{align*}
\]

- The dual has an explicit form

\[
A^*(\mu) = \mu_{12} \log \mu_{12} + (\mu_1 - \mu_{12}) \log(\mu_1 - \mu_{12}) + (\mu_2 - \mu_{12}) \log(\mu_2 - \mu_{12})
\]

\[
+ (1 + \mu_{12} - \mu_1 - \mu_2) \log(1 + \mu_{12} - \mu_1 - \mu_2)
\]

- The variational problem is

\[
A(\theta) = \max_{\{\mu_1, \mu_2, \mu_{12}\} \in \mathcal{M}} \{\theta_1 \mu_1 + \theta_2 \mu_2 + \theta_{12} \mu_{12} - A^*(\mu)\}
\]

- The optimum is attained at

\[
\mu_1(\theta) = \frac{\exp\{\theta_1\} + \exp\{\theta_1 + \theta_2 + \theta_{12}\}}{1 + \exp\{\theta_1\} + \exp\{\theta_2\} + \exp\{\theta_1 + \theta_2 + \theta_{12}\}}
\]
Tree Graphical Models

• Discrete variables \( X_s \in \{0, 1, \ldots, m_s - 1\} \) on a tree \( T = (V,E) \)

\[
\begin{align*}
\mathbb{I}_j(x_s) & \quad \text{for } s = 1, \ldots n, \quad j \in \mathcal{X}_s \\
\mathbb{I}_{jk}(x_s, x_t) & \quad \text{for } (s,t) \in E, \quad (j,k) \in \mathcal{X}_s \times \mathcal{X}_t
\end{align*}
\]

• Sufficient statistics

• Exponential representation of distribution?

• Mean parameters are marginal probabilities:

\[
\begin{align*}
\mu_{s;j} &= \mathbb{E}_{p}[\mathbb{I}_j(X_s)] = \mathbb{P}[X_s = j] \quad \forall j \in \mathcal{X}_s, \quad \mu_s(x_s) = \sum_{j \in \mathcal{X}_s} \mu_{s;j} \mathbb{I}_j(x_s) = \mathbb{P}(X_s = x_s) \\
\mu_{st;jk} &= \mathbb{E}_{p}[\mathbb{I}_{st;jk}(X_s,X_t)] = \mathbb{P}[X_s = j, X_t = k] \quad \forall (j,k) \in \mathcal{X}_s \times \mathcal{X}_t. \\
\mu_{st}(x_s, x_t) &= \sum_{(j,k) \in \mathcal{X}_s \times \mathcal{X}_t} \mu_{st;jk} \mathbb{I}_{jk}(x_s, x_t) = \mathbb{P}(X_s = x_s, X_t = x_t)
\end{align*}
\]
BP on Arbitrary Graphs

• Two main difficulties of the variation formulation

\[ A(\theta) = \sup_{\mu \in \mathcal{M}} \{ \theta^T \mu - A^*(\mu) \} \]

• The marginal polytope is hard to characterize, so let’s use the tree-based outer bound

\[ \mathbb{L}(G) = \left\{ \tau \geq 0 \mid \sum_{x_s} \tau_s(x_s) = 1, \sum_{x_t} \tau_{st}(x_s, x_t) = \tau_s(x_s) \right\} \]

• Exact entropy lacks explicit form, so let’s approximate it using the exact expression for trees

\[ -A^*(\tau) \approx H_{\text{Bethe}}(\tau) := \sum_{s \in V} H_s(\tau_s) - \sum_{(s,t) \in E} I_{st}(\tau_{st}). \]
Bethe Variational Problem

• Combining the two gives us the Bethe variational problem

\[ \max_{\tau \in \mathcal{L}(G)} \left\{ \langle \theta, \tau \rangle + \sum_{s \in V} H_s(\tau_s) - \sum_{(s,t) \in E} I_{st}(\tau_{st}) \right\}. \]

• What is happening?
• Tree-based outer bound

\[ \mathcal{M}(G) \subseteq \mathcal{L}(G) \]
Mean Field Methods

• For a given tractable subgraph $F$, a subset of canonical parameters is

\[ \mathcal{M}(F; \phi) := \{ \tau \in \mathbb{R}^d \ | \ \tau = E_\theta[\phi(X)] \ \text{for some} \ \theta \in \Omega(F) \} \]

• Inner approximation

\[ \mathcal{M}(F; \phi)^o \subseteq \mathcal{M}(G; \phi)^o \]

• Mean field solves the relaxed problem

\[ \max_{\tau \in \mathcal{M}_F(G)} \left\{ \langle \tau, \theta \rangle - A_F^*(\tau) \right\} \]

• \( A_F^* = A^*|_{\mathcal{M}_F(G)} \) is the exact dual function restricted to \( \mathcal{M}_F(G) \)
Example: Naïve Mean Field for Ising Model

• Ising model in \{0,1\} representation

\[
p(x) \propto \exp \left\{ \sum_{s \in V} x_s \theta_s + \sum_{(s,t) \in E} x_s x_t \theta_{st} \right\}
\]

• Mean parameters

\[
\mu_s = E_p[X_s] = P[X_s = 1] \quad \mu_{st} = E_p[X_s X_t] = P[(X_s, X_t) = (1,1)]
\]

• For fully disconnected graph F

\[
\mathcal{M}_F(G) := \{ \tau \in \mathbb{R}^{|V|+|E|} \mid 0 \leq \tau_s \leq 1, \forall s \in V, \tau_{st} = \tau_s \tau_t, \forall (s,t) \in E \}
\]

• The dual decomposes into sum, one for each node

\[
A^*_F(\tau) = \sum_{s \in V} [\tau_s \log \tau_s + (1 - \tau_s) \log(1 - \tau_s)]
\]
Example: Naïve Mean Field for Ising Model

• Mean field problem

\[
A(\theta) \geq \max_{(\tau_1, \ldots, \tau_m) \in [0,1]^m} \left\{ \sum_{s \in V} \theta_s \tau_s + \sum_{(s,t) \in E} \theta_{st} \tau_s \tau_t - A^*_F(\tau) \right\}
\]

• The same objective function as in free energy based approach

• The naïve mean field update equations

\[
\tau_s \leftarrow \sigma \left( \theta_s + \sum_{t \in N(s)} \theta_s \tau_t \right)
\]

• Lower bound on log partition function
Geometry of Mean Field

• Mean field optimization is always non-convex for any exponential family in which the state space is finite

\[ \mathcal{M}(G) = \text{conv}\{\phi(e); e \in \mathcal{X}^m\} \]

• Marginal polytope is a convex hull

\[ \mathcal{M}_F(G) \] contains all the extreme points (if it is a strict subset then it must be non-convex

• Example: two-node Ising

\[ \mathcal{M}_F(G) = \{0 \leq \tau_1 \leq 1, 0 \leq \tau_2 \leq 1, \tau_{12} = \tau_1 \tau_2\} \]

• Parabolic cross section along \( \tau_1 = \tau_2 \)
Monte Carlo Methods

• Draw random samples from the desired distribution
• Yield a stochastic representation of a complex distribution
  • marginals and other expectations can be approximated using sample-based averages
  • \( E_p(f(x)) = \frac{\sum f(X^{(m)})}{|m|} \)
• Asymptotically exact and easy to apply to arbitrary models
• Challenges:
  • how to draw samples from a given dist. (not all distributions can be trivially sampled)?
  • how to make better use of the samples (not all sample are useful, or equally useful, see an example later)?
  • how to know we've sampled enough?
Rejection sampling

• Suppose we wish to sample from dist. $\Pi(X)=\Pi'(X)/Z$.
  • $\Pi(X)$ is difficult to sample, but $\Pi'(X)$ is easy to evaluate
  • Sample from a simpler distribution $Q(X)$
  • Rejection sampling $x^* \sim Q(X)$, accept $x^*$ w.p. $\Pi'(x^*)/kQ(x^*)$

\[ p(x) = \frac{[\Pi'(x)/kQ(x)]Q(x)}{\int [\Pi'(x)/kQ(x)]Q(x)dx} \]

\[ = \frac{\Pi'(x)}{\int \Pi'(x)dx} = \Pi(x) \]

• Correctness:

• Pitfall: We gained a sample but what did we pay?
Markov Chain Monte Carlo

- MCMC algorithms feature adaptive proposals
  - Instead of $Q(x')$ use $Q(x'|x)$ where $x'$ is the new state being sampled and $x$ is the previous sample
  - As $x$ changes $Q(x'|x)$ can also change
Metropolis-Hastings

1. Initialize starting state \( x^{(0)} \), set \( t = 0 \)
2. **Burn-in:** while samples have “not converged”
   - \( x = x^{(t)} \)
   - \( t = t + 1 \),
   - sample \( x^* \sim Q(x^* | x) \) \quad // \text{draw from proposal}
   - sample \( u \sim \text{Uniform}(0,1) \) \quad // \text{draw acceptance threshold}
     - if \( u < A(x^* | x) = \min \left( 1, \frac{P(x^*)Q(x | x^*)}{P(x)Q(x^* | x)} \right) \)
       - \( x^{(t)} = x^* \) \quad // \text{transition}
     - else
       - \( x^{(t)} = x \) \quad // \text{stay in current state}
   - Take samples from \( P(x) = \): Reset \( t=0 \), for \( t = 1:N \)
     - \( x(t+1) \leftarrow \text{Draw sample (} x(t) \text{)} \)
Gibbs Sampling Example

**Variables**

1. If we set $v_1$ to True, we are rewarded by 5 points!
   \[ f_1(a) = \begin{cases} 5, & a = \text{True} \\ 0, & \text{otherwise} \end{cases} \]

2. If we set $v_2$ and $v_3$ to the same, we get 10 more points!
   \[ f_2(a,b) = \begin{cases} 10, & a = b \\ 0, & \text{otherwise} \end{cases} \]

**Factors**

A “Possible World”

---

**Gibbs Sampling: A Descriptive Tutorial**

1. Initialize variables with a random assignment. 
   - $T$ 
   - $F$

2. For each random variable:
   2.1 Calculate the points we earn for each assignment:
   - e.g., $v_2 = T$: 0 points
   - $v_2 = F$: 10 points

   2.2 Randomly pick one assignment:
   - e.g., $P(v_2 = T) = \exp(0) / (\exp(0) + \exp(10))$
   - $P(v_2 = F) = \exp(10) / (\exp(0) + \exp(10))$

3. Generate one sample. Goto 2 if we want more samples.

---

82