

Inference and Representation

DS-GA-1005, CSCI-GA.2569

David Sontag

New York University

Lecture 2, Sept. 19, 2016

Today's lecture

- Markov random fields
 - ① Formalism
 - ② Conditional independence properties
 - ③ Examples
- Factor graphs
- Bayesian networks \Rightarrow Markov random fields (*moralization*)
- Learning graphical models from data

Bayesian networks

Reminder of first lecture

- A **Bayesian network** is specified by a directed *acyclic* graph $G = (V, E)$ with:
 - 1 One node $i \in V$ for each random variable X_i
 - 2 One conditional probability distribution (CPD) per node, $p(x_i \mid \mathbf{x}_{\text{Pa}(i)})$, specifying the variable's probability conditioned on its parents' values
- Corresponds 1-1 with a particular factorization of the joint distribution:

$$p(x_1, \dots, x_n) = \prod_{i \in V} p(x_i \mid \mathbf{x}_{\text{Pa}(i)})$$

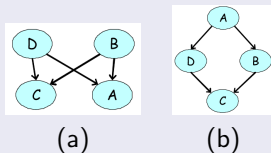
- Powerful framework for designing *algorithms* to perform probability computations

Bayesian networks have limitations

- Recall that G is a **perfect map** for distribution p if $I(G) = I(p)$
- Theorem:** Not every distribution has a perfect map as a DAG

Proof.

(By counterexample.) There is a distribution on 4 variables where the only independencies are $A \perp C \mid \{B, D\}$ and $B \perp D \mid \{A, C\}$. This cannot be represented by any Bayesian network.



Both (a) and (b) encode $(A \perp C \mid B, D)$, but in both cases $(B \not\perp D \mid A, C)$. □

Example

- Let's come up with an example of a distribution p satisfying $A \perp C \mid \{B, D\}$ and $B \perp D \mid \{A, C\}$
- A =Alex's hair color (red, green, blue)
 B =Bob's hair color
 C =Catherine's hair color
 D =David's hair color
- Alex and Bob are friends, Bob and Catherine are friends, Catherine and David are friends, David and Alex are friends
- Friends *never* have the same hair color!

Undirected graphical models

- An alternative representation for joint distributions is as an **undirected graphical model**
- As in BNs, we have one node for each random variable
- Rather than CPDs, we specify (non-negative) **potential functions** over sets of variables associated with cliques C of the graph,

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c)$$

Z is the **partition function** and normalizes the distribution:

$$Z = \sum_{\hat{x}_1, \dots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

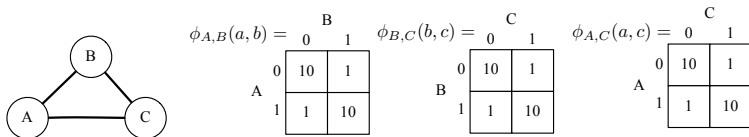
- Like CPD's, $\phi_c(\mathbf{x}_c)$ can be represented as a table, but it is *not normalized*
- Also known as **Markov random fields** (MRFs) or Markov networks

Undirected graphical models

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c),$$

$$Z = \sum_{\hat{x}_1, \dots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

Simple example (potential function on each edge encourages the variables to take the same value):



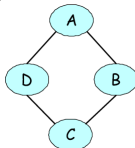
$$p(a, b, c) = \frac{1}{Z} \phi_{A,B}(a, b) \cdot \phi_{B,C}(b, c) \cdot \phi_{A,C}(a, c),$$

where

$$Z = \sum_{\hat{a}, \hat{b}, \hat{c} \in \{0,1\}^3} \phi_{A,B}(\hat{a}, \hat{b}) \cdot \phi_{B,C}(\hat{b}, \hat{c}) \cdot \phi_{A,C}(\hat{a}, \hat{c}) = 2 \cdot 1000 + 6 \cdot 10 = 2060.$$

Hair color example as a MRF

- We now have an **undirected** graph:



- The joint probability distribution is parameterized as

$$p(a, b, c, d) = \frac{1}{Z} \phi_{AB}(a, b) \phi_{BC}(b, c) \phi_{CD}(c, d) \phi_{AD}(a, d) \phi_A(a) \phi_B(b) \phi_C(c) \phi_D(d)$$

- **Pairwise potentials** enforce that no friend has the same hair color:

$$\phi_{AB}(a, b) = 0 \text{ if } a = b, \text{ and } 1 \text{ otherwise}$$

- **Single-node potentials** specify an affinity for a particular hair color, e.g.

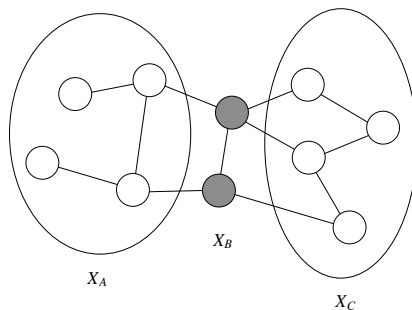
$$\phi_D(\text{"red"}) = 0.6, \quad \phi_D(\text{"blue"}) = 0.3, \quad \phi_D(\text{"green"}) = 0.1$$

The normalization Z makes the potentials **scale invariant**! Equivalent to

$$\phi_D(\text{"red"}) = 6, \quad \phi_D(\text{"blue"}) = 3, \quad \phi_D(\text{"green"}) = 1$$

Markov network structure implies conditional independencies

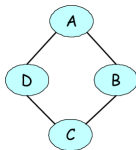
- Let G be the undirected graph where we have one edge for every pair of variables that appear together in a potential
- Conditional independence is given by **graph separation**!



- $X_A \perp X_C \mid X_B$ if there is no path from $a \in \mathbf{A}$ to $c \in \mathbf{C}$ after removing all variables in \mathbf{B}

Example

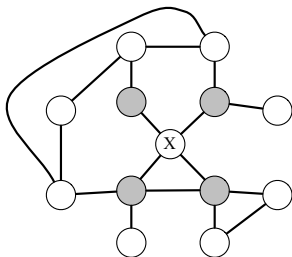
- Returning to hair color example, its undirected graphical model is:



- Since removing A and C leaves no path from D to B , we have $D \perp B \mid \{A, C\}$
- Similarly, since removing D and B leaves no path from A to C , we have $A \perp C \mid \{D, B\}$
- No other independencies implied by the graph

Markov blanket

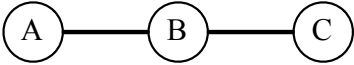
- A set \mathbf{U} is a **Markov blanket** of X if $X \notin \mathbf{U}$ and if \mathbf{U} is a minimal set of nodes such that $X \perp (\mathcal{X} - \{X\} - \mathbf{U}) \mid \mathbf{U}$
- In undirected graphical models, the Markov blanket of a variable is precisely its **neighbors** in the graph:



- In other words, X is independent of the rest of the nodes in the graph given its immediate neighbors

Proof of independence through separation

- We will show that $A \perp C \mid B$ for the following distribution:


$$p(a, b, c) = \frac{1}{Z} \phi_{AB}(a, b) \phi_{BC}(b, c)$$

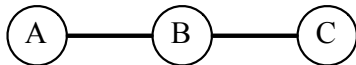
- First, we show that $p(a \mid b)$ can be computed using only $\phi_{AB}(a, b)$:

$$\begin{aligned} p(a \mid b) &= \frac{p(a, b)}{p(b)} \\ &= \frac{\frac{1}{Z} \sum_{\hat{c}} \phi_{AB}(a, b) \phi_{BC}(b, \hat{c})}{\frac{1}{Z} \sum_{\hat{a}, \hat{c}} \phi_{AB}(\hat{a}, b) \phi_{BC}(b, \hat{c})} \\ &= \frac{\phi_{AB}(a, b) \sum_{\hat{c}} \phi_{BC}(b, \hat{c})}{\sum_{\hat{a}} \phi_{AB}(\hat{a}, b) \sum_{\hat{c}} \phi_{BC}(b, \hat{c})} = \frac{\phi_{AB}(a, b)}{\sum_{\hat{a}} \phi_{AB}(\hat{a}, b)}. \end{aligned}$$

- More generally, the probability of a variable conditioned on its Markov blanket depends *only* on potentials involving that node

Proof of independence through separation

- We will show that $A \perp C \mid B$ for the following distribution:



$$p(a, b, c) = \frac{1}{Z} \phi_{AB}(a, b) \phi_{BC}(b, c)$$

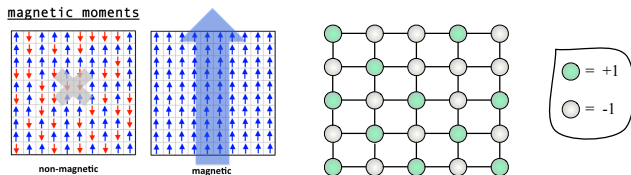
Proof.

$$\begin{aligned} p(a, c \mid b) &= \frac{p(a, c, b)}{\sum_{\hat{a}, \hat{c}} p(\hat{a}, b, \hat{c})} = \frac{\phi_{AB}(a, b) \phi_{BC}(b, c)}{\sum_{\hat{a}, \hat{c}} \phi_{AB}(\hat{a}, b) \phi_{BC}(b, \hat{c})} \\ &= \frac{\phi_{AB}(a, b) \phi_{BC}(b, c)}{\sum_{\hat{a}} \phi_{AB}(\hat{a}, b) \sum_{\hat{c}} \phi_{BC}(b, \hat{c})} \\ &= p(a \mid b) p(c \mid b) \end{aligned}$$



Example: Ising model

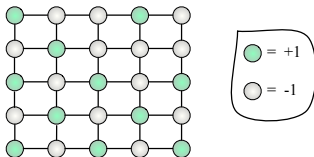
- Invented by the physicist Wilhelm Lenz (1920), who gave it as a problem to his student Ernst Ising
- Mathematical model of ferromagnetism in statistical mechanics
- The spin of an atom is biased by the spins of atoms nearby on the material:



- Each atom $X_i \in \{-1, +1\}$, whose value is the direction of the atom spin
- If a spin at position i is $+1$, what is the probability that the spin at position j is also $+1$?
- Are there phase transitions where spins go from “disorder” to “order”?

Example: Ising model

- Each atom $X_i \in \{-1, +1\}$, whose value is the direction of the atom spin
- The spin of an atom is biased by the spins of atoms nearby on the material:



$$p(x_1, \dots, x_n) = \frac{1}{Z} \exp \left(\sum_{i < j} w_{i,j} x_i x_j - \sum_i u_i x_i \right)$$

- When $w_{i,j} > 0$, nearby atoms encouraged to have the same spin (called **ferromagnetic**), whereas $w_{i,j} < 0$ encourages $X_i \neq X_j$
- Node potentials $\exp(-u_i x_i)$ encode the bias of the individual atoms
- Scaling the parameters makes the distribution more or less spiky

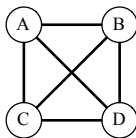
Higher-order potentials

- The examples so far have all been **pairwise MRFs**, involving only node potentials $\phi_i(X_i)$ and pairwise potentials $\phi_{i,j}(X_i, X_j)$
- Often we need **higher-order** potentials, e.g.

$$\phi(x, y, z) = 1[x + y + z \geq 1],$$

where X, Y, Z are binary, enforcing that at least one of the variables takes the value 1

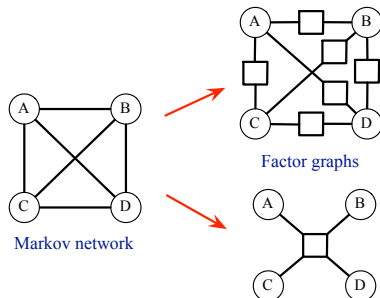
- Although Markov networks are useful for understanding independencies, they hide much of the distribution's structure:



Does this have pairwise potentials, or one potential for all 4 variables?

Factor graphs

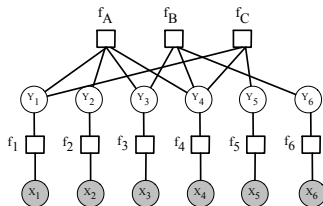
- G does not reveal the structure of the distribution: maximum cliques vs. subsets of them
- A **factor graph** is a bipartite undirected graph with variable nodes and factor nodes. Edges are only between the variable nodes and the factor nodes
- Each factor node is associated with a single potential, whose scope is the set of variables that are neighbors in the factor graph



- The distribution is same as the MRF – this is just a different data structure

Example: Low-density parity-check codes

- Error correcting codes for transmitting a message over a noisy channel (invented by Gallager in the 1960's, then re-discovered in 1996)



- Each of the top row factors enforces that its variables have even parity:

$$f_A(Y_1, Y_2, Y_3, Y_4) = 1 \text{ if } Y_1 \otimes Y_2 \otimes Y_3 \otimes Y_4 = 0, \text{ and } 0 \text{ otherwise}$$

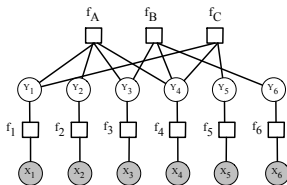
- Thus, the only assignments \mathbf{Y} with non-zero probability are the following (called **codewords**):

3 bits encoded using 6 bits

000000, 011001, 110010, 101011, 111100, 100101, 001110, 010111

- $f_i(Y_i, X_i) = p(X_i | Y_i)$, the likelihood of a bit flip according to noise model

Example: Low-density parity-check codes



- The *decoding* problem for LDPCs is to find

$$\operatorname{argmax}_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x})$$

This is called the **maximum a posteriori** (MAP) assignment

- Since Z and $p(\mathbf{x})$ are constants with respect to the choice of \mathbf{y} , can equivalently solve (taking the log of $p(\mathbf{y}, \mathbf{x})$):

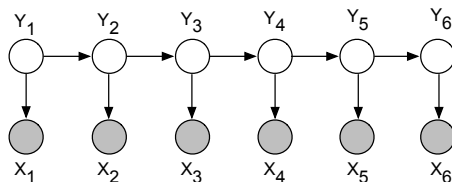
$$\operatorname{argmax}_{\mathbf{y}} \sum_{c \in C} \theta_c(\mathbf{y}_c, \mathbf{x}_c),$$

where $\theta_c(\mathbf{x}_c) = \log \phi_c(\mathbf{y}_c, \mathbf{x}_c)$

- This is a discrete optimization problem!

Converting BNs to Markov networks

What is the equivalent Markov network for a hidden Markov model?



Many inference algorithms are more conveniently given for undirected models – this shows how they can be applied to Bayesian networks

Moralization of Bayesian networks

- Procedure for converting a Bayesian network into a Markov network
- The **moral graph** $\mathcal{M}[G]$ of a BN $G = (V, E)$ is an undirected graph over V that contains an undirected edge between X_i and X_j if
 - 1 there is a directed edge between them (in either direction)
 - 2 X_i and X_j are both parents of the same node



(term historically arose from the idea of “marrying the parents” of the node)

- The addition of the moralizing edges leads to the loss of some independence information, e.g., $A \rightarrow C \leftarrow B$, where $A \perp B$ is lost

Converting BNs to Markov networks

- 1 Moralize the directed graph to obtain the undirected graphical model:



- 2 Introduce one potential function for each CPD:

$$\phi_i(x_i, \mathbf{x}_{pa(i)}) = p(x_i \mid \mathbf{x}_{pa(i)})$$

- So, converting a hidden Markov model to a Markov network is simple:



- For variables having > 1 parent, factor graph notation is useful

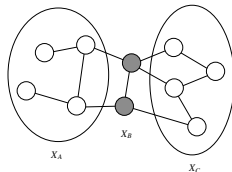
Factorization implies conditional independencies

- $p(\mathbf{x})$ is a *Gibbs distribution* over G if it can be written as

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c),$$

where the variables in each potential $c \in C$ form a clique in G

- Recall that conditional independence is given by graph separation:



- Theorem (**soundness of separation**): If $p(\mathbf{x})$ is a Gibbs distribution for G , then G is an I-map for $p(\mathbf{x})$, i.e. $I(G) \subseteq I(p)$

Proof: Suppose \mathbf{B} separates \mathbf{A} from \mathbf{C} . Then we can write

$$p(\mathbf{X}_A, \mathbf{X}_B, \mathbf{X}_C) = \frac{1}{Z} f(\mathbf{X}_A, \mathbf{X}_B) g(\mathbf{X}_B, \mathbf{X}_C).$$

Conditional independencies implies factorization

- Theorem (**soundness of separation**): If $p(\mathbf{x})$ is a Gibbs distribution for G , then G is an I-map for $p(\mathbf{x})$, i.e. $I(G) \subseteq I(p)$
- What about the converse? We need one more assumption:
- A distribution is **positive** if $p(\mathbf{x}) > 0$ for all \mathbf{x}
- Theorem (**Hammersley-Clifford**, 1971): If $p(\mathbf{x})$ is a positive distribution and G is an I-map for $p(\mathbf{x})$, then $p(\mathbf{x})$ is a Gibbs distribution that factorizes over G
- Proof is in Koller & Friedman book (as is counter-example for when $p(\mathbf{x})$ is not positive)
- This is important for **learning**:
 - Prior knowledge is often in the form of conditional independencies (i.e., a graph structure G)
 - Hammersley-Clifford tells us that it suffices to search over Gibbs distributions for G – allows us to *parameterize* the distribution

Today's lecture

- Markov random fields
 - ① Formalism
 - ② Conditional independence properties
 - ③ Examples
- Factor graphs
- Bayesian networks \Rightarrow Markov random fields (*moralization*)
- Learning graphical models from data

- Let's assume that the domain is governed by some underlying distribution p^* , which is induced by some network model $\mathcal{M}^* = (\mathcal{G}^*, \theta^*)$
- We are given a dataset \mathcal{D} of M samples from p^*
- The standard assumption is that the data instances are **independent and identically distributed (IID)**
- We are also given a family of models \mathcal{M} , and our task is to learn some model $\hat{\mathcal{M}} \in \mathcal{M}$ (i.e., in this family) that defines a distribution $p_{\hat{\mathcal{M}}}$
- We can learn model parameters for a fixed structure, or both the structure and model parameters

What should our goal be?

- Return a model $\hat{\mathcal{M}}$ that precisely captures the distribution p^* from which our data was sampled
- This is in general not achievable because of
 - computational reasons
 - limited data only provides a rough approximation of the true underlying distribution
- We need to select $\hat{\mathcal{M}}$ to construct the "best" approximation to \mathcal{M}^*
- What is "best"?

What is “best”?

This depends on what we want to do

- 1 Density estimation: we are interested in the full distribution (so later we can compute whatever conditional probabilities we want)
- 2 Specific prediction tasks: we are using the distribution to make a prediction
We discuss this in depth in lectures 9 and 10.
- 3 Structure or knowledge discovery: we are interested in the model itself (often of interest in data science)

Learning for density estimation

- We want to learn the full distribution so that later we can answer *any* probabilistic inference query
- In this setting we can view the learning problem as **density estimation**
- We want to construct $\hat{\mathcal{M}}$ as “close” as possible to p^*
- How do we evaluate “closeness”?
- **KL-divergence** is one possibility:

$$\mathbf{D}(p^* || p_\theta) = \mathbf{E}_{\mathbf{x} \sim p^*} \left[\log \left(\frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} \right) \right]$$

- $\mathbf{D}(P^* || \hat{P}) = 0$ iff the two distributions are the same.

Expected log-likelihood

- We can simplify this somewhat:

$$\mathbf{D}(p^* || p_\theta) = \mathbf{E}_{\mathbf{x} \sim p^*} \left[\log \left(\frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} \right) \right] = -\mathbf{H}(p^*) - \mathbf{E}_{\mathbf{x} \sim p^*} [\log p_\theta(\mathbf{x})]$$

- The first term does not depend on θ .
- Then, *minimizing* the KL-divergence from the true distribution is equivalent to *maximizing* the **expected log-likelihood**:

$$\mathbf{E}_{\mathbf{x} \sim p^*} [\log p_\theta(\mathbf{x})]$$

- Asks that p_θ assign high probability to instances sampled from p^* , so as to reflect the true distribution
 - Because of log, samples \mathbf{x} where $p_\theta(\mathbf{x}) \approx 0$ weigh heavily in objective
- **Problem:** In general we do not know p^* .

- Approximate the expected log-likelihood

$$\mathbf{E}_{\mathbf{x} \sim p^*} [\log p_{\theta}(\mathbf{x})]$$

with the *empirical log-likelihood*:

$$\mathbf{E}_{\mathcal{D}} [\log p_{\theta}(\mathbf{x})] = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \log p_{\theta}(\mathbf{x})$$

- Maximum likelihood learning is then:

$$\max_{\theta} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \log p_{\theta}(\mathbf{x})$$

How to avoid overfitting?

- Hard constraints, e.g. by selecting a less expressive hypothesis class:
 - Bayesian networks with at most d parents
 - Bayesian networks with parametric conditional probability distributions (e.g. logistic, noisy-or, neural networks)
 - Pairwise MRFs (instead of arbitrary higher-order potentials)
- Augment the learning objective function with **regularization**:

$$\text{objective}(\mathbf{x}, \mathcal{M}) = \text{loss}(\mathbf{x}, \mathcal{M}) + R(\mathcal{M})$$

(often equivalent to MAP estimation where we put a prior over parameters θ and maximize $\log p(\theta \mid \mathbf{x}) = \log p(\mathbf{x}; \theta) + \log p(\theta) - \text{constant}$)

ML estimation in Bayesian networks

- Suppose that we know the Bayesian network structure G
- Let $\theta_{x_i | \mathbf{x}_{pa(i)}}$ be the parameter giving the value of the CPD $p(x_i | \mathbf{x}_{pa(i)}; \theta)$
- Maximum likelihood estimation corresponds to solving:

$$\max_{\theta} \sum_{n=1}^N \log p(\mathbf{x}^n; \theta) = \max_{\theta} \ell(\theta; \mathcal{D})$$

subject to the non-negativity and normalization constraints

- This is equal to:

$$\begin{aligned} \max_{\theta} \sum_{n=1}^N \log p(\mathbf{x}^n; \theta) &= \max_{\theta} \sum_{n=1}^N \sum_{i=1}^{|V|} \log p(x_i^n | \mathbf{x}_{pa(i)}^n; \theta) \\ &= \max_{\theta} \sum_{i=1}^{|V|} \sum_{n=1}^N \log p(x_i^n | \mathbf{x}_{pa(i)}^n; \theta) \end{aligned}$$

- The optimization problem decomposes into an independent optimization problem for each CPD!

ML estimation in Bayesian networks

$$\begin{aligned}\ell(\theta; \mathcal{D}) = \log p(\mathcal{D}; \theta) &= \sum_{i=1}^{|V|} \sum_{n=1}^N \log p(x_i^n \mid \mathbf{x}_{pa(i)}^n; \theta) \\ &= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} \sum_{\substack{\hat{\mathbf{x}} \in \mathcal{D}: \\ \hat{x}_i, \hat{\mathbf{x}}_{pa(i)} = \mathbf{x}_i, \mathbf{x}_{pa(i)}}} \log p(x_i \mid \mathbf{x}_{pa(i)}; \theta) \\ &= \sum_{i=1}^{|V|} \sum_{\mathbf{x}_{pa(i)}} \sum_{\mathbf{x}_i} N_{\mathbf{x}_i, \mathbf{x}_{pa(i)}} \log \theta_{\mathbf{x}_i | \mathbf{x}_{pa(i)}},\end{aligned}$$

where $N_{\mathbf{x}_i, \mathbf{x}_{pa(i)}}$ is the number of times that the (partial) assignment $x_i, \mathbf{x}_{pa(i)}$ is observed in the training data

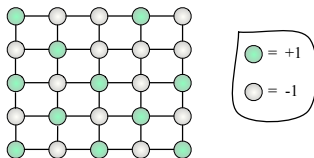
- We have the closed form ML solution:

$$\theta_{\mathbf{x}_i | \mathbf{x}_{pa(i)}}^{ML} = \frac{N_{\mathbf{x}_i, \mathbf{x}_{pa(i)}}}{\sum_{\hat{\mathbf{x}}_i} N_{\hat{\mathbf{x}}_i, \mathbf{x}_{pa(i)}}}$$

- We were able to estimate each CPD independently because the objective **decomposes** by variable and parent assignment

ML estimation in Markov networks

- How do we learn the parameters of an Ising model?



$$p(x_1, \dots, x_n) = \frac{1}{Z} \exp \left(\sum_{i < j} w_{i,j} x_i x_j - \sum_i u_i x_i \right)$$

Bad news for Markov networks

- The global normalization constant $Z(\theta)$ kills decomposability:

$$\begin{aligned}\theta^{ML} &= \arg \max_{\theta} \log \prod_{\mathbf{x} \in \mathcal{D}} p(\mathbf{x}; \theta) \\ &= \arg \max_{\theta} \sum_{\mathbf{x} \in \mathcal{D}} \left(\sum_c \log \phi_c(\mathbf{x}_c; \theta) - \log Z(\theta) \right) \\ &= \arg \max_{\theta} \left(\sum_{\mathbf{x} \in \mathcal{D}} \sum_c \log \phi_c(\mathbf{x}_c; \theta) \right) - |\mathcal{D}| \log Z(\theta)\end{aligned}$$

- The log-partition function prevents us from decomposing the objective into a sum over terms for each potential
- Solving for the parameters becomes much more complicated