All of Graphical Models

Xiaojin Zhu

Department of Computer Sciences
University of Wisconsin–Madison, USA

Tutorial at ICMLA 2011

The Whole Tutorial in One Slide

- Given GM = joint distribution $p(x_1, \ldots, x_n)$
- Do inference $= p(X_Q \mid X_E)$, in general $X_Q \cup X_E \subset \{x_1 \ldots x_n\}$
- If $p(x_1, \ldots, x_n)$ not given, estimate it from data

Outline

Outline

Life without Graphical Models

. . . is fine mathematically:

- The universe is reduced to a set of random variables $x_1, \ldots, x_n$
  - e.g., $x_1, \ldots, x_{n-1}$ can be the discrete or continuous features
  - e.g., $x_n \equiv y$ can be the discrete class label
- The joint $p(x_1, \ldots, x_n)$ completely describes how the universe works
- "Machine learning": estimate $p(x_1, \ldots, x_n)$ from training data $X^{(1)}, \ldots, X^{(N)}$, where $X^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$
- "Prediction": $y^* = \operatorname{argmax} p(x_n \mid x_1^*, \ldots, x_n^*)$, a.k.a. inference
  - by the definition of conditional probability

$$p(x_n \mid x_1^*, \ldots, x_n^*) = \frac{p(x_1^*, \ldots, x_n^*, x_n)}{\sum_v p(x_1^*, \ldots, x_n^*, x_n = v)}$$

Conclusion

- Life without graphical models is just fine
- So why are we still here?

Life can be Better for Computer Scientists

- Given GM = joint distribution $p(x_1, \ldots, x_n)$
  - exponential naïve storage (2^n for binary r.v.)
  - hard to interpret (conditional independence)
- Do inference $= p(X_Q \mid X_E)$, in general $X_Q \cup X_E \subset \{x_1 \ldots x_n\}$
  - Often can't do it computationally
- If $p(x_1, \ldots, x_n)$ not given, estimate it from data
  - Can't do it either

Acknowledgments Before We Start

Much of this tutorial is based on

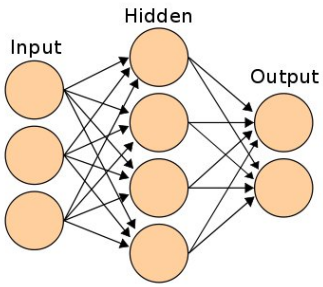
- Koller & Friedman, *Probabilistic Graphical Models*. MIT 2009
- Wainwright & Jordan, *Graphical Models, Exponential Families, and Variational Inference*. FTML 2008
- Bishop, *Pattern Recognition and Machine Learning*. Springer 2006.

Outline

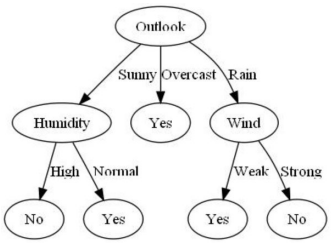
Graphical-Model-Nots

- "Graphical model" is the study of *probabilistic models*
- Just because there is a graph with nodes and edges doesn't mean it's GM

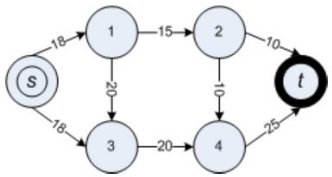
These are not graphical models



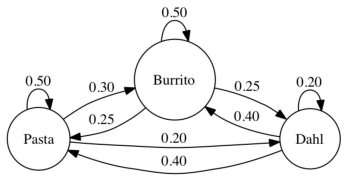
neural network



decision tree



network flow



HMM template

Outline

Bayesian Network

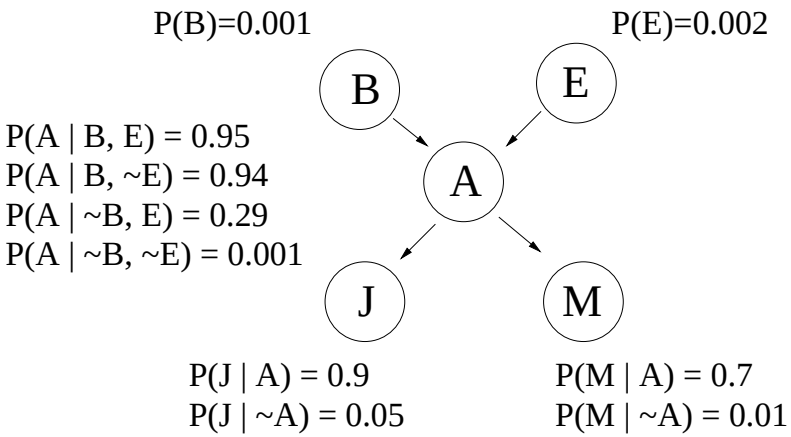
- A directed graph has nodes $X = (x_1, \ldots, x_n)$, some of them connected by directed edges $x_i \to x_j$
- A cycle is a directed path $x_1 \to \ldots \to x_k$ where $x_1 = x_k$
- A directed acyclic graph (DAG) contains no cycles
- A Bayesian network on the DAG is a family of distributions satisfying

$$\{p \mid p(X) = \prod_i p(x_i \mid Pa(x_i))\}$$

  where $Pa(x_i)$ is the set of parents of x_i.
- $p(x_i \mid Pa(x_i))$ is the conditional probability distribution (CPD) at x_i
- By specifying the CPDs for all i, we specify a particular distribution $p(X)$

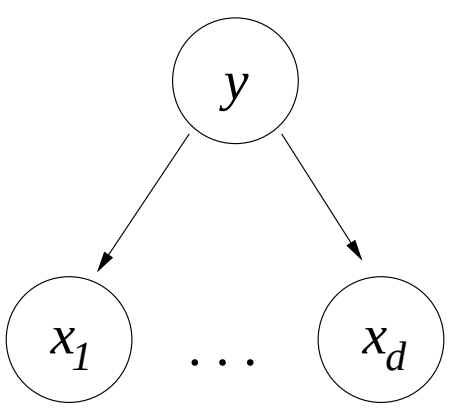
Example: Alarm

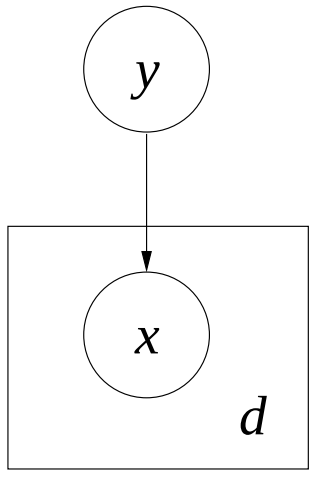
Binary variables



$$
\begin{aligned}
& P(B, \sim E, A, J, \sim M) \\
= \quad & P(B)P(\sim E)P(A \mid B, \sim E)P(J \mid A)P(\sim M \mid A) \\
= \quad & 0.001 \times (1 - 0.002) \times 0.94 \times 0.9 \times (1 - 0.7) \\
\approx \quad & .000253
\end{aligned}
$$

Example: Naive Bayes

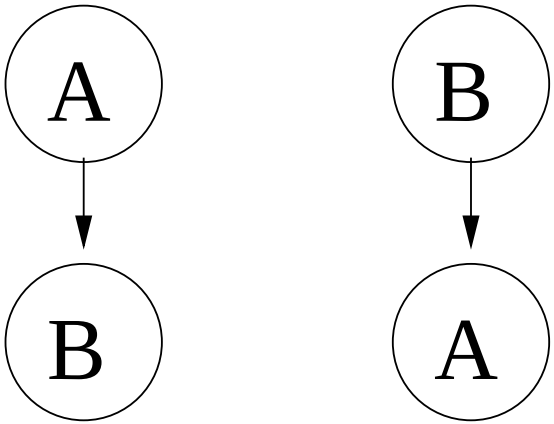




- $p(y, x_1, \ldots x_d) = p(y) \prod_{i=1}^{d} p(x_i \mid y)$
- Used extensively in natural language processing
- Plate representation on the right

No Causality Whatsoever

$P(A)=a$
$P(B|A)=b$
$P(B|\sim A)=c$

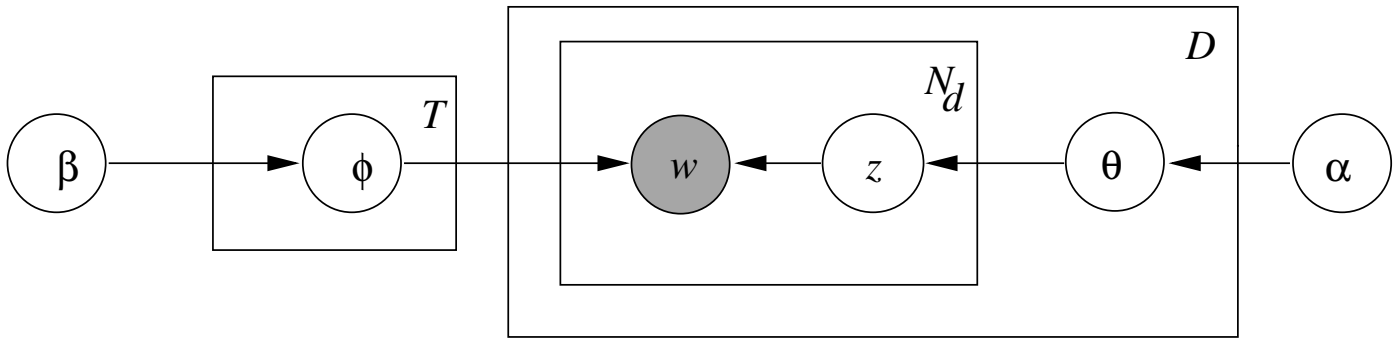


$P(B)=ab+(1-a)c$
$P(A|B)=ab/(ab+(1-a)c)$
$P(A|\sim B)=a(1-b)/(1-ab-(1-a)c)$

The two BNs are equivalent in all respects

- Bayesian networks imply no causality at all
- They only encode the joint probability distribution (hence correlation)
- However, people tend to design BNs based on causal relations

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

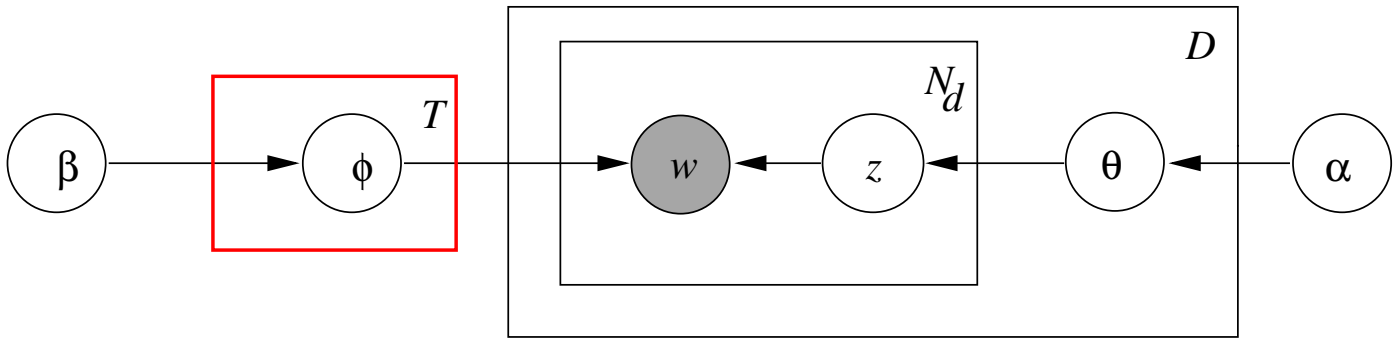
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

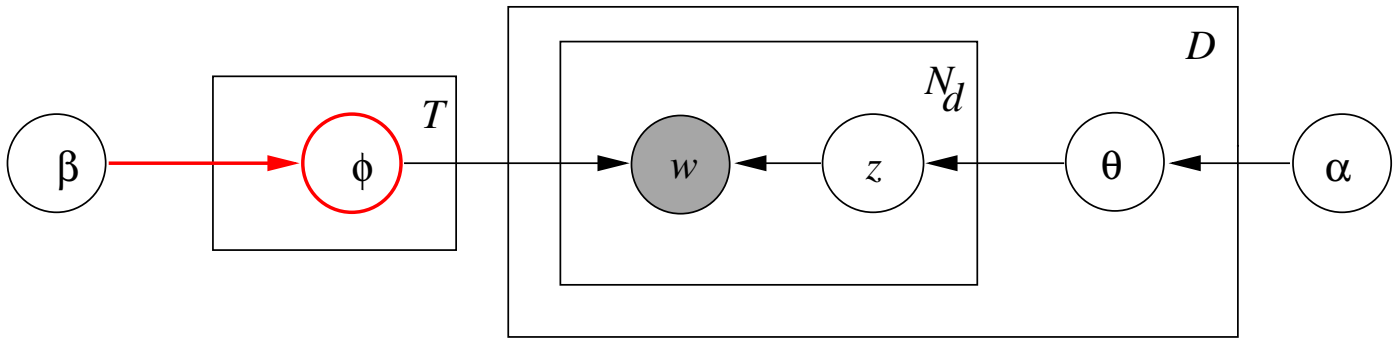
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

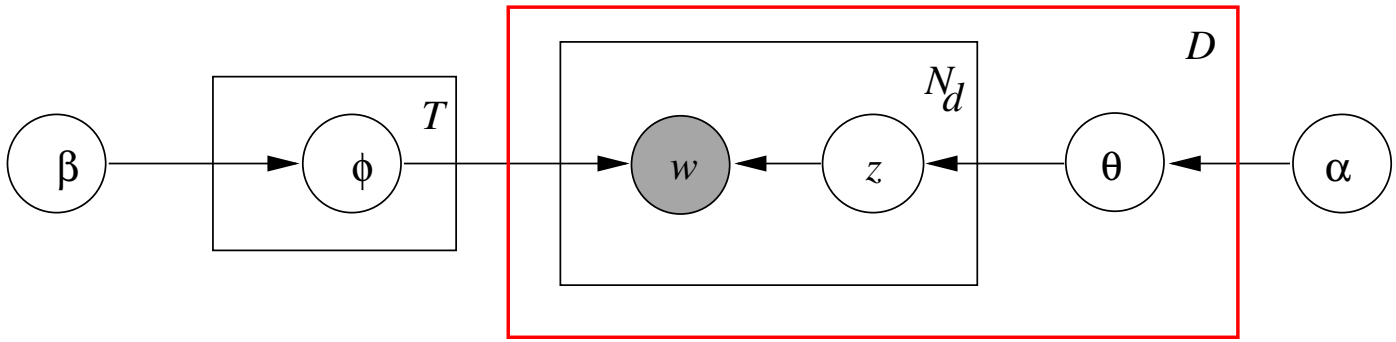
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

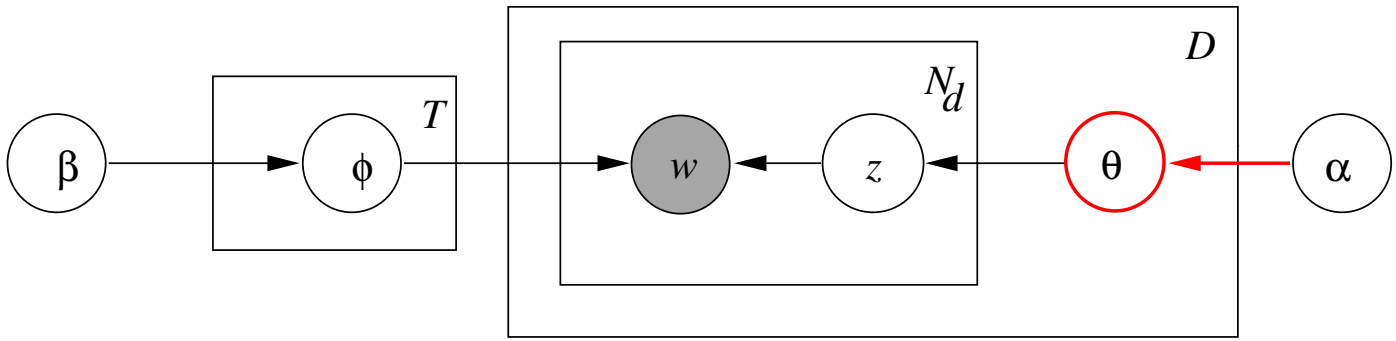
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

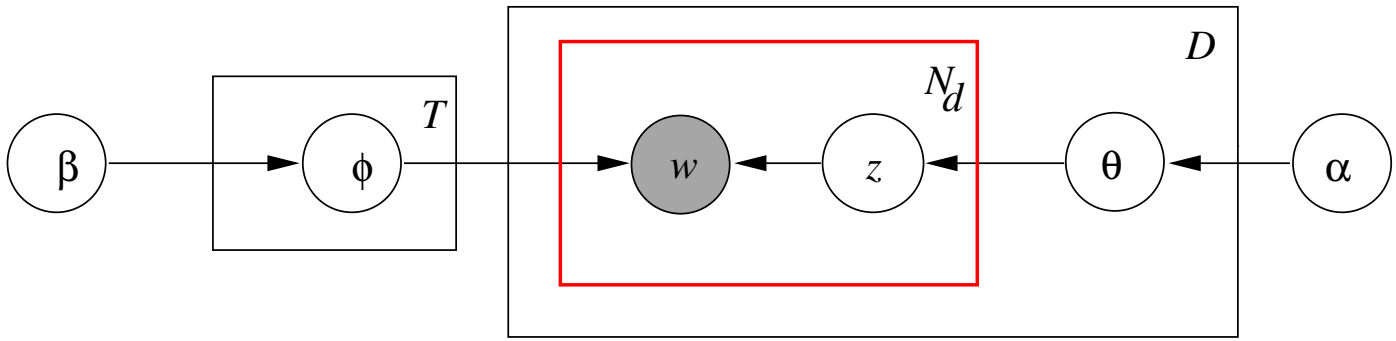
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

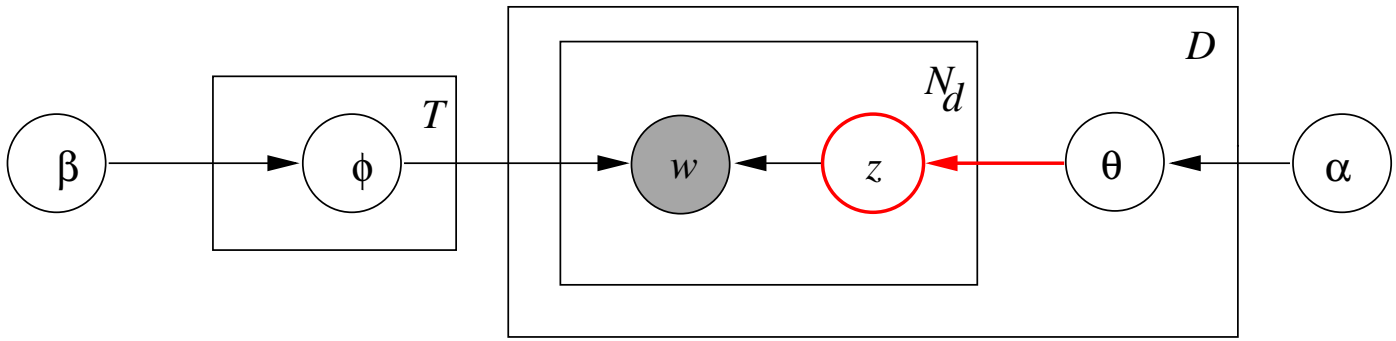
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

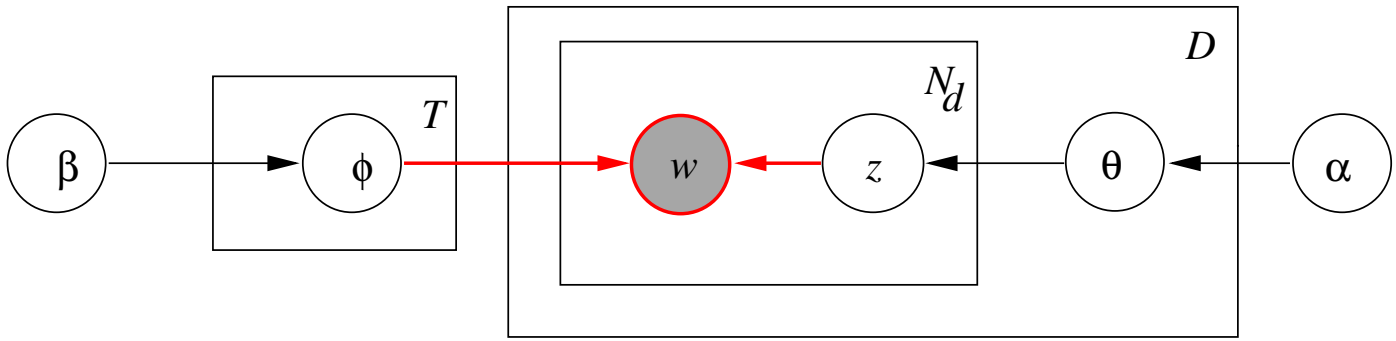
For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Example: Latent Dirichlet Allocation (LDA)



A generative model for $p(\phi, \theta, z, w \mid \alpha, \beta)$:

For each topic t

$\phi_t \sim \text{Dirichlet}(\beta)$

For each document d

$\theta \sim \text{Dirichlet}(\alpha)$

For each word position in d

topic $z \sim \text{Multinomial}(\theta)$

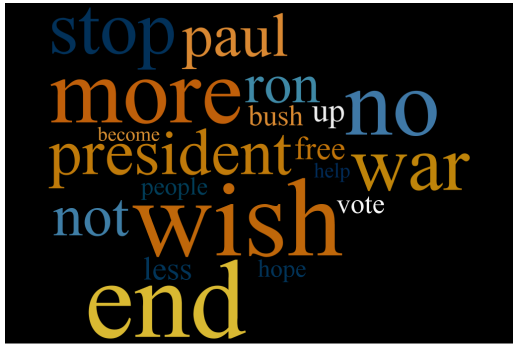
word $w \sim \text{Multinomial}(\phi_z)$

Inference goals: $p(z \mid w, \alpha, \beta), \text{argmax}_{\phi,\theta}\, p(\phi, \theta \mid w, \alpha, \beta)$

Some Topics by LDA on the Wish Corpus

$p(\text{word} \mid \text{topic})$

"troops"



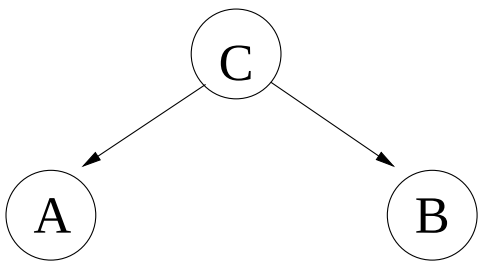
"election"

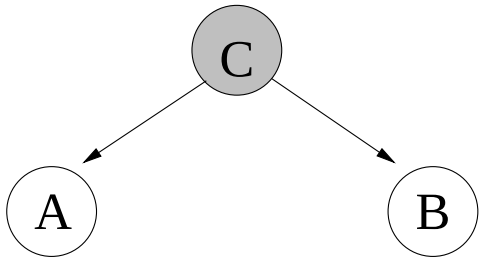
"love"

Conditional Independence

- Two r.v.s A, B are independent if $P(A, B) = P(A)P(B)$ or $P(A|B) = P(A)$ (the two are equivalent)
- Two r.v.s A, B are conditionally independent given C if $P(A, B \mid C) = P(A \mid C)P(B \mid C)$ or $P(A \mid B, C) = P(A \mid C)$ (the two are equivalent)
- This extends to groups of r.v.s
- Conditional independence in a BN is precisely specified by d-separation ("directed separation")

d-Separation Case 1: Tail-to-Tail



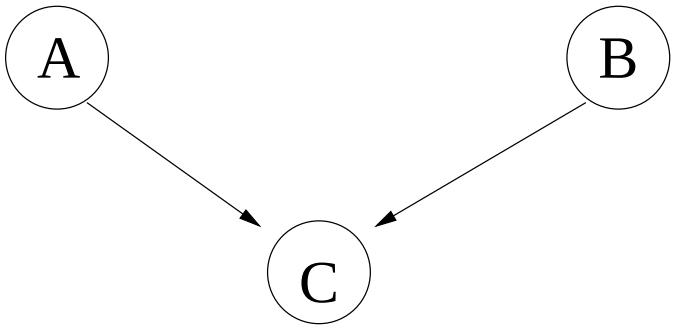


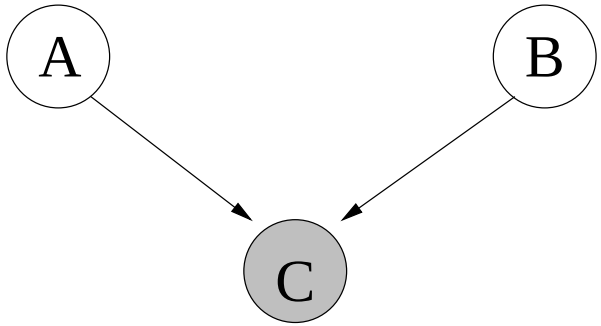
- A, B in general dependent
- A, B conditionally independent given C
- C is a tail-to-tail node, blocks the undirected path A-B

d-Separation Case 2: Head-to-Tail

- A, B in general dependent
- A, B conditionally independent given C
- C is a head-to-tail node, blocks the path A-B

d-Separation Case 3: Head-to-Head





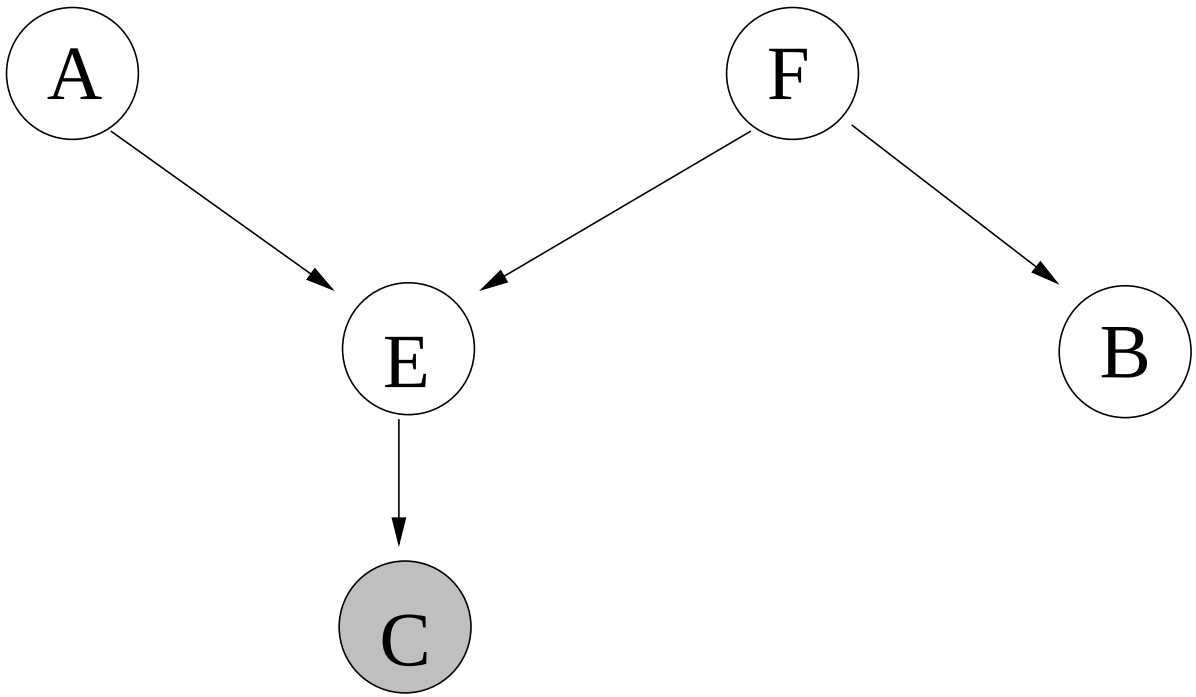
- A, B in general independent
- A, B conditionally dependent given C, or any of C's descendants
- C is a head-to-head node, unblocks the path A-B

d-Separation

- Any groups of nodes A and B are conditionally independent given another group C, if all undirected paths from any node in A to any node in B are *blocked*
- A path is blocked if it includes a node x such that either
  - The path is head-to-tail or tail-to-tail at x and $x \in$ C, or
  - The path is head-to-head at x, and neither x nor any of its descendants is in C.

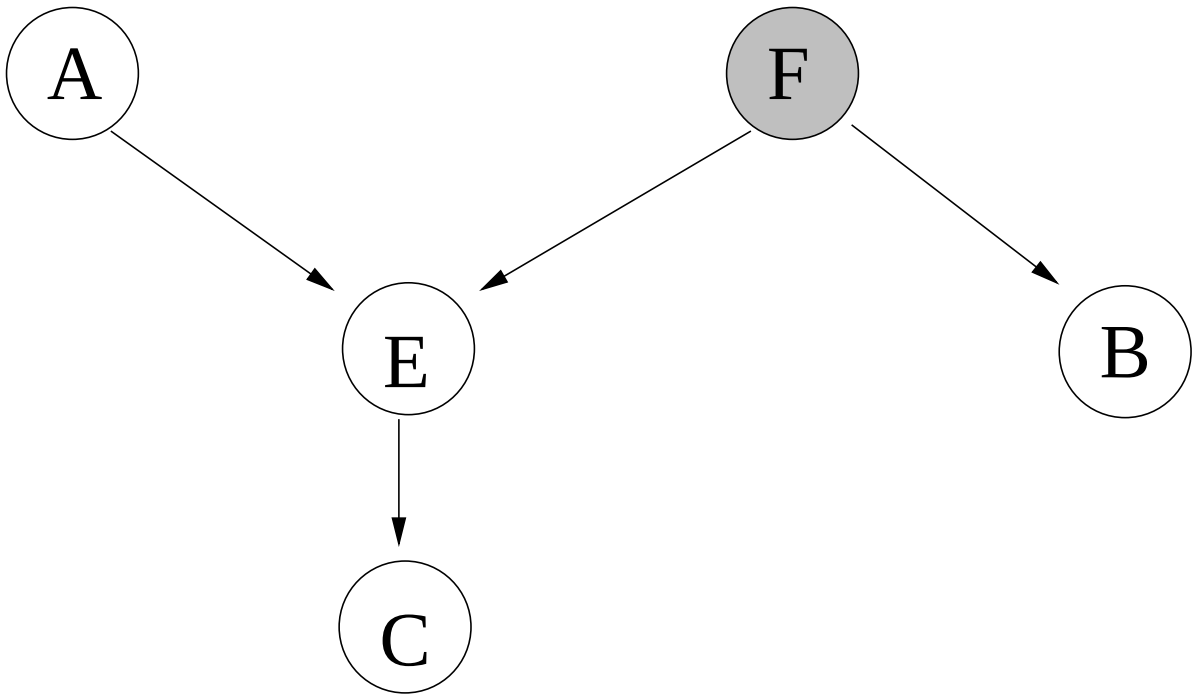
d-Separation Example 1

- The path from A to B not blocked by either E or F
- A, B dependent given C



d-Separation Example 2

- The path from A to B is blocked both at E and F
- A, B conditionally independent given F



Outline

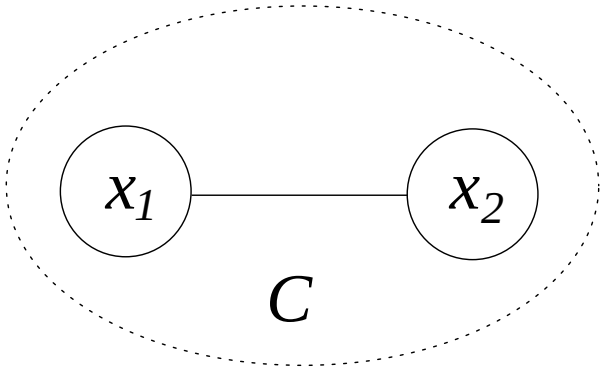
Markov Random Fields

- The efficiency of directed graphical model (acyclic graph, locally normalized CPDs) also makes it restrictive
- A clique C in an undirected graph is a fully connected set of nodes (note: full of loops!)
- Define a nonnegative potential function $\psi_C : X_C \mapsto \mathbb{R}_+$
- An undirected graphical model (aka Markov Random Field) on the graph is a family of distributions satisfying

$$\left\{ p \mid p(X) = \frac{1}{Z} \prod_C \psi_C(X_C) \right\}$$

- $Z = \int \prod_C \psi_C(X_C) dX$ is the partition function

Example: A Tiny Markov Random Field



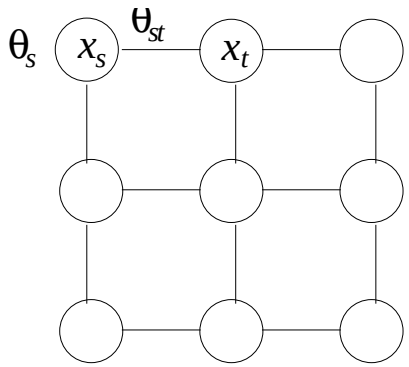
- $x_1, x_2 \in \{-1, 1\}$
- A single clique $\psi_C(x_1, x_2) = e^{ax_1x_2}$
- $p(x_1, x_2) = \frac{1}{Z} e^{ax_1x_2}$
- $Z = (e^a + e^{-a} + e^{-a} + e^a)$
- $p(1, 1) = p(-1, -1) = e^a/(2e^a + 2e^{-a})$
- $p(-1, 1) = p(1, -1) = e^{-a}/(2e^a + 2e^{-a})$
- When the parameter $a > 0$, favor homogeneous chains
- When the parameter $a < 0$, favor inhomogeneous chains

Log Linear Models

- Real-valued feature functions $f_1(X), \ldots, f_k(X)$
- Real-valued weights $w_1, \ldots, w_k$

$$p(X) = \frac{1}{Z} \exp\left(-\sum_{i=1}^{k} w_i f_i(X)\right)$$

Example: The Ising Model

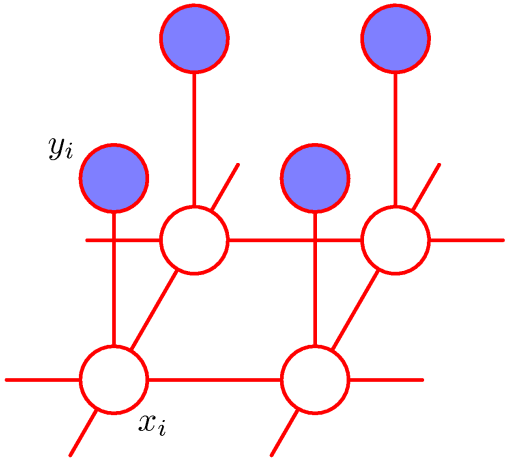


This is an undirected model with $x \in \{0, 1\}$.

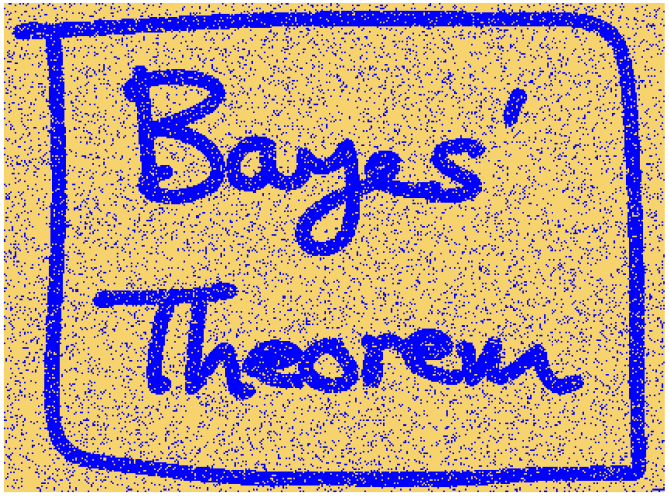
$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t\right)$$

- $f_s(X) = x_s$, $f_{st}(X) = x_s x_t$
- $w_s = -\theta_s$, $w_{st} = -\theta_{st}$

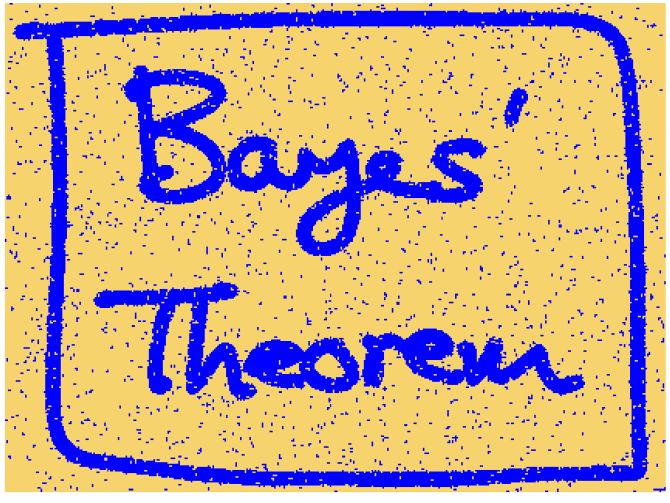
Example: Image Denoising



noisy image



$\text{argmax}_X P(X|Y)$



[From Bishop PRML]

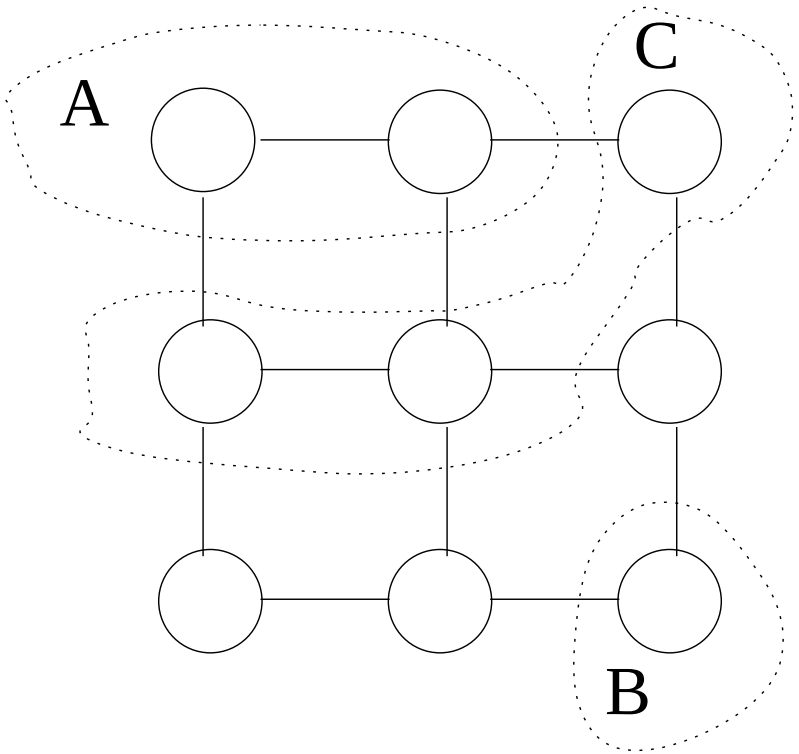
Example: Gaussian Random Field

$$p(X) \sim N(\mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu)^\top \Sigma^{-1}(X-\mu)\right)$$

- Multivariate Gaussian
- The $n \times n$ covariance matrix Σ positive semi-definite
- Let $\Omega = \Sigma^{-1}$ be the precision matrix
- x_i, x_j are conditionally independent given all other variables, if and only if $\Omega_{ij} = 0$
- When $\Omega_{ij} \neq 0$, there is an edge between x_i, x_j

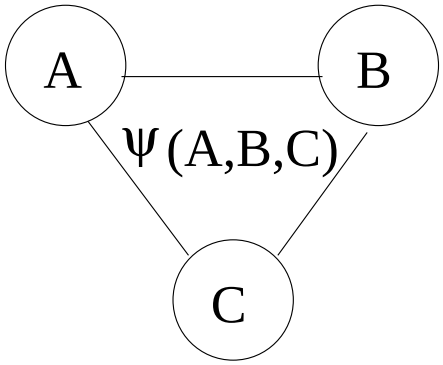
Conditional Independence in Markov Random Fields

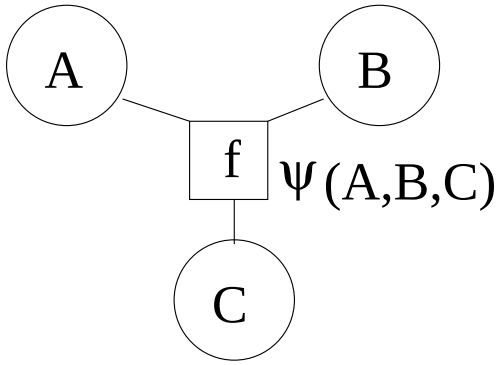
- Two group of variables A, B are conditionally independent given another group C, if
  - Remove C and all edges involving C
  - A, B beome disconnected

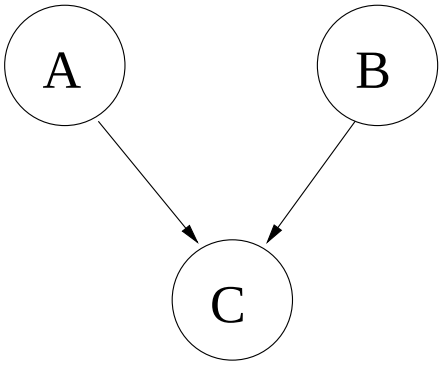


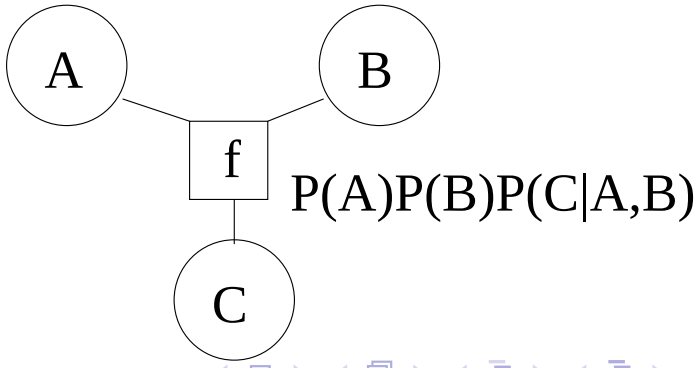
Factor Graph

- For both directed and undirected graphical models
- Bipartite: edges between a variable node and a factor node
- Factors represent computation









Outline

Outline

Inference by Enumeration

- Let $X = (X_Q, X_E, X_O)$ for query, evidence, and other variables.
- Infer $P(X_Q \mid X_E)$
- By definition

$$P(X_Q \mid X_E) = \frac{P(X_Q, X_E)}{P(X_E)} = \frac{\sum_{X_O} P(X_Q, X_E, X_O)}{\sum_{X_Q, X_O} P(X_Q, X_E, X_O)}$$

- Summing exponential number of terms: with k variables in X_O each taking r values, there are r^k terms

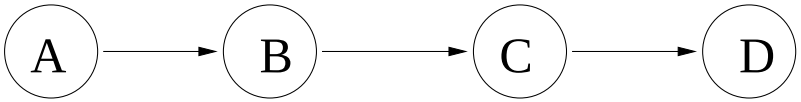
Details of the summing problem

- There are a bunch of "other" variables $x_1, \ldots, x_k$
- We sum over r values each variable can take $\sum_{x_i=v_1}^{v_r}$
- This is exponential (r^k): $\sum_{x_1 \ldots x_k}$
- We want $\sum_{x_1 \ldots x_k} p(X)$
- For a graphical model, the joint probability factors $p(X) = \prod_{j=1}^{m} f_j(X_{(j)})$
- Each factor f_j operates on $X_{(j)} \subseteq X$

Eliminating a Variable

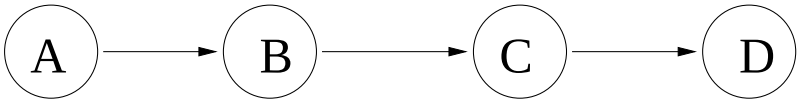
- Rearrange factors $\sum_{x_1 \ldots x_k} f_1^- \ldots f_l^- f_{l+1}^+ \ldots f_m^+$ by whether $x_1 \in X_{(j)}$
- Obviously equivalent: $\sum_{x_2 \ldots x_k} f_1^- \ldots f_l^- \left(\sum_{x_1} f_{l+1}^+ \ldots f_m^+ \right)$
- Introduce a new factor $f_{m+1}^- = \left(\sum_{x_1} f_{l+1}^+ \ldots f_m^+ \right)$
- f_{m+1}^- contains the union of variables in $f_{l+1}^+ \ldots f_m^+$ except x_1
- In fact, x_1 disappears altogether in $\sum_{x_2 \ldots x_k} f_1^- \ldots f_l^- f_{m+1}^-$
- Dynamic programming: compute f_{m+1}^- once, use it thereafter
- Hope: f_{m+1}^- contains very few variables
- Recursively eliminate other variables in turn

Example: Chain Graph



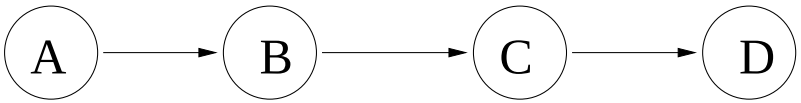
- Binary variables
- Say we want $P(D) = \sum_{A,B,C} P(A)P(B|A)P(C|B)P(D|C)$
- Let $f_1(A) = P(A)$. Note f_1 is an array of size two:
  $P(A = 0)$
  $P(A = 1)$
- $f_2(A, B)$ is a table of size four:
  $P(B = 0|A = 0)$
  $P(B = 0|A = 1)$
  $P(B = 1|A = 0)$
  $P(B = 1|A = 1))$
- $\sum_{A,B,C} f_1(A)f_2(A, B)f_3(B, C)f_4(C, D) =$
  $\sum_{B,C} f_3(B, C)f_4(C, D)(\sum_A f_1(A)f_2(A, B))$

Example: Chain Graph



- $f_1(A)f_2(A,B)$ an array of size four: match A values
  $P(A=0)P(B=0|A=0)$
  $P(A=1)P(B=0|A=1)$
  $P(A=0)P(B=1|A=0)$
  $P(A=1)P(B=1|A=1)$
- $f_5(B) \equiv \sum_A f_1(A)f_2(A,B)$ an array of size two
  $P(A=0)P(B=0|A=0) + P(A=1)P(B=0|A=1)$
  $P(A=0)P(B=1|A=0) + P(A=1)P(B=1|A=1)$
- For this example, $f_5(B)$ happens to be $P(B)$
- $\sum_{B,C} f_3(B,C)f_4(C,D)f_5(B) =$
  $\sum_C f_4(C,D)(\sum_B f_3(B,C)f_5(B))$, and so on
- In the end, $f_7(D) = (P(D=0), P(D=1))$

Example: Chain Graph



- Computation for $P(D)$: 12 $\times$, 6 $+$
- Enumeration: 48 $\times$, 14 $+$
- Saving depends on elimination order. Finding optimal order NP-hard; there are heuristic methods.
- Saving depends more critically on the graph structure (tree width), can be intractable

Handling Evidence

- For evidence variables X_E, simply plug in their value e
- Eliminate variables $X_O = X - X_E - X_Q$
- The final factor will be the *joint* $f(X_Q) = P(X_Q, X_E = e)$
- Normalize to answer query:

$$P(X_Q \mid X_E = e) = \frac{f(X_Q)}{\sum_{X_Q} f(X_Q)}$$

Summary: Exact Inference

- Enumeration
- Variable elimination
- Not covered: junction tree (aka clique tree)

Exact, but intractable for large graphs

Outline

Inference by Monte Carlo

- Consider the inference problem $p(X_Q = c_Q \mid X_E)$ where $X_Q \cup X_E \subseteq \{x_1 \dots x_n\}$

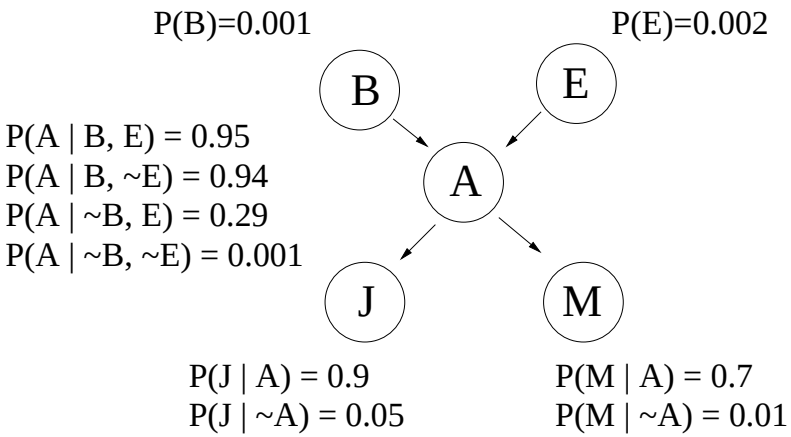
$$p(X_Q = c_Q \mid X_E) = \int 1_{(x_Q = c_Q)} p(x_Q \mid X_E) dx_Q$$

- If we can draw samples $x_Q^{(1)}, \dots x_Q^{(m)} \sim p(x_Q \mid X_E)$, an unbiased estimator is

$$p(X_Q = c_Q \mid X_E) \approx \frac{1}{m} \sum_{i=1}^{m} 1_{(x_Q^{(i)} = c_Q)}$$

- The variance of the estimator decreases as $\mathbb{V}/m$
- Inference reduces to sampling from $p(x_Q \mid X_E)$

Forward Sampling Example



To generate a sample $X = (B, E, A, J, M)$:

1. Sample $B \sim \text{Ber}(0.001)$: $r \sim U(0, 1)$. If $(r < 0.001)$ then $B = 1$ else $B = 0$
2. Sample $E \sim \text{Ber}(0.002)$
3. If $B = 1$ and $E = 1$, sample $A \sim \text{Ber}(0.95)$, and so on
4. If $A = 1$ sample $J \sim \text{Ber}(0.9)$ else $J \sim \text{Ber}(0.05)$
5. If $A = 1$ sample $M \sim \text{Ber}(0.7)$ else $M \sim \text{Ber}(0.01)$

Works for Bayesian networks.

Inference with Forward Sampling

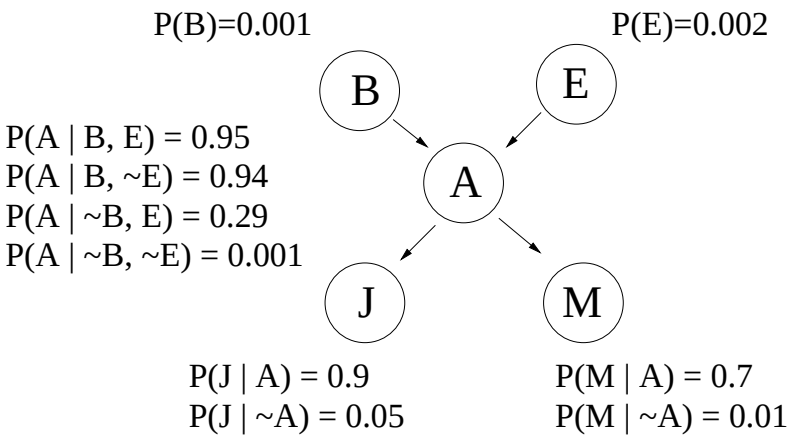
- Say the inference task is $P(B = 1 \mid E = 1, M = 1)$
- Throw away all samples except those with $(E = 1, M = 1)$

$$p(B = 1 \mid E = 1, M = 1) \approx \frac{1}{m} \sum_{i=1}^{m} 1_{(B^{(i)}=1)}$$

  where m is the number of surviving samples
- Can be highly inefficient (note $P(E = 1)$ tiny)
- Does not work for Markov Random Fields

Gibbs Sampler Example: $P(B = 1 \mid E = 1, M = 1)$

- Gibbs sampler is a Markov Chain Monte Carlo (MCMC) method.
- Directly sample from $p(x_Q \mid X_E)$
- Works for both graphical models
- Initialization:
  - Fix evidence; randomly set other variables
  - e.g. $X^{(0)} = (B = 0, E = 1, A = 0, J = 0, M = 1)$

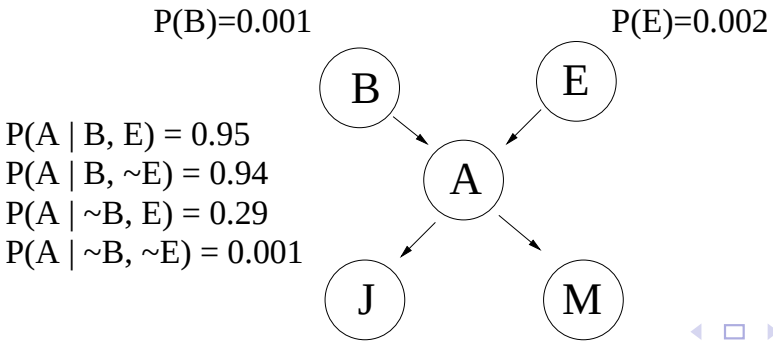


Gibbs Update

- For each non-evidence variable x_i, fixing all other nodes X_{-i}, resample its value $x_i \sim P(x_i \mid X_{-i})$
- This is equivalent to $x_i \sim P(x_i \mid \text{MarkovBlanket}(x_i))$
- For a Bayesian network $\text{MarkovBlanket}(x_i)$ includes x_i's parents, spouses, and children

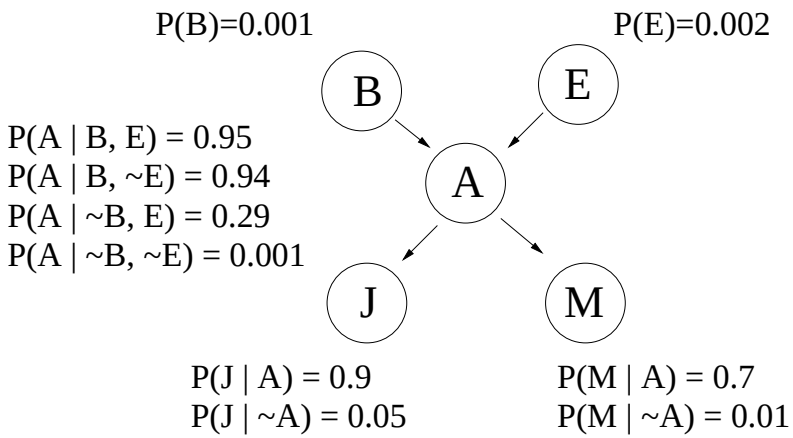
$$P(x_i \mid \text{MarkovBlanket}(x_i)) \propto P(x_i \mid Pa(x_i)) \prod_{y \in C(x_i)} P(y \mid Pa(y))$$

  where $Pa(x)$ are the parents of x, and $C(x)$ the children of x.
- For many graphical models the Markov Blanket is small.
- For example,
  $B \sim P(B \mid E = 1, A = 0) \propto P(B)P(A = 0 \mid B, E = 1)$

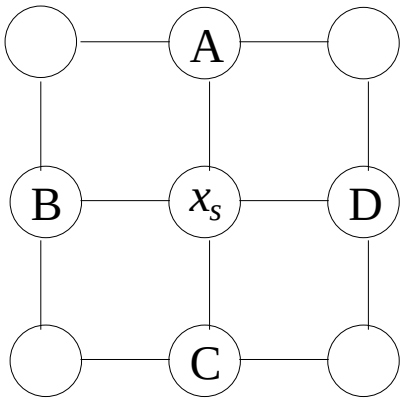


Gibbs Update

- Say we sampled $B = 1$. Then $X^{(1)} = (B = 1, E = 1, A = 0, J = 0, M = 1)$
- Starting from $X^{(1)}$, sample $A \sim P(A \mid B = 1, E = 1, J = 0, M = 1)$ to get $X^{(2)}$
- Move on to J, then repeat $B, A, J, B, A, J \ldots$
- Keep all *later* samples. $P(B = 1 \mid E = 1, M = 1)$ is the fraction of samples with $B = 1$.



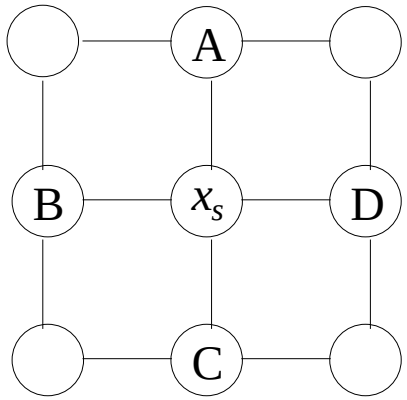
Gibbs Example 2: The Ising Model



This is an undirected model with $x \in \{0, 1\}$.

$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t \right)$$

Gibbs Example 2: The Ising Model



- The Markov blanket of x_s is A, B, C, D
- In general for undirected graphical models

$$p(x_s \mid x_{-s}) = p(x_s \mid x_{N(s)})$$

$N(s)$ is the neighbors of s.

- The Gibbs update is

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

Gibbs Sampling as a Markov Chain

- A Markov chain is defined by a transition matrix $T(X' \mid X)$
- Certain Markov chains have a stationary distribution π such that $\pi = T\pi$
- Gibbs sampler is such a Markov chain with $T_i((X_{-i}, x_i') \mid (X_{-i}, x_i)) = p(x_i' \mid X_{-i})$, and stationary distribution $p(x_Q \mid X_E)$
- But it takes time for the chain to reach stationary distribution (mix)
  - Can be difficult to assert mixing
  - In practice "burn in": discard $X^{(0)}, \ldots, X^{(T)}$
  - Use all of $X^{(T+1)}, \ldots$ for inference (they are correlated)
  - Do not thin

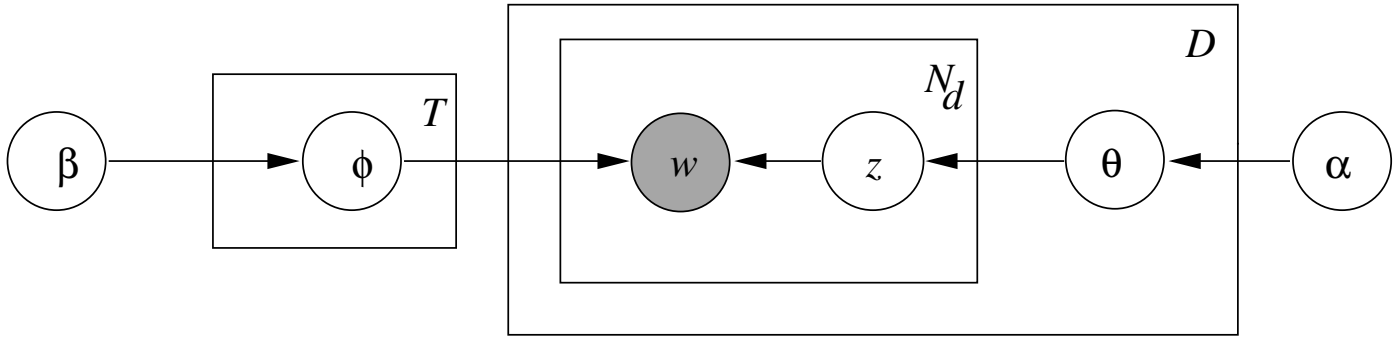
Collapsed Gibbs Sampling

- In general, $\mathbb{E}_p[f(X)] \approx \frac{1}{m}\sum_{i=1}^{m} f(X^{(i)})$ if $X^{(i)} \sim p$
- Sometimes $X = (Y, Z)$ where Z has closed-form operations
- If so,

$$\begin{aligned}\mathbb{E}_p[f(X)] &= \mathbb{E}_{p(Y)}\mathbb{E}_{p(Z|Y)}[f(Y,Z)] \\ &\approx \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{p(Z|Y^{(i)})}[f(Y^{(i)},Z)]\end{aligned}$$

  if $Y^{(i)} \sim p(Y)$
- No need to sample Z: it is collapsed
- Collapsed Gibbs sampler $T_i((Y_{-i}, y_i') \mid (Y_{-i}, y_i)) = p(y_i' \mid Y_{-i})$
- Note $p(y_i' \mid Y_{-i}) = \int p(y_i', Z \mid Y_{-i})dZ$

Example: Collapsed Gibbs Sampling for LDA



Collapse θ, ϕ, Gibbs update:

$$P(z_i = j \mid \mathbf{z}_{-i}, \mathbf{w}) \propto \frac{n_{-i,j}^{(w_i)} + \beta n_{-i,j}^{(d_i)} + \alpha}{n_{-i,j}^{(\cdot)} + W\beta n_{-i,\cdot}^{(d_i)} + T\alpha}$$

- $n_{-i,j}^{(w_i)}$: number of times word w_i has been assigned to topic j, excluding the current position
- $n_{-i,j}^{(d_i)}$: number of times a word from document d_i has been assigned to topic j, excluding the current position
- $n_{-i,j}^{(\cdot)}$: number of times any word has been assigned to topic j, excluding the current position
- $n_{-i,\cdot}^{(d_i)}$: length of document d_i, excluding the current position

Summary: Markov Chain Monte Carlo

- Gibbs sampling
- Not covered: block Gibbs, Metropolis-Hastings

Unbiased (after burn-in), but can have high variance

To learn more, come to Prof. Prasad Tetali's tutorial "Markov Chain Mixing with Applications" 2pm Monday.

Outline

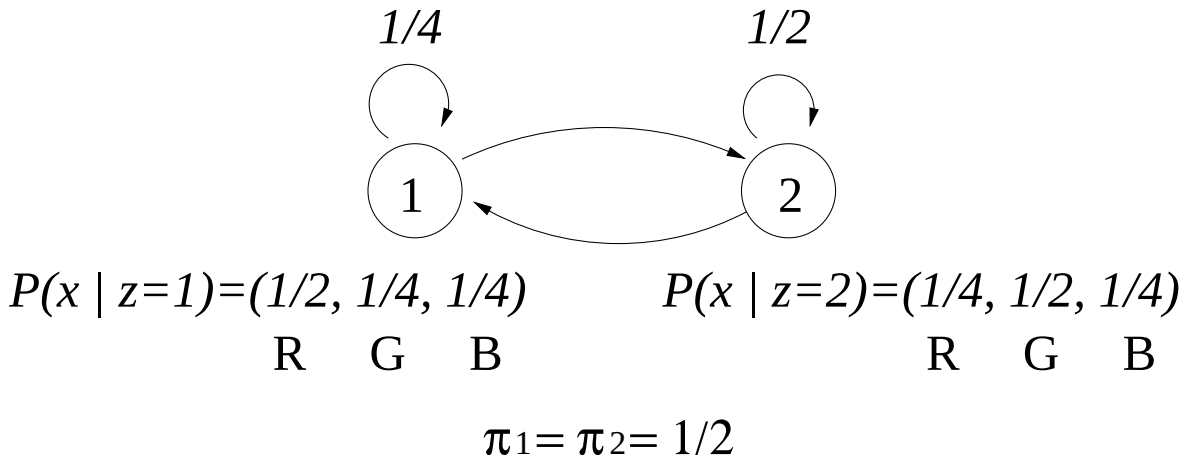
Outline

The Sum-Product Algorithm

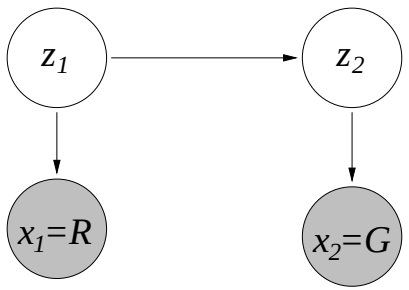
- Also known as belief propagation (BP)
- Exact if the graph is a tree; otherwise known as "loopy BP", approximate
- The algorithm involves passing *messages* on the factor graph
- Alternative view: variational approximation (more later)

Example: A Simple HMM

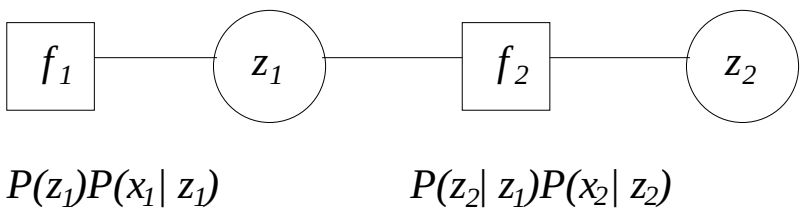
- The Hidden Markov Model template (not a graphical model)



- Observing $x_1 = R, x_2 = G$, the directed graphical model



- Factor graph



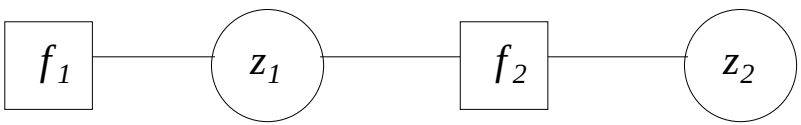
Messages

A message is a vector of length K, where K is the number of values x takes.
There are two types of messages:

1. $\mu_{f \to x}$: message from a factor node f to a variable node x
   $\mu_{f \to x}(i)$ is the ith element, $i = 1 \ldots K$.
2. $\mu_{x \to f}$: message from a variable node x to a factor node f

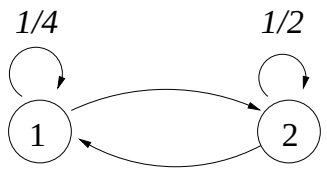
Leaf Messages

- Assume tree factor graph. Pick an arbitrary root, say z_2
- Start messages at leaves.
- If a leaf is a factor node f, $\mu_{f \to x}(x) = f(x)$
- If a leaf is a variable node x, $\mu_{x \to f}(x) = 1$



$P(z_1)P(x_1|\, z_1)$ $\qquad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

$$\mu_{f_1 \to z_1}(z_1 = 1) = P(z_1 = 1)P(R|z_1 = 1) = 1/2 \cdot 1/2 = 1/4$$
$$\mu_{f_1 \to z_1}(z_1 = 2) = P(z_1 = 2)P(R|z_1 = 2) = 1/2 \cdot 1/4 = 1/8$$



$P(x \mid z=1)=(1/2,\ 1/4,\ 1/4)$ $\qquad$ $P(x \mid z=2)=(1/4,\ 1/2,\ 1/4)$

R G B $\qquad$ R G B

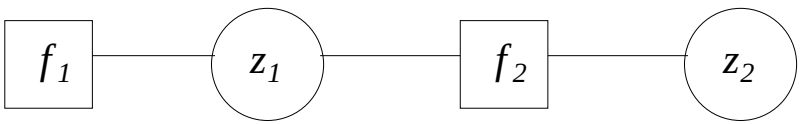
$\pi_1 = \pi_2 = 1/2$

Message from Variable to Factor

- A node (factor or variable) can send out a message if all other incoming messages have arrived
- Let x be in factor f_s.

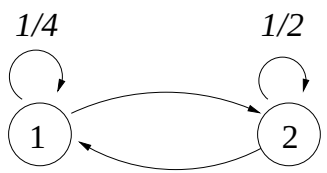
$$\mu_{x \to f_s}(x) = \prod_{f \in ne(x) \backslash f_s} \mu_{f \to x}(x)$$

- $ne(x) \backslash f_s$ are factors connected to x excluding f_s.



$P(z_1)P(x_1|\, z_1)$ $\qquad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

$$\mu_{z_1 \to f_2}(z_1 = 1) = 1/4$$
$$\mu_{z_1 \to f_2}(z_1 = 2) = 1/8$$



$P(x \mid z=1)=(1/2, 1/4, 1/4)$ $\qquad$ $P(x \mid z=2)=(1/4, 1/2, 1/4)$

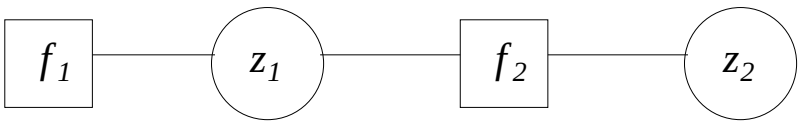
R G B $\qquad$ R G B

$\pi_1 = \pi_2 = 1/2$

Message from Factor to Variable

- Let x be in factor f_s. Let the other variables in f_s be $x_{1:M}$.

$$\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m)$$



$P(z_1)P(x_1|\, z_1)$ $\qquad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

$$\begin{aligned}
\mu_{f_2 \to z_2}(s) &= \sum_{s'=1}^{2} \mu_{z_1 \to f_2}(s') f_2(z_1 = s', z_2 = s) \\
&= 1/4 P(z_2 = s | z_1 = 1) P(x_2 = G | z_2 = s) \\
&\quad + 1/8 P(z_2 = s | z_1 = 2) P(x_2 = G | z_2 = s)
\end{aligned}$$

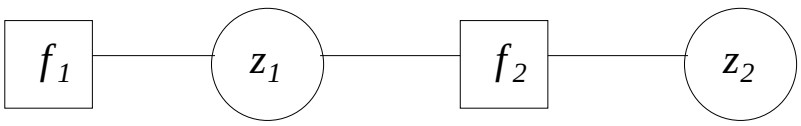
We get

$\mu_{f_2 \to z_2}(z_2 = 1) = 1/32$

$\mu_{f_2 \to z_2}(z_2 = 2) = 1/8$

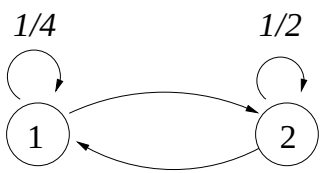
Up to Root, Back Down

- The message has reached the root, pass it back down



$P(z_1)P(x_1|\, z_1)$ $\qquad$ $P(z_2|\, z_1)P(x_2|\, z_2)$

$$\mu_{z_2 \to f_2}(z_2 = 1) = 1$$
$$\mu_{z_2 \to f_2}(z_2 = 2) = 1$$

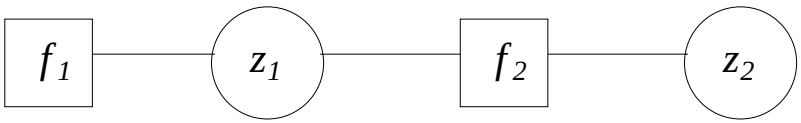


$P(x \mid z=1)=(1/2, 1/4, 1/4)$ $\qquad$ $P(x \mid z=2)=(1/4, 1/2, 1/4)$

R G B $\qquad$ R G B

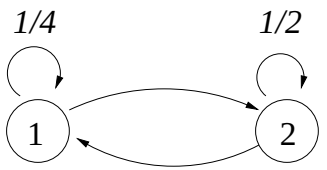
$\pi_1 = \pi_2 = 1/2$

Keep Passing Down



$P(z_1)P(x_1|\, z_1)$ $P(z_2|\, z_1)P(x_2|\, z_2)$

$\mu_{f_2 \to z_1}(s) = \sum_{s'=1}^{2} \mu_{z_2 \to f_2}(s') f_2(z_1 = s, z_2 = s')$
$= 1P(z_2 = 1|z_1 = s)P(x_2 = G|z_2 = 1)$
$\quad + 1P(z_2 = 2|z_1 = s)P(x_2 = G|z_2 = 2)$. We get
$\mu_{f_2 \to z_1}(z_1 = 1) = 7/16$
$\mu_{f_2 \to z_1}(z_1 = 2) = 3/8$



$P(x \mid z=1)=(1/2, 1/4, 1/4)$ $P(x \mid z=2)=(1/4, 1/2, 1/4)$

R G B R G B

$\pi_1 = \pi_2 = 1/2$

From Messages to Marginals

Once a variable receives all incoming messages, we compute its marginal as

$$p(x) \propto \prod_{f \in ne(x)} \mu_{f \to x}(x)$$

In this example

$P(z_1|x_1, x_2) \propto \mu_{f_1 \to z_1} \cdot \mu_{f_2 \to z_1} = \begin{pmatrix} 1/4 \\ 1/8 \end{pmatrix} \cdot \begin{pmatrix} 7/16 \\ 3/8 \end{pmatrix} = \begin{pmatrix} 7/64 \\ 3/64 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix}$

$P(z_2|x_1, x_2) \propto \mu_{f_2 \to z_2} = \begin{pmatrix} 1/32 \\ 1/8 \end{pmatrix} \Rightarrow \begin{pmatrix} 0.2 \\ 0.8 \end{pmatrix}$

One can also compute the marginal of the *set of* variables x_s involved in a factor f_s

$$p(x_s) \propto f_s(x_s) \prod_{x \in ne(f)} \mu_{x \to f}(x)$$

Handling Evidence

Observing $x = v$,

- we can absorb it in the factor (as we did); or
- set messages $\mu_{x \to f}(x) = 0$ for all $x \neq v$

Observing X_E,

- multiplying the incoming messages to $x \notin X_E$ gives the *joint* (not $p(x|X_E)$):

$$p(x, X_E) \propto \prod_{f \in ne(x)} \mu_{f \to x}(x)$$

- The conditional is easily obtained by normalization

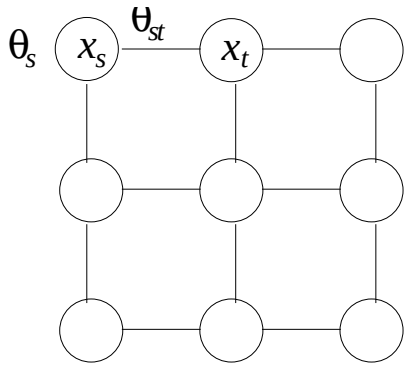
$$p(x|X_E) = \frac{p(x, X_E)}{\sum_{x'} p(x', X_E)}$$

Loopy Belief Propagation

- So far, we assumed a tree graph
- When the factor graph contains loops, pass messages indefinitely until convergence
- But convergence may not happen
- But in many cases loopy BP still works well, empirically

Outline

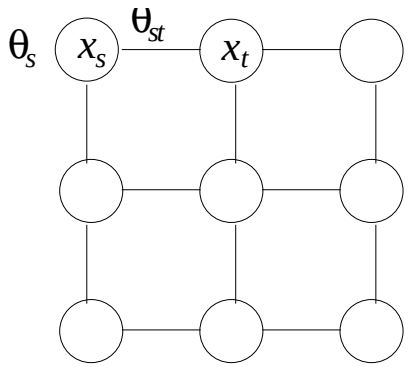
Example: The Ising Model



The random variables x take values in $\{0, 1\}$.

$$p_\theta(x) = \frac{1}{Z} \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t \right)$$

The Conditional



- Markovian: the conditional distribution for x_s is

$$p(x_s \mid x_{-s}) = p(x_s \mid x_{N(s)})$$

  $N(s)$ is the neighbors of s.

- This reduces to

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

- Gibbs sampling would draw x_s like this.

The Mean Field Algorithm for Ising Model

$$p(x_s = 1 \mid x_{N(s)}) = \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} x_t)) + 1}$$

- Instead of Gibbs sampling, let μ_s be the estimated marginal $p(x_s = 1)$

$$\mu_s \leftarrow \frac{1}{\exp(-(\theta_s + \sum_{t \in N(s)} \theta_{st} \mu_t)) + 1}$$

- The μ's are updated iteratively
- The Mean Field algorithm is coordinate ascent and guaranteed to converge to a local optimal (more later).

Outline

Exponential Family

- Let $\phi(X) = (\phi_1(X), \ldots, \phi_d(X))^\top$ be d *sufficient statistics*, where $\phi_i : \mathcal{X} \mapsto \mathbb{R}$
- Note X is all the nodes in a Graphical model
- $\phi_i(X)$ sometimes called a feature function
- Let $\theta = (\theta_1, \ldots, \theta_d)^\top \in \mathbb{R}^d$ be *canonical parameters*.
- The *exponential family* is a family of probability densities:

$$p_\theta(\mathbf{x}) = \exp\left(\theta^\top \phi(\mathbf{x}) - A(\theta)\right)$$

Exponential Family

$$p_\theta(\mathbf{x}) = \exp\left(\theta^\top \phi(\mathbf{x}) - A(\theta)\right)$$

- The key is the inner product between parameters θ and sufficient statistics ϕ.
- A is the log partition function,

$$A(\theta) = \log \int \exp\left(\theta^\top \phi(\mathbf{x})\right) \nu(d\mathbf{x})$$

- $A = \log Z$

Minimal vs. Overcomplete Models

- Parameters for which the density is normalizable:

$$\Omega = \{\theta \in \mathbb{R}^d \mid A(\theta) < \infty\}$$

- A *minimal* exponential family is where the ϕ's are linearly independent.
- An *overcomplete* exponential family is where the ϕ's are linearly dependent:

$$\exists \alpha \in \mathbb{R}^d,\ \alpha^\top \phi(\mathbf{x}) = \text{constant } \forall \mathbf{x}$$

- Both minimal and overcomplete representations are useful.

Exponential Family Example 1: Bernoulli

$p(x) = \beta^x(1-\beta)^{1-x}$ for $x \in \{0, 1\}$ and $\beta \in (0, 1)$.

- Does not look like an exponential family!
- Can be rewritten as

$$p(x) = \exp\left(x \log \beta + (1-x) \log(1-\beta)\right)$$

- Now in exponential family form with $\phi_1(x) = x, \phi_2(x) = 1 - x$, $\theta_1 = \log \beta, \theta_2 = \log(1-\beta)$, and $A(\theta) = 0$.
- Overcomplete: $\alpha_1 = \alpha_2 = 1$ makes $\alpha^\top \phi(x) = 1$ for all x

Exponential Family Example 1: Bernoulli

$$p(x) = \exp\left(x \log \beta + (1 - x) \log(1 - \beta)\right)$$

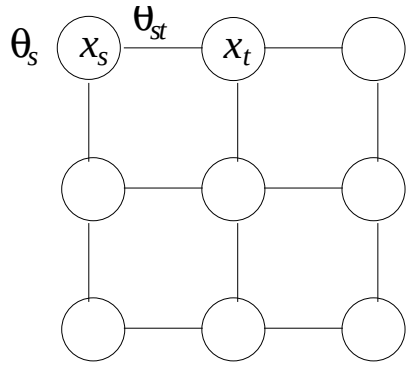
- Can be further rewritten as

$$p(x) = \exp\left(x\theta - \log(1 + \exp(\theta))\right)$$

- Minimal exponential family with
  $\phi(x) = x, \theta = \log \frac{\beta}{1-\beta}, A(\theta) = \log(1 + \exp(\theta))$.

Many distributions (e.g., Gaussian, exponential, Poisson, Beta) are in the exponential family, but not all (e.g., the Laplace distribution).

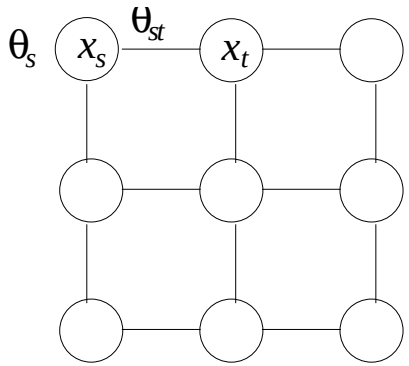
Exponential Family Example 2: Ising Model



$$p_\theta(\mathbf{x}) = \exp\left(\sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t - A(\theta)\right)$$

- Binary random variable $x_s \in \{0, 1\}$
- $d = |V| + |E|$ sufficient statistics: $\phi(\mathbf{x}) = (\ldots x_s \ldots x_{st} \ldots)^\top$
- This is a regular ($\Omega = \mathbb{R}^d$), minimal exponential family.

Exponential Family Example 3: Potts Model



- Similar to Ising model but generalizing $x_s \in \{0, \ldots, r-1\}$.
- Indicator functions $f_{sj}(\mathbf{x}) = 1$ if $x_s = j$ and 0 otherwise, and $f_{stjk}(\mathbf{x}) = 1$ if $x_s = j \wedge x_t = k$, and 0 otherwise.

$$p_\theta(\mathbf{x}) = \exp\left(\sum_{sj} \theta_{sj} f_{sj}(\mathbf{x}) + \sum_{stjk} \theta_{stjk} f_{stjk}(\mathbf{x}) - A(\theta)\right)$$

- $d = r|V| + r^2|E|$
- Regular but overcomplete, because $\sum_{j=0}^{r-1} \theta_{sj}(\mathbf{x}) = 1$ for any $s \in V$ and all $\mathbf{x}$.
- The Potts model is a special case where the parameters are tied: $\theta_{stkk} = \alpha$, and $\theta_{stjk} = \beta$ for $j \neq k$.

Important Relation

For sufficient statistics defined by indicator functions

- e.g., $\phi_{sj}(\mathbf{x}) = f_{sj}(\mathbf{x}) = 1$ if $x_s = j$ and 0 otherwise
- The marginal can be obtained via the mean

$$\mathbb{E}_\theta[\phi_{sj}(\mathbf{x})] = P(x_s = j)$$

- Since inference is about computing the marginal, in this case it is equivalent to computing the mean.

Mean Parameters

- Let p be any density (not necessarily in exponential family).
- Given sufficient statistics ϕ, the *mean parameters* $\mu = (\mu_1, \ldots, \mu_d)^\top$ is

$$\mu_i = \mathbb{E}_p[\phi_i(\mathbf{x})] = \int \phi_i(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$$

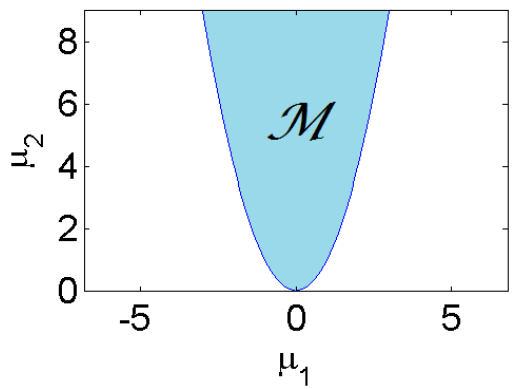
- The set of mean parameters

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

- If $\mu^{(1)}, \mu^{(2)} \in \mathcal{M}$, there must exist $p^{(1)}, p^{(2)}$
- The convex combinations of $p^{(1)}, p^{(2)}$ leads to another mean parameter in $\mathcal{M}$
- Therefore $\mathcal{M}$ is convex

Example: The First Two Moments

- Let $\phi_1(x) = x, \phi_2(x) = x^2$
- For any p (not necessarily Gaussian) on x, the mean parameters $\mu = (\mu_1, \mu_2) = (\mathbb{E}(x), \mathbb{E}(x^2))^\top$.
- Note $\mathbb{V}(x) = \mathbb{E}(x^2) - \mathbb{E}^2(x) = \mu_2 - \mu_1^2 \geq 0$ for any p
- $\mathcal{M}$ is not $\mathbb{R}^2$ but rather the subset $\mu_1 \in \mathbb{R}, \mu_2 \geq \mu_1^2$.



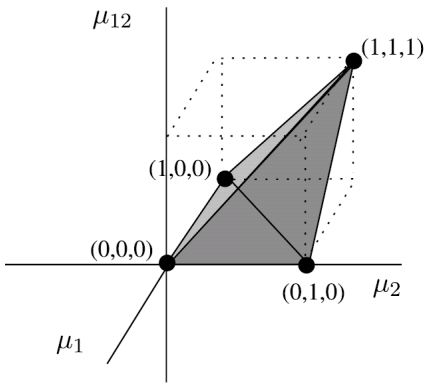
The Marginal Polytope

- The *marginal polytope* is defined for discrete x_s
- Recall $\mathcal{M} = \{\mu \in \mathbb{R}^d \mid \mu = \sum_{\mathbf{x}} \phi(\mathbf{x}) p(\mathbf{x}) \text{ for some } p\}$
- p can be a point mass function on a particular $\mathbf{x}$.
- In fact any p is a convex combination of such point mass functions.
- $\mathcal{M} = conv\{\phi(\mathbf{x}), \forall \mathbf{x}\}$ is a convex hull, called the marginal polytope.

Marginal Polytope Example

Tiny Ising model: two nodes $x_1, x_2 \in \{0, 1\}$ connected by an edge.

- minimal sufficient statistics $\phi(x_1, x_2) = (x_1, x_2, x_1 x_2)^\top$.
- only 4 different $\mathbf{x} = (x_1, x_2)$.
- the marginal polytope is $\mathcal{M} = conv\{(0,0,0), (0,1,0), (1,0,0), (1,1,1)\}$



- the convex hull is a polytope inside the unit cube.
- the three coordinates are node marginals $\mu_1 \equiv \mathbb{E}_p[x_1 = 1]$, $\mu_2 \equiv \mathbb{E}_p[x_2 = 1]$ and edge marginal $\mu_{12} \equiv \mathbb{E}_p[x_1 = x_2 = 1]$, hence the name.

The Log Partition Function A

- For any regular exponential family, $A(\theta)$ is convex in θ.
- Strictly convex for minimal exponential family.
- Nice property: $\frac{\partial A(\theta)}{\partial \theta_i} = \mathbb{E}_\theta[\phi_i(\mathbf{x})]$
- Therefore, $\nabla A = \mu$, the mean parameters of p_θ.

Conjugate Duality

The *conjugate dual function* A^* to A is defined as

$$A^*(\mu) = \sup_{\theta \in \Omega} \theta^\top \mu - A(\theta)$$

Such definition, where a quantity is expressed as the solution to an optimization problem, is called a *variational* definition.

- For any $\mu \in \mathcal{M}$'s interior, let $\theta(\mu)$ satisfy $\mathbb{E}_{\theta(\mu)}[\phi(\mathbf{x})] = \nabla A(\theta(\mu)) = \mu$.
- Then $A^*(\mu) = -H(p_{\theta(\mu)})$ the negative entropy.
- The dual of the dual gives back A: $A(\theta) = \sup_{\mu \in \mathcal{M}} \mu^\top \theta - A^*(\mu)$
- For all $\theta \in \Omega$, the supremum is attained uniquely at the $\mu \in \mathcal{M}^0$ by the moment matching conditions $\mu = \mathbb{E}_\theta[\phi(\mathbf{x})]$.

Example: Conjugate Dual for Bernoulli

- Recall the minimal exponential family for Bernoulli with $\phi(x) = x, A(\theta) = \log(1 + \exp(\theta)), \Omega = \mathbb{R}$.
- By definition

$$A^*(\mu) = \sup_{\theta \in \mathbb{R}} \theta\mu - \log(1 + \exp(\theta))$$

- Taking derivative and solve

$$A^*(\mu) = \mu \log \mu + (1 - \mu) \log(1 - \mu)$$

i.e., the negative entropy.

Inference with Variational Representation

$A(\theta) = \sup_{\mu \in \mathcal{M}} \mu^\top \theta - A^*(\mu)$ is attained by $\mu = \mathbb{E}_\theta[\phi(\mathbf{x})]$.

- Want to compute the marginals $P(x_s = j)$? They are the mean parameters $\mu_{sj} = \mathbb{E}_\theta[\phi_{ij}(\mathbf{x})]$ under standard overcomplete representation.
- Want to compute the mean parameters μ_{sj}? They are the $\arg\sup$ to the optimization problem above.
- This variational representation is exact, not approximate (will relax it next to derive loopy BP and mean field)

The Difficulties with Variational Representation

$$A(\theta) = \sup_{\mu \in \mathcal{M}} \mu^\top \theta - A^*(\mu)$$

- Difficult to solve even though it is a convex problem
- Two issues:
  - Although the marginal polytope $\mathcal{M}$ is convex, it can be quite complex (exponential number of vertices)
  - The dual function $A^*(\mu)$ usually does not admit an explicit form.
- Variational approximation modifies the optimization problem so that it is tractable, at the price of an approximate solution.
- Next, we cast mean field and sum-product algorithms as variational approximations.

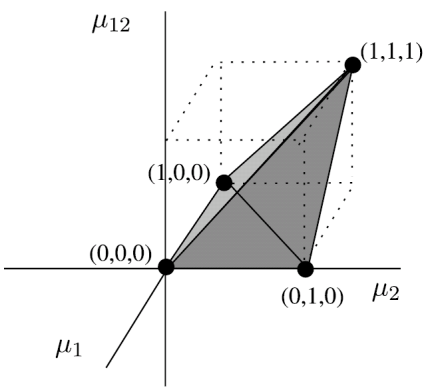
The Mean Field Method as Variational Approximation

- The mean field method replaces $\mathcal{M}$ with a simpler subset $\mathcal{M}(F)$ on which $A^*(\mu)$ has a closed form.
- Consider the fully disconnected subgraph $F = (V, \emptyset)$ of the original graph $G = (V, E)$
- Set all $\theta_i = 0$ if ϕ_i involves edges
- The densities in this sub-family are all fully factorized:

$$p_\theta(\mathbf{x}) = \prod_{s \in V} p(x_s; \theta_s)$$

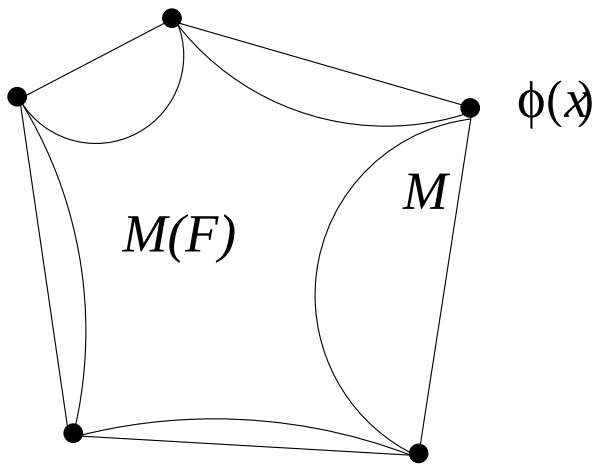
The Geometry of $\mathcal{M}(F)$

- Let $\mathcal{M}(F)$ be the mean parameters of the fully factorized sub-family. In general, $\mathcal{M}(F) \subset \mathcal{M}$
- Recall $\mathcal{M}$ is the convex hull of extreme points $\{\phi(\mathbf{x})\}$.
- It turns out the extreme points $\{\phi(\mathbf{x})\} \in \mathcal{M}(F)$.
- Example:
  - The tiny Ising model $x_1, x_2 \in \{0, 1\}$ with $\phi = (x_1, x_2, x_1 x_2)^\top$
  - The point mass distribution $p(\mathbf{x} = (0, 1)^\top) = 1$ is realized as a limit to the series $p(\mathbf{x}) = \exp(\theta_1 x_1 + \theta_2 x_2 - A(\theta))$ where $\theta_1 \to -\infty$ and $\theta_2 \to \infty$.
  - This series is in F because $\theta_{12} = 0$.
  - Hence the extreme point $\phi(\mathbf{x}) = (0, 1, 0)$ is in $\mathcal{M}(F)$.



The Geometry of $\mathcal{M}(F)$

- Because the extreme points of $\mathcal{M}$ are in $\mathcal{M}(F)$, if $\mathcal{M}(F)$ were convex, we would have $\mathcal{M} = \mathcal{M}(F)$.
- But in general $\mathcal{M}(F)$ is a true subset of $\mathcal{M}$
- Therefore, $\mathcal{M}(F)$ is a nonconvex inner set of $\mathcal{M}$



The Mean Field Method as Variational Approximation

- Recall the exact variational problem

$$A(\theta) = \sup_{\mu \in \mathcal{M}} \mu^\top \theta - A^*(\mu)$$

  attained by solution to inference problem $\mu = \mathbb{E}_\theta[\phi(\mathbf{x})]$.
- The mean field method simply replaces $\mathcal{M}$ with $\mathcal{M}(F)$

$$\mathcal{L}(\theta) = \sup_{\mu \in \mathcal{M}(F)} \mu^\top \theta - A^*(\mu)$$

- Obvious $\mathcal{L}(\theta) \leq A(\theta)$.
- The original solution μ^* may not be in $\mathcal{M}(F)$
- Even if $\mu^* \in \mathcal{M}(F)$, may hit local maximum and not find it
- Why both? Because $A^*(\mu) = -H(p_\theta(\mu))$ has a very simple form for $M(F)$

Example: Mean Field for Ising Model

- The mean parameters for the Ising model are the node and edge marginals: $\mu_s = p(x_x = 1), \mu_{st} = p(x_s = 1, x_t = 1)$
- Fully factorized $\mathcal{M}(F)$ means no edge. $\mu_{st} = \mu_s \mu_t$
- For $\mathcal{M}(F)$, the dual function $A^*(\mu)$ has the simple form

$$A^*(\mu) = \sum_{s \in V} -H(\mu_s) = \sum_{s \in V} \mu_s \log \mu_s + (1 - \mu_s) \log(1 - \mu_s)$$

- Thus the mean field problem is

$$\begin{aligned} \mathcal{L}(\theta) &= \sup_{\mu \in \mathcal{M}(F)} \mu^\top \theta - \sum_{s \in V} (\mu_s \log \mu_s + (1 - \mu_s) \log(1 - \mu_s)) \\ &= \max_{(\mu_1 \ldots \mu_m) \in [0,1]^m} \left(\sum_{s \in V} \theta_s \mu_s + \sum_{(s,t) \in E} \theta_{st} \mu_s \mu_t + \sum_{s \in V} H(\mu_s) \right) \end{aligned}$$

Example: Mean Field for Ising Model

$$\mathcal{L}(\theta) = \max_{(\mu_1 \ldots \mu_m) \in [0,1]^m} \left(\sum_{s \in V} \theta_s \mu_s + \sum_{(s,t) \in E} \theta_{st} \mu_s \mu_t + \sum_{s \in V} H(\mu_s) \right)$$

- Bilinear in μ, not jointly concave
- But concave in a single dimension μ_s, fixing others.
- Iterative coordinate-wise maximization: fixing μ_t for $t \neq s$ and optimizing μ_s.
- Setting the partial derivative w.r.t. μ_s to 0 yields:

$$\mu_s = \frac{1}{1 + \exp\left(-(\theta_s + \sum_{(s,t) \in E} \theta_{st} \mu_t)\right)}$$

  as we've seen before.
- Caution: mean field converges to a local maximum depending on the initialization of $\mu_1 \ldots \mu_m$.

The Sum-Product Algorithm as Variational Approximation

$$A(\theta) = \sup_{\mu \in \mathcal{M}} \mu^\top \theta - A^*(\mu)$$

The sum-product algorithm makes two approximations:

- it relaxes $\mathcal{M}$ to an *outer* set L
- it replaces the dual A^* with an approximation.

$$A(\theta) = \sup_{\mu \in L} \mu^\top \theta - \tilde{A}^*(\mu)$$

The Outer Relaxation

- For overcomplete exponential families on discrete nodes, the mean parameters are node and edge marginals $\mu_{sj} = p(x_s = j), \mu_{stjk} = p(x_s = j, x_t = k)$.
- The marginal polytope is $\mathcal{M} = \{\mu \mid \exists p \text{ with marginals } \mu\}$.
- Now consider $\tau \in \mathbb{R}_+^d$ satisfying "node normalization" and "edge-node marginal consistency" conditions:

$$\sum_{j=0}^{r-1} \tau_{sj} = 1 \qquad \forall s \in V$$

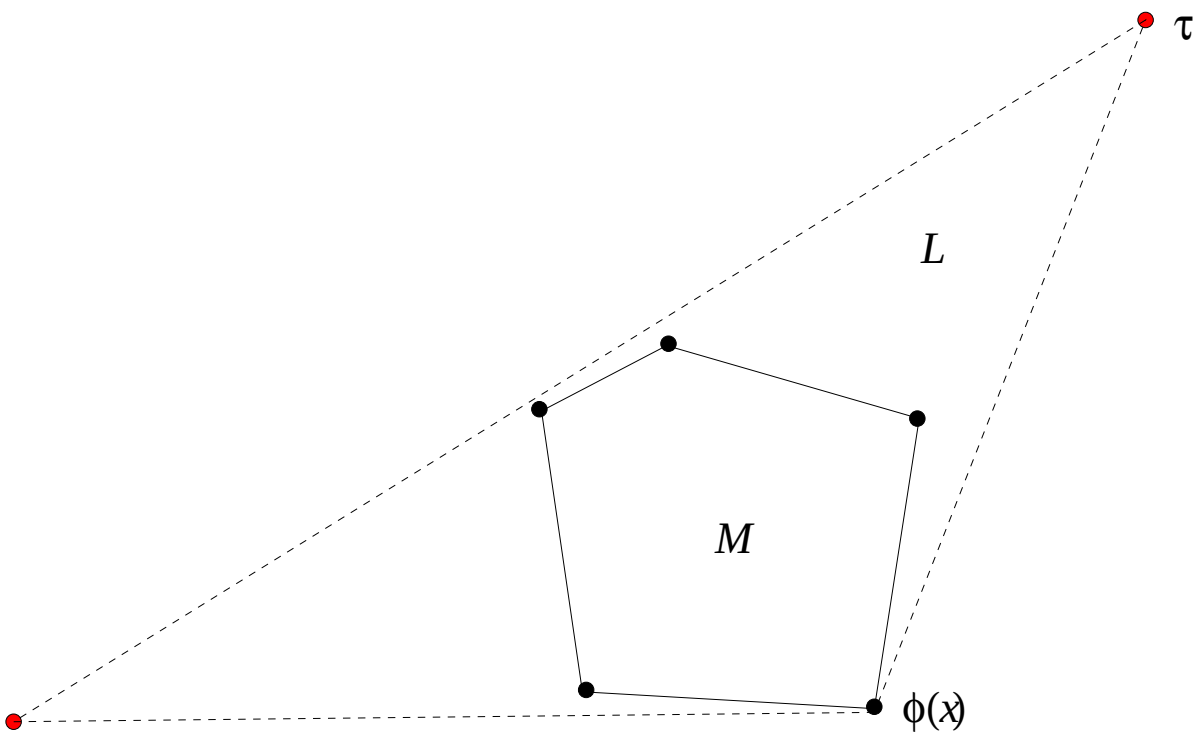
$$\sum_{k=0}^{r-1} \tau_{stjk} = \tau_{sj} \qquad \forall s, t \in V, j = 0 \ldots r-1$$

$$\sum_{j=0}^{r-1} \tau_{stjk} = \tau_{tk} \qquad \forall s, t \in V, k = 0 \ldots r-1$$

- Define $L = \{\tau \text{ satisfying the above conditions}\}$.

The Outer Relaxation

- If the graph is a tree then $\mathcal{M} = L$
- If the graph has cycles then $\mathcal{M} \subset L$
  - L is too lax to satisfy other constraints that true marginals need to satisfy
- Nice property: L is still a polytope, but much simpler than $\mathcal{M}$.



The first approximation in sum-product is to replace $\mathcal{M}$ with L.

Approximating A^*

- Recall μ are node and edge marginals
- If the graph is a tree, one can exactly reconstruct the *joint* probability

$$p_\mu(\mathbf{x}) = \prod_{s \in V} \mu_{s x_s} \prod_{(s,t) \in E} \frac{\mu_{st x_s x_t}}{\mu_{s x_s} \mu_{t x_t}}$$

- If the graph is a tree, the entropy of the joint distribution is

$$H(p_\mu) = -\sum_{s \in V} \sum_{j=0}^{r-1} \mu_{sj} \log \mu_{sj} - \sum_{(s,t) \in E} \sum_{j,k} \mu_{stjk} \log \frac{\mu_{stjk}}{\mu_{sj} \mu_{tk}}$$

- Neither holds for graph with cycles.

Approximating A^*

Define the *Bethe entropy* for $\tau \in L$ on loopy graphs in the same way:

$$H_{Bethe}(p_\tau) = -\sum_{s \in V} \sum_{j=0}^{r-1} \tau_{sj} \log \tau_{sj} - \sum_{(s,t) \in E} \sum_{j,k} \tau_{stjk} \log \frac{\tau_{stjk}}{\tau_{sj}\tau_{tk}}$$

Note H_{Bethe} is not a true entropy. The second approximation in sum-product is to replace $A^*(\tau)$ with $-H_{Bethe}(p_\tau)$.

The Sum-Product Algorithm as Variational Approximation

With these two approximations, we arrive at the variational problem

$$\tilde{A}(\theta) = \sup_{\tau \in L} \tau^\top \theta + H_{Bethe}(p_\tau)$$

- Optimality conditions require the gradients vanish w.r.t. both τ and the Lagrangian multipliers on constraints $\tau \in L$.
- The sum-product algorithm can be derived as an iterative fixed point procedure to satisfy optimality conditions.
- At the solution, $\tilde{A}(\theta)$ is not guaranteed to be either an upper or a lower bound of $A(\theta)$
- τ may not correspond to a true marginal distribution

Summary: Variational Inference

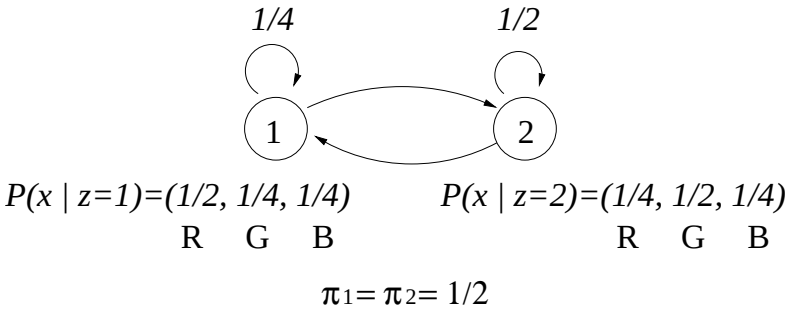
- The sum-product algorithm (loopy belief propagation)
- The mean field method
- Not covered: Expectation Propagation

Efficient computation. But often *unknown* bias in solution.

Outline

Maximizing Problems

Recall the HMM example



There are two senses of "best states" $z_{1:N}$ given $x_{1:N}$:

1. So far we computed the marginal $p(z_n|x_{1:N})$
   - We can define "best" as $z_n^* = \arg\max_k p(z_n = k|x_{1:N})$
   - However $z^*_{1:N}$ as a whole may not be the best
   - In fact $z^*_{1:N}$ can even have zero probability!
2. An alternative is to find

$$z^*_{1:N} = \arg\max_{z_{1:N}} p(z_{1:N}|x_{1:N})$$

   - finds the most likely *state configuration* as a whole
   - The max-sum algorithm solves this
   - Generalizes the Viterbi algorithm for HMMs

Intermediate: The Max-Product Algorithm

Simple modification to the sum-product algorithm: replace $\sum$ with $\max$ in the factor-to-variable messages.

$$\mu_{f_s \to x}(x) = \max_{x_1} \ldots \max_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m)$$

$$\mu_{x_m \to f_s}(x_m) = \prod_{f \in ne(x_m) \setminus f_s} \mu_{f \to x_m}(x_m)$$

$$\mu_{x_{\text{leaf}} \to f}(x) = 1$$

$$\mu_{f_{\text{leaf}} \to x}(x) = f(x)$$

Intermediate: The Max-Product Algorithm

- As in sum-product, pick an arbitrary variable node x as the root
- Pass messages up from leaves until they reach the root
- Unlike sum-product, do not pass messages back from root to leaves
- At the root, multiply incoming messages

$$p^{\max} = \max_x \left(\prod_{f \in ne(x)} \mu_{f \to x}(x) \right)$$

- This is the probability of the most likely state configuration

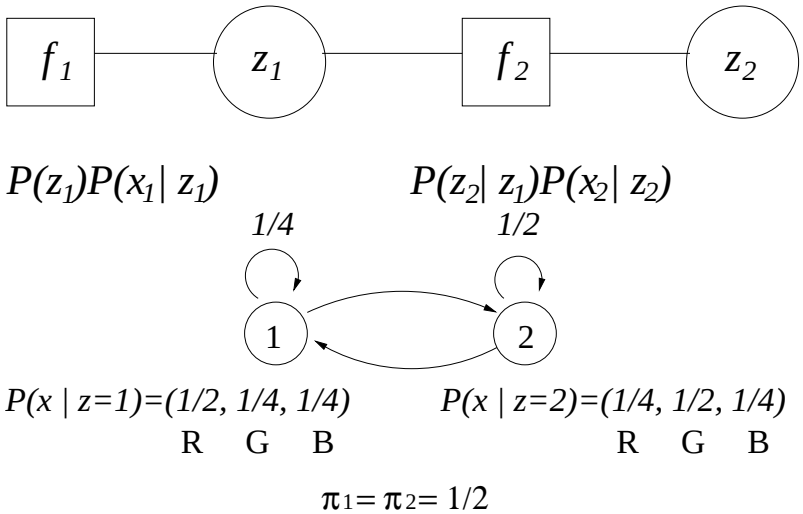
Intermediate: The Max-Product Algorithm

- To identify the configuration itself, keep *back pointers*:
- When creating the message

$$\mu_{f_s \to x}(x) = \max_{x_1} \ldots \max_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m=1}^{M} \mu_{x_m \to f_s}(x_m)$$

  for each x value, we separately create M pointers back to the values of $x_1, \ldots, x_M$ that achieve the maximum.
- At the root, backtrack the pointers.

Intermediate: The Max-Product Algorithm



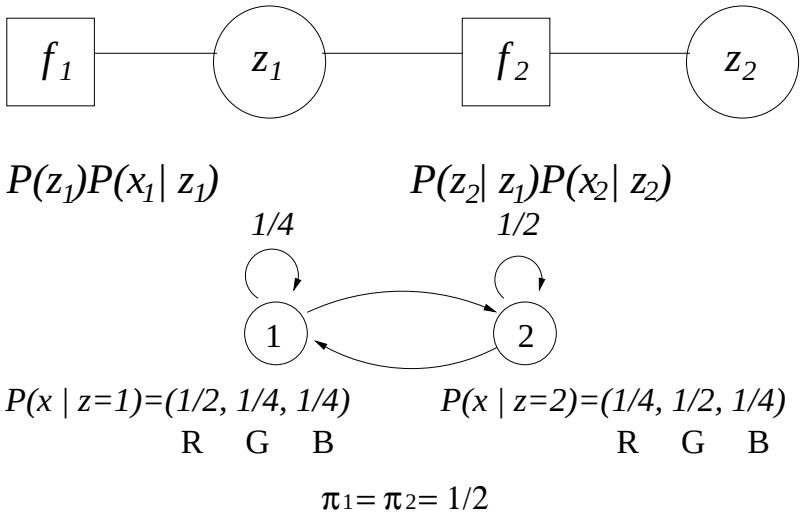
- Message from leaf f_1

$$\mu_{f_1 \to z_1}(z_1 = 1) = P(z_1 = 1)P(R|z_1 = 1) = 1/2 \cdot 1/2 = 1/4$$
$$\mu_{f_1 \to z_1}(z_1 = 2) = P(z_1 = 2)P(R|z_1 = 2) = 1/2 \cdot 1/4 = 1/8$$

- The second message

$$\mu_{z_1 \to f_2}(z_1 = 1) = 1/4$$
$$\mu_{z_1 \to f_2}(z_1 = 2) = 1/8$$

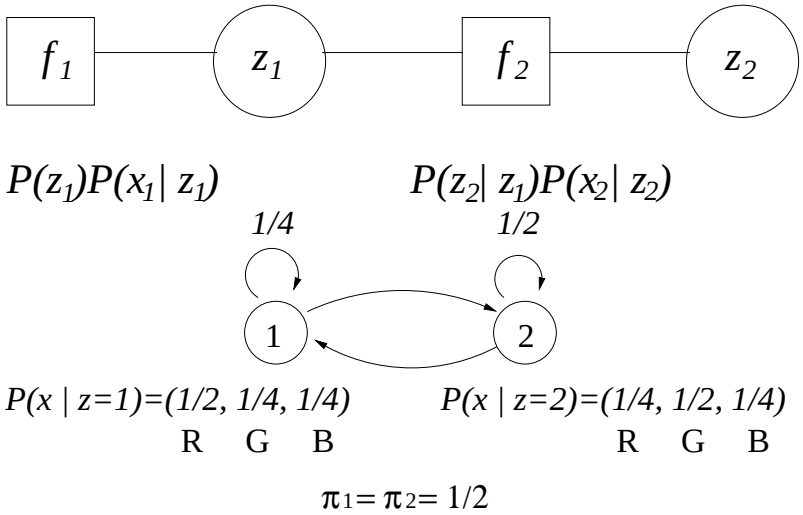
Intermediate: The Max-Product Algorithm



$$
\begin{aligned}
& \mu_{f_2 \to z_2}(z_2 = 1) \\
= \quad & \max_{z_1} f_2(z_1, z_2)\mu_{z_1 \to f_2}(z_1) \\
= \quad & \max_{z_1} P(z_2 = 1 \mid z_1)P(x_2 = G \mid z_2 = 1)\mu_{z_1 \to f_2}(z_1) \\
= \quad & \max(1/4 \cdot 1/4 \cdot 1/4, 1/2 \cdot 1/4 \cdot 1/8) = 1/64
\end{aligned}
$$

Back pointer for $z_2 = 1$: either $z_1 = 1$ or $z_1 = 2$

Intermediate: The Max-Product Algorithm

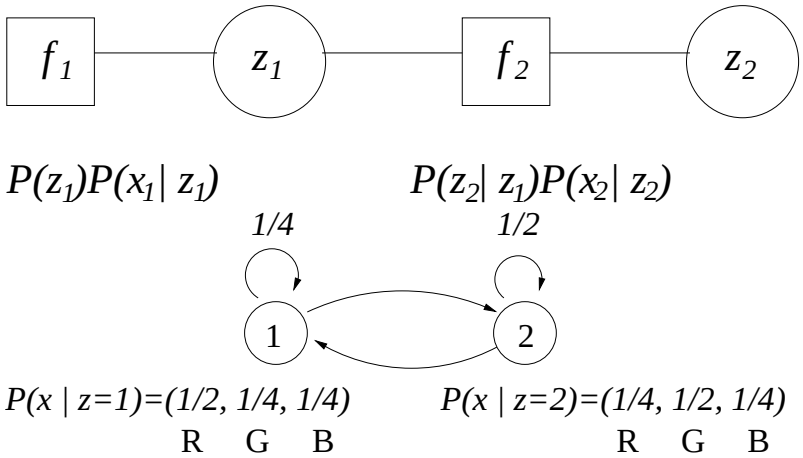


The other element of the same message:

$$
\begin{aligned}
& \mu_{f_2 \to z_2}(z_2 = 2) \\
= \quad & \max_{z_1} f_2(z_1, z_2)\mu_{z_1 \to f_2}(z_1) \\
= \quad & \max_{z_1} P(z_2 = 2 \mid z_1)P(x_2 = G \mid z_2 = 2)\mu_{z_1 \to f_2}(z_1) \\
= \quad & \max(3/4 \cdot 1/2 \cdot 1/4, 1/2 \cdot 1/2 \cdot 1/8) = 3/32
\end{aligned}
$$

Back pointer for $z_2 = 2$: $z_1 = 1$

Intermediate: The Max-Product Algorithm



$\pi_1 = \pi_2 = 1/2$

$$\mu_{f_2 \to z_2} = \begin{pmatrix} 1/64 & \to z_1 = 1,2 \\ 3/32 & \to z_1 = 1 \end{pmatrix}$$

At root z_2,

$$\max_{s=1,2} \mu_{f_2 \to z_2}(s) = 3/32$$

$$z_2 = 2 \to z_1 = 1$$

$$z^*_{1:2} = \arg\max_{z_{1:2}} p(z_{1:2}|x_{1:2}) = (1, 2)$$

In this example, sum-product and max-product produce the same best sequence; In general they differ.

From Max-Product to Max-Sum

The *max-sum algorithm* is equivalent to the max-product algorithm, but work in $\log$ space to avoid underflow.

$$\begin{aligned}
\mu_{f_s \to x}(x) &= \max_{x_1 \ldots x_M} \log f_s(x, x_1, \ldots, x_M) + \sum_{m=1}^{M} \mu_{x_m \to f_s}(x_m) \\
\mu_{x_m \to f_s}(x_m) &= \sum_{f \in ne(x_m) \setminus f_s} \mu_{f \to x_m}(x_m) \\
\mu_{x_{\text{leaf}} \to f}(x) &= 0 \\
\mu_{f_{\text{leaf}} \to x}(x) &= \log f(x)
\end{aligned}$$

When at the root,

$$\log p^{\max} = \max_x \left(\sum_{f \in ne(x)} \mu_{f \to x}(x) \right)$$

The back pointers are the same.

Outline

Parameter Learning

- Assume the graph structure is given
- Learning in exponential family: estimate θ from iid data $\mathbf{x}_1 \ldots \mathbf{x}_n$.
- Principle: maximum likelihood
- Distinguish two cases:
  - *fully observed data*: all dimensions of $\mathbf{x}$ are observed
  - *partially observed data*: some dimensions of $\mathbf{x}$ are unobserved.

Fully Observed Data

$$p_\theta(\mathbf{x}) = \exp\left(\theta^\top \phi(\mathbf{x}) - A(\theta)\right)$$

- Given iid data $\mathbf{x}_1 \ldots \mathbf{x}_n$, the log likelihood is

$$\ell(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i) = \theta^\top \left(\frac{1}{n}\sum_{i=1}^{n} \phi(\mathbf{x}_i)\right) - A(\theta) = \theta^\top \hat{\mu} - A(\theta)$$

- $\hat{\mu} \equiv \frac{1}{n}\sum_{i=1}^{n} \phi(\mathbf{x}_i)$ is the mean parameter of the empirical distribution on $\mathbf{x}_1 \ldots \mathbf{x}_n$. Clearly $\hat{\mu} \in \mathcal{M}$.
- Maximum likelihood: $\theta^{ML} = \arg\sup_{\theta \in \Omega} \theta^\top \hat{\mu} - A(\theta)$
- The solution is $\theta^{ML} = \theta(\hat{\mu})$, the exponential family density whose mean parameter matches $\hat{\mu}$.
- When $\hat{\mu} \in \mathcal{M}^0$ and ϕ minimal, there is a unique maximum likelihood solution θ^{ML}.

Partially Observed Data

- Each item $(\mathbf{x}, \mathbf{z})$ where $\mathbf{x}$ observed, $\mathbf{z}$ unobserved
- Full data $(\mathbf{x}_1, \mathbf{z}_1) \ldots (\mathbf{x}_n, \mathbf{z}_n)$, but we only observe $\mathbf{x}_1 \ldots \mathbf{x}_n$
- The incomplete likelihood $\ell(\theta) = \frac{1}{n} \sum_{i=1}^{n} \log p_\theta(\mathbf{x}_i)$ where $p_\theta(\mathbf{x}_i) = \int p_\theta(\mathbf{x}_i, \mathbf{z}) d\mathbf{z}$
- Can be written as $\ell(\theta) = \frac{1}{n} \sum_{i=1}^{n} A_{\mathbf{x}_i}(\theta) - A(\theta)$
- New log partition function of $p_\theta(\mathbf{z} \mid \mathbf{x}_i)$, one per item:

$$A_{\mathbf{x}_i}(\theta) = \log \int \exp(\theta^\top \phi(\mathbf{x}_i, \mathbf{z}')) d\mathbf{z}'$$

- Expectation-Maximization (EM) algorithm: lower bound $A_{\mathbf{x}_i}$

EM as Variational Lower Bound

- Mean parameter realizable by any distribution on $\mathbf{z}$ while holding $\mathbf{x}_i$ fixed:
  $\mathcal{M}_{\mathbf{x}_i} = \{\mu \in \mathbb{R}^d \mid \mu = \mathbb{E}_p[\phi(\mathbf{x}_i, \mathbf{z})] \text{ for some } p\}$
- The variational definition $A_{\mathbf{x}_i}(\theta) = \sup_{\mu \in \mathcal{M}_{\mathbf{x}_i}} \theta^\top \mu - A^*_{\mathbf{x}_i}(\mu)$
- Trivial variational lower bound:
  $A_{\mathbf{x}_i}(\theta) \geq \theta^\top \mu^i - A^*_{\mathbf{x}_i}(\mu^i), \forall \mu^i \in \mathcal{M}_{\mathbf{x}_i}$
- Lower bound $\mathcal{L}$ on the incomplete log likelihood:

$$\begin{aligned}
\ell(\theta) &= \frac{1}{n} \sum_{i=1}^{n} A_{\mathbf{x}_i}(\theta) - A(\theta) \\
&\geq \frac{1}{n} \sum_{i=1}^{n} \left(\theta^\top \mu^i - A^*_{\mathbf{x}_i}(\mu^i) \right) - A(\theta) \\
&\equiv \mathcal{L}(\mu^1, \ldots, \mu^n, \theta)
\end{aligned}$$

Exact EM: The E-Step

The EM algorithm is coordinate ascent on $\mathcal{L}(\mu^1, \ldots, \mu^n, \theta)$.

- In the E-step, maximizes each μ^i

$$\mu^i \leftarrow \arg \max_{\mu^i \in \mathcal{M}_{\mathbf{x}_i}} \mathcal{L}(\mu^1, \ldots, \mu^n, \theta)$$

- Equivalently, $\operatorname{argmax}_{\mu^i \in \mathcal{M}_{\mathbf{x}_i}} \theta^\top \mu^i - A^*_{\mathbf{x}_i}(\mu^i)$
- This is the variational representation of the mean parameter $\mu^i(\theta) = \mathbb{E}_\theta[\phi(\mathbf{x}_i, \mathbf{z})]$
- The E-step is named after this $\mathbb{E}_\theta[]$ under the current parameters θ

Exact EM: The M-Step

- In the M-step, maximize θ holding the μ's fixed:

$$\theta \leftarrow \arg\max_{\theta \in \Omega} \mathcal{L}(\mu^1, \ldots, \mu^n, \theta) = \arg\max_{\theta \in \Omega} \theta^\top \hat{\mu} - A(\theta)$$

- $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mu^i$
- The solution $\theta(\hat{\mu})$ satisfies $\mathbb{E}_{\theta(\hat{\mu})}[\phi(\mathbf{x})] = \hat{\mu}$
- Standard fully observed maximum likelihood problem, hence the name M-step

Variational EM

For loopy graphs E-step often intractable.

- Can't maximize

$$\max_{\mu^i \in \mathcal{M}_{\mathbf{x}_i}} \theta^\top \mu^i - A^*_{\mathbf{x}_i}(\mu^i)$$

- Improve but not necessarily maximize: "generalized EM"
- The mean field method maximizes

$$\max_{\mu^i \in \mathcal{M}_{\mathbf{x}_i}(F)} \theta^\top \mu^i - A^*_{\mathbf{x}_i}(\mu^i)$$

  - up to local maximum
  - recall $\mathcal{M}_{\mathbf{x}_i}(F)$ is an inner approximation to $\mathcal{M}_{\mathbf{x}_i}$
- Mean field E-step leads to generalized EM
- The sum-product algorithm does not lead to generalized EM

Outline

Score-Based Structure Learning

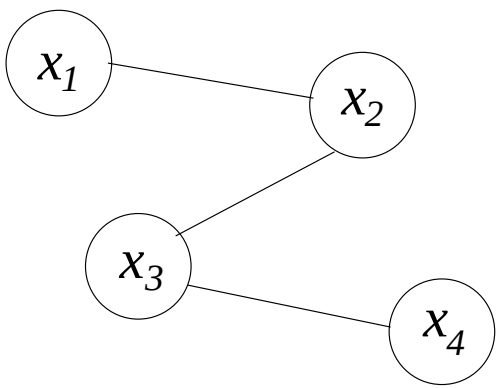
- Let $\mathcal{M}$ be all allowed candidate features
- Let $M \subseteq \mathcal{M}$ be a log-linear model structure

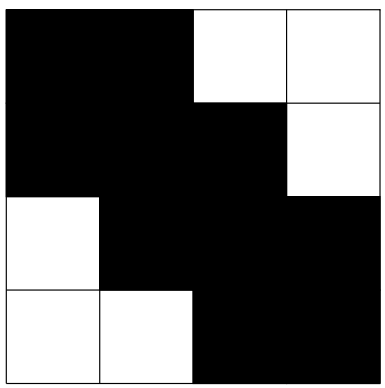
$$P(X \mid M, \theta) = \frac{1}{Z} \exp\left(\sum_{i \in M} \theta_i f_i(X)\right)$$

- A score for the model M can be $\max_\theta \ln P(\text{Data} \mid M, \theta)$
- The score is always better for larger M – needs regularization
- M and θ treated separately

Structure Learning for Gaussian Random Fields

- Consider a p-dimensional multivariate Gaussian $N(\mu, \Sigma)$
- The graphical model has p nodes $x_1, \ldots, x_p$
- The edge between x_i, x_j is absent if and only if $\Omega_{ij} = 0$, where $\Omega = \Sigma^{-1}$
- Equivalently, x_i, x_j are conditionally independent given other variables





Structure Learning for Gaussian Random Fields

- Let data be $X^{(1)}, \ldots, X^{(n)} \sim N(\mu, \Sigma)$
- The log likelihood is $\frac{n}{2} \log |\Omega| - \frac{1}{2} \sum_{i=1}^{n} (X^{(i)} - \mu)^\top \Omega (X^{(i)} - \mu)$
- The maximum likelihood estimate of Σ is the sample covariance

$$S = \frac{1}{n} \sum_{i} (X^{(i)} - \bar{X})^\top (X^{(i)} - \bar{X})$$

  where $\bar{X}$ is the sample mean
- S^{-1} is not a good estimate of Ω when n is small

Structure Learning for Gaussian Random Fields

- For centered data, minimize a regularized problem instead:

$$-\log|\Omega| + \frac{1}{n}\sum_{i=1}^{n} {X^{(i)}}^{\top}\Omega X^{(i)} + \lambda\sum_{i\neq j}|\Omega_{ij}|$$

- Known as glasso

Recap

- Given GM = joint distribution $p(x_1, \ldots, x_n)$
  - BN or MRF
  - conditional independence
- Do inference = $p(X_Q \mid X_E)$, in general $X_Q \cup X_E \subset \{x_1 \ldots x_n\}$
  - exact, MCMC, variational
- If $p(x_1, \ldots, x_n)$ not given, estimate it from data
  - parameter and structure learning

Much on-going research!