

# Today's lecture

Approximate inference in graphical models.

- Forward and Backward KL divergence
- Variational Inference
- Mean Field: Naive and Structured
- Marginal Polytope
- Local Polytope
- Relaxation methods
- Loopy BP
- LP relaxations for MAP inference

Figures from D. Sontag, Murphy's book

# Approximate marginal inference

- Given the joint  $p(x_1, \dots, x_n)$  represented as a graphical model, we want to perform **marginal inference**, e.g.,  $p(x_i|e)$
- We showed in last lecture that doing this exactly is NP-hard
- We also covered variable elimination (VE), which can solve these type of queries for any graphical model, but ...
- Almost all approximate inference algorithms in practice are
  - Variational algorithms (e.g., mean-field, loopy belief propagation)
  - Sampling methods (e.g., Gibbs sampling, MCMC)

- **Goal:** Approximate a difficult distribution  $p(\mathbf{x}|\mathbf{e})$  with a new distribution  $q(\mathbf{x})$ 
  - $p(\mathbf{x}|\mathbf{e})$  and  $q(\mathbf{x})$  should be "close"
  - Computation on  $q(\mathbf{x})$  should be easy
- How should we measure distance between distributions?
- The **Kullback-Leibler divergence** (KL-divergence) between two distributions  $p$  and  $q$  is defined as

$$D(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

- It measures the expected number of extra bits (nats) required to describe samples from  $p(\mathbf{x})$  using a code based on  $q$  instead of  $p$
- $D(p||q) \geq 0$  for all  $p, q$ , with equality if and only if  $p = q$
- The KL-divergence is asymmetric

# KL-divergence

- Suppose  $p$  is the true distribution

$$D(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

- This is difficult to optimize because the expectations w.r.t.  $p$  are typically intractable
- We can reverse the KL

$$D(q||p) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})}$$

- Typically the expectation w.r.t.  $q$  will be tractable, but ...
- ... computing  $p(\mathbf{x})$  is still hard, due to the partition function
- What can we do?

- Let's look at the unnormalized distribution

$$\begin{aligned} J(q) &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{\tilde{p}(\mathbf{x})} \\ &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{Z \cdot p(\mathbf{x})} \\ &= \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} - \log Z \\ &= KL(q||p) - \log Z \end{aligned}$$

- Since  $Z$  is constant, by minimizing  $J(q)$ , we will force  $q$  to become close to  $p$
- The KL is always non-negative, so we see that  $J(q)$  is an upper bound on the negative log likelihood (NLL)

$$J(q) = KL(q||p) - \log Z \geq -\log Z = -\log p(\mathcal{D})$$

# Alternative Interpretations

- (1). We can alternatively write

$$J(q) = \mathbb{E}_q[\log q(\mathbf{x})] + \mathbb{E}_q[-\log \tilde{p}(\mathbf{x})] = -\mathbb{H}(q) + \mathbb{E}_q[E(\mathbf{x})]$$

which is the expected energy minus the entropy.

- In physics,  $J(q)$  is called the **variational free energy** or **Helmholtz free energy**
- (2). Another alternative:

$$\begin{aligned} J(q) &= \mathbb{E}_q[\log q(\mathbf{x}) - \log p(\mathbf{x})p(\mathcal{D})] \\ &= \mathbb{E}_q[\log q(\mathbf{x}) - \log p(\mathbf{x}) - \log p(\mathcal{D})] \\ &= \mathbb{E}_q[-\log p(\mathcal{D})] + KL(q||p) \end{aligned}$$

- This is the expected NLL plus a penalty term that measures how far apart the two distributions are

- Before we do something let's inspect again

$$KL(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

- What is the difference between the solution to

$$\arg \min_q KL(p||q)$$

and

$$\arg \min_q KL(q||p)$$

- They differ only when  $q$  is minimized over a restricted set of probability distribution  $Q = \{q_1, \dots\}$ , and  $p \neq q$ . Why?

# Forward or Reverse KL

- Minimizing  $KL(p||q)$  or  $KL(q||p)$  will give different results
- **I projection**, or **Information projection**

$$KL(q||p) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})}$$

This is infinite if  $p(\mathbf{x}) = 0$  and  $q(\mathbf{x}) > 0$ . Thus we must ensure that if  $p(\mathbf{x}) = 0$  then  $q(\mathbf{x}) = 0$

- Thus the reverse KL is **zero forcing** and  $q$  will under-estimate the support of  $p$
- **M projection** or **moment projection**

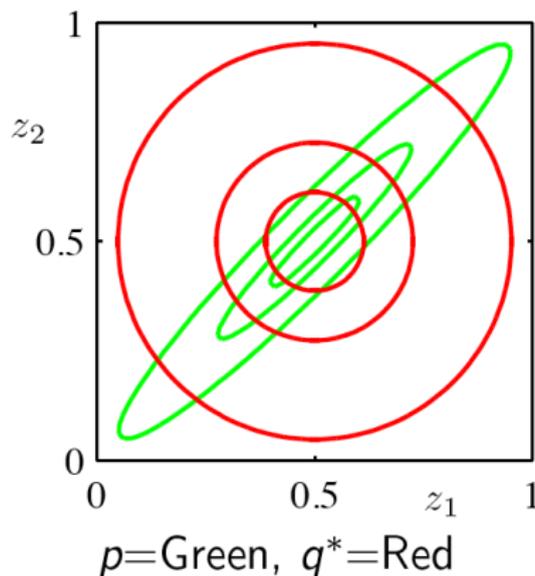
$$KL(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

- This is infinite if  $q(\mathbf{x}) = 0$  and  $p(\mathbf{x}) > 0$ . This is **zero avoiding**, and the forward KL over-estimates the support of  $p$

# KL divergence - M projection

$$q^* = \arg \min_{q \in Q} KL(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

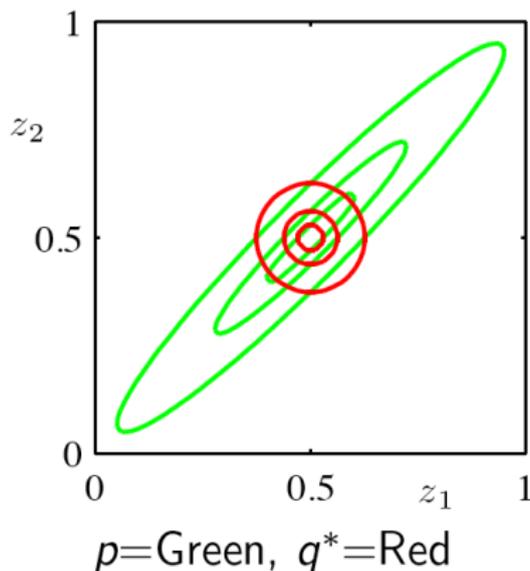
$p(\mathbf{x})$  is a 2D Gaussian and  $Q$  is the set of all Gaussian distributions with diagonal covariance matrices



# KL divergence - I projection

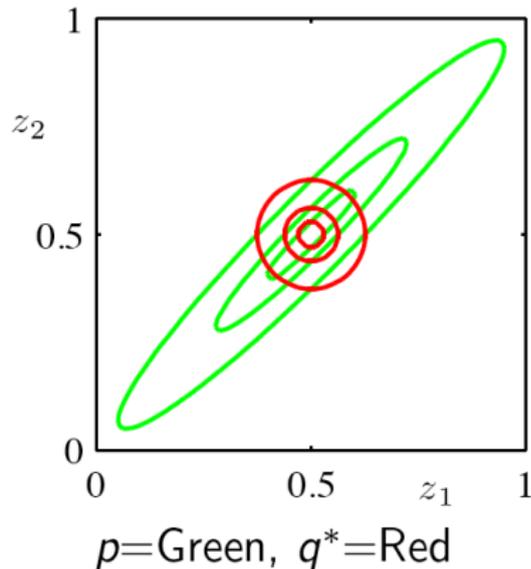
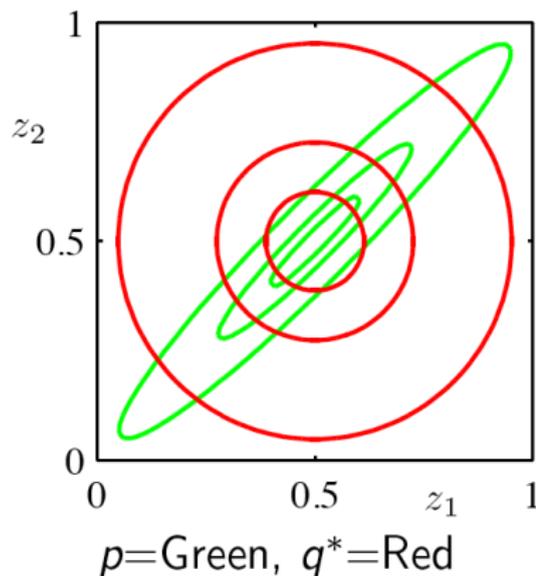
$$q^* = \arg \min_{q \in Q} KL(q||p) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})}$$

$p(\mathbf{x})$  is a 2D Gaussian and  $Q$  is the set of all Gaussian distributions with diagonal covariance matrices



# KL Divergence (single Gaussian)

- In this example, both the M-projection and I-projection find an approximate  $q(\mathbf{x})$  that has the correct mean (i.e.,  $\mathbb{E}_p(\mathbf{z}) = \mathbb{E}_q(\mathbf{x})$ )

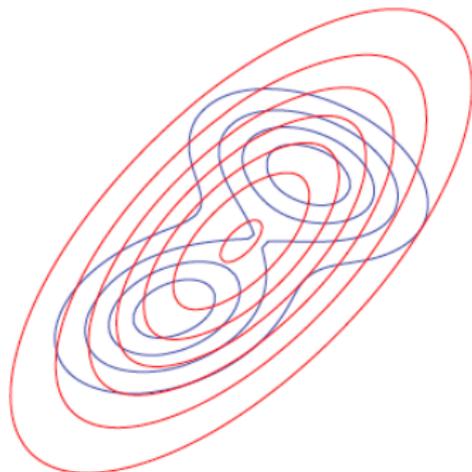


What if  $p(\mathbf{x})$  is multimodal?

# M projection (Mixture of Gaussians)

$$q^* = \arg \min_{q \in Q} KL(p||q) = \sum_{\mathbf{x}} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}$$

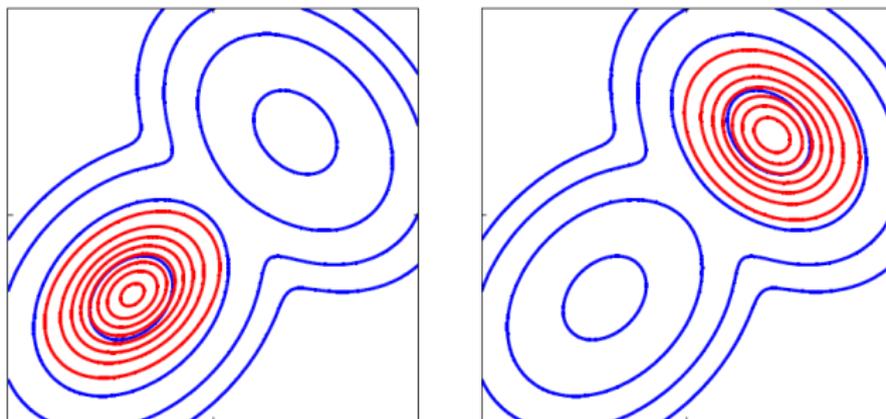
$p(\mathbf{x})$  is a mixture of two 2D Gaussians and  $Q$  is the set of all 2D Gaussian distributions (with arbitrary covariance matrices)



M-projection yields a distribution  $q(\mathbf{x})$  with the correct mean and covariance.

# I projection (Mixture of Gaussians)

$$q^* = \arg \min_{q \in Q} KL(q||p) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})}$$



$p$ =Blue,  $q^*$ =Red (two local minima!)

The I-projection does not necessarily yield the correct moments

# Mean Field

- One of the most popular variational inference algorithms [Opper & Saad 01]
- Assume that the posterior fully factorizes

$$q(\mathbf{x}) = \prod_i q_i(x_i)$$

- Our goal is to

$$\min_{q_1, \dots, q_D} KL(q||p)$$

where we optimize over the parameters of each marginal distribution  $q_i$

- Minimize the upper bound  $J(q) \geq -\log p(\mathcal{D})$  or alternatively we want to maximize the lower bound

$$L(q) = -J(q) = \sum_{\mathbf{x}} q(\mathbf{x}) \log \frac{\tilde{p}(\mathbf{x})}{q(\mathbf{x})} \leq \log p(\mathcal{D})$$

- We can do the maximization one node at a time, in an iterative fashion

# Mean Field Updates

- Focus on  $q_j$  (holding all other terms constant)

$$\begin{aligned}L(q_j) &= \sum_{\mathbf{x}} \prod_i q_i(\mathbf{x}) \left[ \log \tilde{p}(\mathbf{x}) - \sum_k \log q_k(\mathbf{x}_k) \right] \\&= \sum_{\mathbf{x}_j} \sum_{\mathbf{x}_{-j}} q_j(\mathbf{x}_j) \prod_{i \neq j} q_i(\mathbf{x}_i) \left[ \log \tilde{p}(\mathbf{x}) - \sum_k \log q_k(\mathbf{x}_k) \right] \\&= \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \log \tilde{p}(\mathbf{x}) - \\&\quad \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \left[ \sum_{k \neq j} \log q_k(\mathbf{x}_k) + \log q_j(\mathbf{x}_j) \right] \\&= \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \log f_j(\mathbf{x}_j) - \sum_{\mathbf{x}_j} q_j(\mathbf{x}_j) \log q_j(\mathbf{x}_j) + \text{const}\end{aligned}$$

where

$$\log f_j(\mathbf{x}_j) = \sum_{\mathbf{x}_{-j}} \prod_{i \neq j} q_i(\mathbf{x}_i) \log \tilde{p}(\mathbf{x}) = \mathbb{E}_{-q_j}[\log \tilde{p}(\mathbf{x})]$$

- So we average out all the variables except  $\mathbf{x}_j$ , and can rewrite  $L(q_j)$  as

$$L(q_j) = -KL(q_j || f_j)$$

# Variational Inference for Graphical Models

- Suppose that we have an arbitrary graphical model

$$p(\mathbf{x}; \theta) = \frac{1}{Z(\theta)} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c) = \exp \left( \sum_{c \in \mathcal{C}} \theta_c(\mathbf{x}_c) - \ln Z(\theta) \right)$$

- We can compute the KL

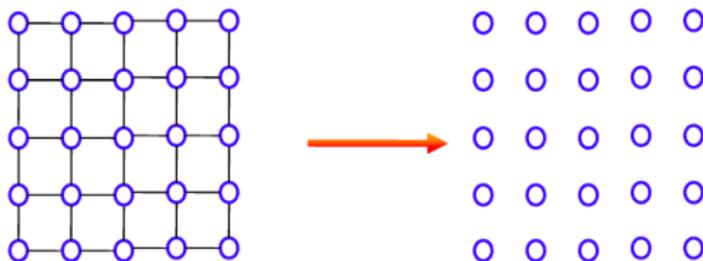
$$\begin{aligned} KL(q||p) &= \sum_{\mathbf{x}} q(\mathbf{x}) \ln \frac{q(\mathbf{x})}{p(\mathbf{x})} \\ &= - \sum_{\mathbf{x}} q(\mathbf{x}) \ln p(\mathbf{x}) - \sum_{\mathbf{x}} q(\mathbf{x}) \ln \frac{1}{q(\mathbf{x})} \\ &= - \sum_{\mathbf{x}} q(\mathbf{x}) \left( \sum_{c \in \mathcal{C}} \theta_c(\mathbf{x}_c) - \ln Z(\theta) \right) - H(q(\mathbf{x})) \\ &= - \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}} q(\mathbf{x}) \theta_c(\mathbf{x}_c) + \sum_{\mathbf{x}} q(\mathbf{x}) \ln Z(\theta) - H(q(\mathbf{x})) \\ &= - \sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + \ln Z(\theta) - H(q(\mathbf{x})) \end{aligned}$$

- The partition function can be considered as constant when minimizing over  $q$

# Mean Field for Variational Inference

$$\max_{q \in \mathcal{Q}} \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} q(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))$$

- Although this function is concave and thus in theory should be easy to optimize, we need some compact way of representing  $q(\mathbf{x})$
- **Mean field:** assume a factored representation of the joint distribution



$$q(\mathbf{x}) = \prod_{i \in V} q_i(x_i)$$

This is called "naive" mean field

# Naive Mean Field

- Suppose that  $Q$  consists of all fully factorized distributions, then we can simplify

$$\max_{q \in Q} \sum_{c \in C} \sum_{\mathbf{x}_c} q(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))$$

$$\text{since } q(\mathbf{x}_c) = \prod_{i \in C} q_i(x_i)$$

- The joint entropy decomposes as a sum of local entropies

$$\begin{aligned} H(q) &= - \sum_{\mathbf{x}} q(\mathbf{x}) \ln q(\mathbf{x}) \\ &= - \sum_{\mathbf{x}} q(\mathbf{x}) \ln \prod_{i \in V} q_i(x_i) = - \sum_{\mathbf{x}} q(\mathbf{x}) \sum_{i \in V} \ln q_i(x_i) \\ &= - \sum_{i \in V} \sum_{\mathbf{x}} q(\mathbf{x}) \ln q_i(x_i) \\ &= - \sum_{i \in V} \sum_{\mathbf{x}_i} q_i(x_i) \ln q_i(x_i) \sum_{\mathbf{x}_{-i}} q(\mathbf{x}_{-i} | x_i) = \sum_{i \in V} H(q_i) \end{aligned}$$

# Naive Mean Field

- Suppose that  $Q$  consists of all fully factorized distributions, then we can simplify

$$\max_{q \in Q} \sum_{c \in C} \sum_{\mathbf{x}_c} q(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))$$

since  $q(\mathbf{x}_c) = \prod_{i \in C} q_i(x_i)$

- The joint entropy decomposes as a sum of local ones  $H(q) = \sum_{i \in V} H(q_i)$
- Putting these together, we obtain

$$\max_q \sum_{c \in C} \sum_{\mathbf{x}_c} \theta_c(\mathbf{x}_c) \prod_{i \in C} q_i(x_i) + \sum_{i \in V} H(q_i)$$

subject to the constraints

$$q_i(x_i) \geq 0 \quad \forall i \in V, x_i$$

$$\sum_{x_i} q_i(x_i) = 1 \quad \forall i \in V$$

# Naive Mean Field for Pairwise MRFs

- For pairwise MRFs we have

$$\max_q \sum_{ij \in E} \sum_{x_i, x_j} \theta_{ij}(x_i, x_j) q_i(x_i) q_j(x_j) - \sum_{i \in V} \sum_{x_i} q_i(x_i) \ln q_i(x_i) \quad (1)$$

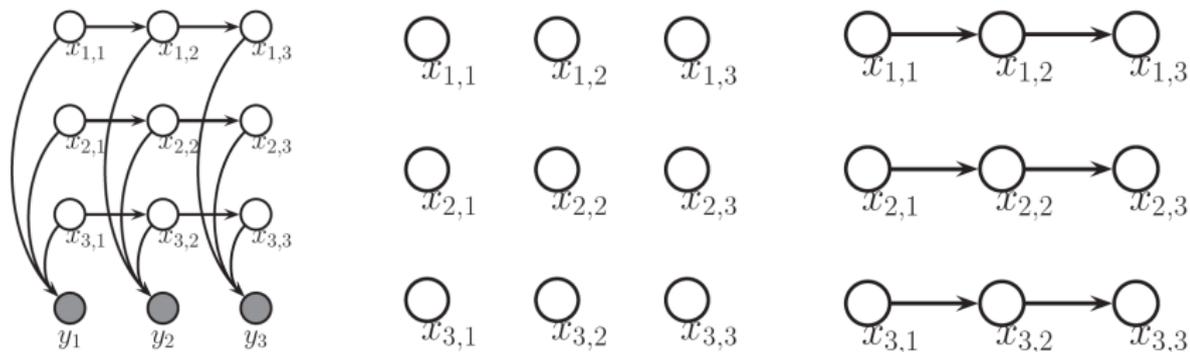
- This is a non-concave optimization problem, with many local maxima!
- We can do block coordinate ascent
  - 1 For each  $i \in V$ 
    - Fully maximize Eq. (1) wrt  $\{q_i(x_i), \forall x_i\}$
  - 2 repeat until convergence
- Constructing the Lagrangian, taking the derivatives and setting to zero yields the update

$$q_i(x_i) \leftarrow \frac{1}{Z_i} \exp \left\{ \theta_i(x_i) + \sum_{j \in N(i)} \sum_{x_j} q_j(x_j) \theta_{ij}(x_i, x_j) \right\}$$

- See *Mean field example for the Ising Model*, Murphy 21.3.2

# Structured mean-field approximations

- Rather than assuming a fully-factored distribution for  $q$ , we can use a structured approximation, such as a spanning tree
- For example, for a factorial HMM, a good approximation may be a product of chain-structured models (see Murphy 21.4.1)



# Approximate Inference via Loopy BP

- Mean field inference approximates posterior as product of marginal distributions
- Allows use of different forms for each variable: useful when inferring statistical parameters of models, or regression weights
- An alternative approximate inference algorithm is **loopy belief propagation**
- Same algorithm shown to do exact inference in trees last class
- In loopy graphs, BP not guaranteed to give correct results, may not converge, but often works well in practice

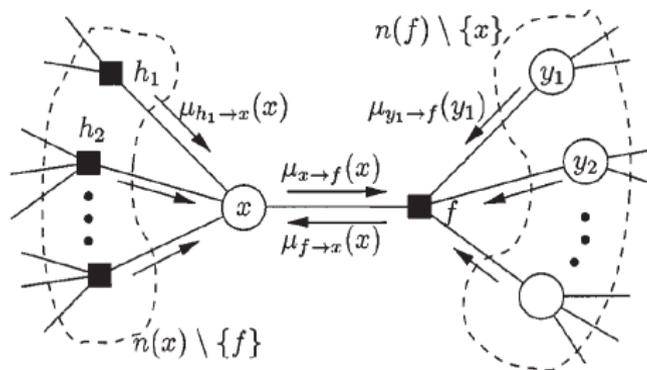
---

**Algorithm 22.1:** Loopy belief propagation for a pairwise MRF

---

- 1 Input: node potentials  $\psi_s(x_s)$ , edge potentials  $\psi_{st}(x_s, x_t)$ ;
  - 2 Initialize messages  $m_{s \rightarrow t}(x_t) = 1$  for all edges  $s - t$ ;
  - 3 Initialize beliefs  $\text{bel}_s(x_s) = 1$  for all nodes  $s$ ;
  - 4 **repeat**
  - 5     Send message on each edge  
    
$$m_{s \rightarrow t}(x_t) = \sum_{x_s} \left( \psi_s(x_s) \psi_{st}(x_s, x_t) \prod_{u \in \text{nbr}_s \setminus t} m_{u \rightarrow s}(x_s) \right);$$
  - 6     Update belief of each node  $\text{bel}_s(x_s) \propto \psi_s(x_s) \prod_{t \in \text{nbr}_s} m_{t \rightarrow s}(x_s)$ ;
  - 7 **until** *beliefs don't change significantly*;
  - 8 Return marginal beliefs  $\text{bel}_s(x_s)$ ;
-

# Loopy BP for Factor Graph



$$m_{i \rightarrow f}(x_i) = \prod_{h \in M(i) \setminus f} m_{h \rightarrow i}(x_i)$$

$$m_{f \rightarrow i}(x_i) = \sum_{\mathbf{x}_c \setminus x_i} f(\mathbf{x}_c) \prod_{j \in N(f) \setminus i} m_{j \rightarrow f}(x_j)$$

$$\mu_i(x_i) \propto \prod_{f \in M(i)} m_{f \rightarrow i}(x_i)$$

# Convergence of LBP

- Can we predict when will converge?
  - Unroll messages across time in a *computation tree*:  $T$  iterations of LBP is exact computation in tree of height  $T + 1$
  - if leaves' effect on root diminishes over time will converge
- Can we make it more likely to converge?
  - Damp the messages to avoid oscillations
  - Can we speed up convergence?
- Change from synchronous to asynchronous updates
  - Update sets of nodes at a time, e.g., spanning trees (*tree reparameterization*)

- More theoretical analysis of LBP from variational point of view: (Wainwright & Jordan, 2008)
- Dense tome
- Simplify by considering pairwise UGMs, discrete variables

# Variational Inference for Graphical Models

- Suppose that we have an arbitrary graphical model

$$p(\mathbf{x}; \theta) = \frac{1}{Z(\theta)} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c) = \exp \left( \sum_{c \in \mathcal{C}} \theta_c(\mathbf{x}_c) - \ln Z(\theta) \right)$$

- We can compute the KL

$$\begin{aligned} KL(q||p) &= \sum_{\mathbf{x}} q(\mathbf{x}) \ln \frac{q(\mathbf{x})}{p(\mathbf{x})} \\ &= - \sum_{\mathbf{x}} q(\mathbf{x}) \ln p(\mathbf{x}) - \sum_{\mathbf{x}} q(\mathbf{x}) \ln \frac{1}{q(\mathbf{x})} \\ &= - \sum_{\mathbf{x}} q(\mathbf{x}) \left( \sum_{c \in \mathcal{C}} \theta_c(\mathbf{x}_c) - \ln Z(\theta) \right) - H(q(\mathbf{x})) \\ &= - \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}} q(\mathbf{x}) \theta_c(\mathbf{x}_c) + \sum_{\mathbf{x}} q(\mathbf{x}) \ln Z(\theta) - H(q(\mathbf{x})) \\ &= - \sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + \ln Z(\theta) - H(q(\mathbf{x})) \end{aligned}$$

- The partition function is a constant when minimizing over  $q$

# The log-partition Function

- Since  $KL(q||p) \geq 0$  we have

$$-\sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + \ln Z(\theta) - H(q(\mathbf{x})) \geq 0$$

which implies

$$\ln Z(\theta) \geq \sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + H(q(\mathbf{x}))$$

- Thus, any approximating distribution  $q(\mathbf{x})$  gives a lower bound on the log-partition function
- Recall that  $KL(p||q) = 0$  if and only if  $p = q$ . Thus, if we optimize over all distributions we have

$$\ln Z(\theta) = \max_q \sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + H(q(\mathbf{x}))$$

- This casts exact inference as a variational optimization problem

# Rewriting Objective in terms of Moments

$$\begin{aligned}\ln Z(\theta) &= \max_q \sum_{c \in \mathcal{C}} \mathbb{E}_q[\theta_c(\mathbf{x}_c)] + H(q(\mathbf{x})) \\ &= \max_q \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}} q(\mathbf{x}) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x})) \\ &= \max_q \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} q(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))\end{aligned}$$

- Assume that  $p(\mathbf{x})$  is in the exponential family, and let  $\mathbf{f}(\mathbf{x})$  be its sufficient statistic vector
- Define  $\mu_q = \mathbb{E}[\mathbf{f}(\mathbf{x})]$  to be the marginals of  $q(\mathbf{x})$
- We can re-write the objective as

$$\ln Z(\theta) = \max_{\mu \in M} \max_{q: \mathbb{E}_q[\mathbf{f}(\mathbf{x})] = \mu} \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} \mu_c(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))$$

where  $M$  is the **marginal polytope**, having all valid marginal vectors

# Rewriting Objective in terms of Moments

- We next push the max inside

$$\ln Z(\theta) = \max_{\mu \in M} \sum_{c \in \mathcal{C}} \sum_{\mathbf{x}_c} \theta_c(\mathbf{x}_c) \mu_c(\mathbf{x}_c) + H(\mu)$$

$$H(\mu) = \max_{q: \mathbb{E}_q[\mathbf{f}(\mathbf{x})] = \mu} H(q)$$

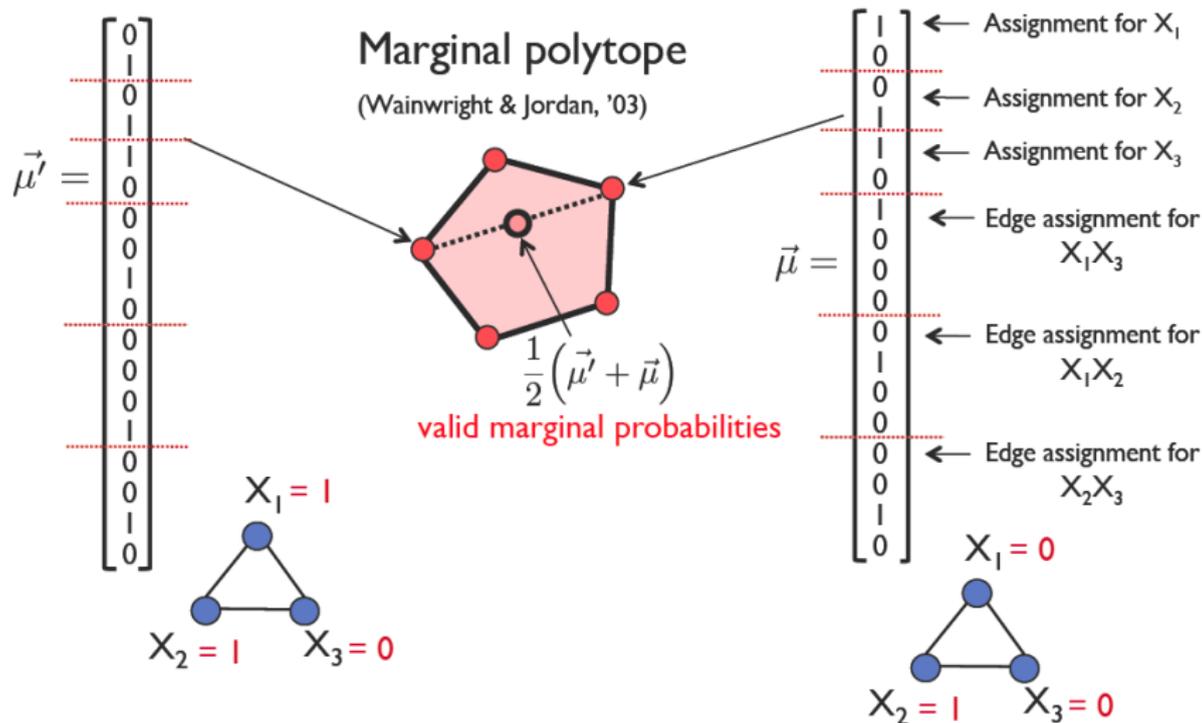
- For discrete random variables, the marginal polytope  $M$  is the set of all mean parameters for the given model that can be generated from a valid prob. distribution

$$M = \left\{ \mu \in \mathbb{R}^d \mid \exists p \text{ s.t. } \mu = \sum_{\mathbf{x} \in \mathcal{X}^m} p(\mathbf{x}) \mathbf{f}(\mathbf{x}) \text{ for some } p(\mathbf{x}) \geq 0, \sum_{\mathbf{x} \in \mathcal{X}^m} p(\mathbf{x}) = 1 \right\}$$
$$= \text{conv} \{ \mathbf{f}(\mathbf{x}), \mathbf{x} \in \mathcal{X}^m \}$$

with conv the convex hull (it has exponential number of facets)

- For a discrete-variable MRF, the sufficient statistic vector  $\mathbf{f}(\mathbf{x})$  is simply the concatenation of indicator functions for each clique of variables that appear together in a potential function
- For example, if we have a pairwise MRF on binary variables with  $m = |V|$  variables and  $|E|$  edges,  $d = 2m + 4|E|$

# Marginal Polytope for Discrete MRFs



$$\ln Z(\theta) = \max_{\mu \in M} \sum_{c \in C} \sum_{\mathbf{x}_c} \theta_c(\mathbf{x}_c) \mu_c(\mathbf{x}_c) + H(\mu)$$

We still haven't achieved anything, because:

- The marginal polytope  $M$  is complex to describe (in general, exponentially many vertices and facets)
- $H(\mu)$  is very difficult to compute or optimize over

We now make two approximations:

- We replace  $M$  with a relaxation of the marginal polytope, e.g. the local consistency constraints  $M_L$
- We replace  $H(\mu)$  with a function  $\tilde{H}(\mu)$  which approximates  $H(\mu)$

# Local Consistency Constraints

- For every "cluster" of variables to choose a local assignment

$$\mu_i(x_i) \in \{0, 1\} \quad \forall i \in V, x_i$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in V$$

$$\mu_{ij}(x_i, x_j) \in \{0, 1\} \quad \forall i, j \in E, x_i, x_j$$

$$\sum_{x_i, x_j} \mu_{ij}(x_i, x_j) = 1 \quad \forall i, j \in E$$

- Enforce that these local assignments are globally consistent

$$\mu_i(x_i) = \sum_{x_j} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_i$$

$$\mu_j(x_j) = \sum_{x_i} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_j$$

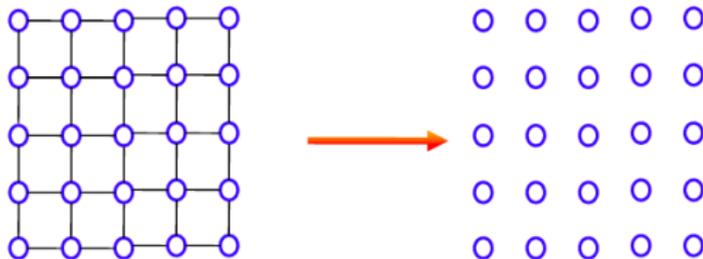
- The local consistency polytope,  $M_L$  is defined by these constraints
- The  $\mu_i$  and  $\mu_{ij}$  are called pseudo-marginals

polytope for a tree-structured MRF, and the pseudomarginals are the marginals. marginal polytope, i.e.,  $M \subseteq M_L$

# Mean-field vs relaxation

$$\max_{q \in Q} \sum_{c \in C} \sum_{\mathbf{x}_c} q(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + H(q(\mathbf{x}))$$

- **Relaxation** algorithms work directly with pseudo-marginals which may not be consistent with any joint distribution
- **Mean-field** algorithms assume a factored representation of the joint distribution



$$q(\mathbf{x}) = \prod_{i \in V} q_i(x_i)$$

# Naive Mean-Field

- Using the same notation naive mean-field is:

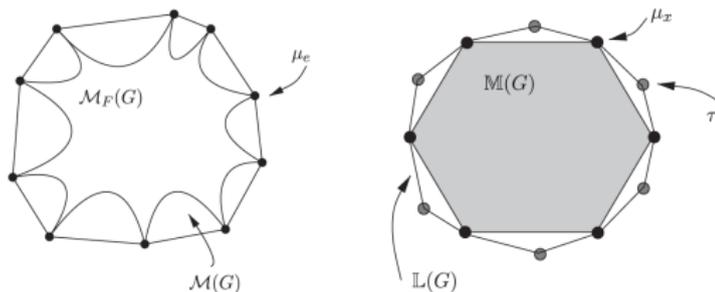
$$(*) \max_{\mu} \sum_{c \in C} \sum_{\mathbf{x}_c} \mu_c(\mathbf{x}_c) \theta_c(\mathbf{x}_c) + \sum_{i \in V} H(\mu_i) \quad \text{subject to}$$

$$\mu_i(x_i) \geq 0, \quad \forall i \in V, x_i$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in V$$

$$\mu_c(\mathbf{x}_c) = \prod_{i \in c} \mu_i(x_i)$$

- Corresponds to optimizing over an inner bound on the marginal polytope:



- We obtain a lower bound on the partition function, i.e.,  $(*) \leq \ln Z(\theta)$

# MAP Inference

- Recall the MAP inference task

$$\arg \max_{\mathbf{x}} p(\mathbf{x}), \quad p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c)$$

we assume any evidence has been subsumed into the potentials

- As the partition function is a constant we can alternatively

$$\arg \max_{\mathbf{x}} \prod_{c \in \mathcal{C}} \phi_c(\mathbf{x}_c)$$

This is the **max product** inference task

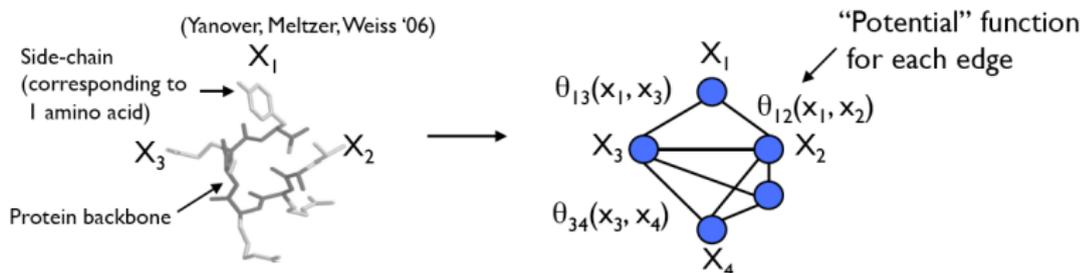
- Since the log is monotonic, let  $\theta_c(\mathbf{x}_c) = \log \phi_c(\mathbf{x}_c)$

$$\arg \max_{\mathbf{x}} \sum_{c \in \mathcal{C}} \theta_c(\mathbf{x}_c)$$

This is called the **max-sum**

# Application: protein side-chain placement

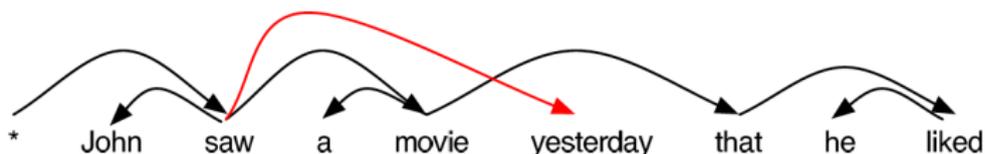
- Find "minimum energy" configuration of amino acid side-chains along fixed carbon backbone:



- Orientations of the side-chains are represented by discretized angles called rotamers
- Rotamer choices for nearby amino acids are energetically coupled (attractive and repulsive forces)

# Application: Dependency parser

- Given a sentence, predict the dependency tree that relates the words



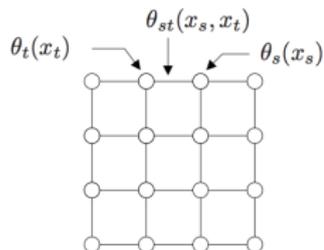
- Arc from head word of each phrase to words that modify it
- May be non-projective: each word and its descendants may not be a contiguous subsequence
- $m$  words  $\Rightarrow m(m - 1)$  binary arc selection variables  $x_{ij} \in \{0, 1\}$
- We represent the problem as

$$\max_{\mathbf{x}} \theta_T(\mathbf{x}) + \sum_{ij} \theta_{ij}(x_{ij}) + \sum_i \theta_i(\mathbf{x}_{|i})$$

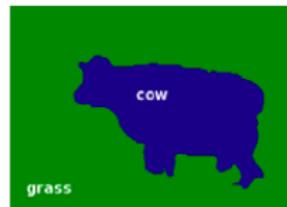
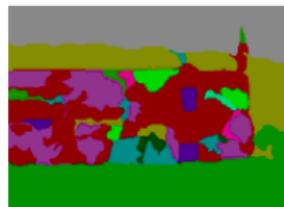
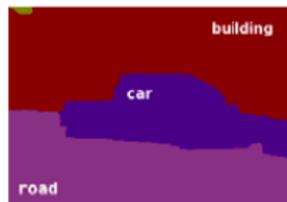
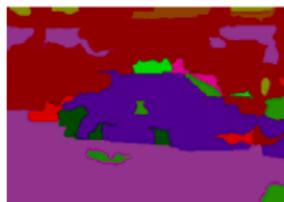
with  $\mathbf{x}_{|i} = \{x_{ij}\}_{j \neq i}$  (all outgoing edges)

# Application: Semantic Segmentation

- Use Potts to encode that neighboring pixels are likely to have the same discrete label and hence belong to the same segment



$$p(\mathbf{x}, \theta) = \max_{\mathbf{x}} \sum_i \theta_i(x_i) + \sum_{i,j} \theta_{i,j}(x_i, x_j)$$



# MAP as an integer linear program (ILP)

- MAP as a discrete optimization problem is

$$\mathbf{x}^* = \arg \max_{\mathbf{x}} \sum_{i \in V} \theta_i(x_i) + \sum_{ij \in E} \theta_{ij}(x_i, x_j)$$

- To turn this into an integer linear program (ILP) we introduce indicator variables

- ①  $\mu_i(x_i)$ , one for each  $i \in V$  and state  $x_i$
- ②  $\mu_{ij}(x_i, x_j)$ , one for each edge  $ij \in E$  and pair of states  $x_i, x_j$

- The objective function is then

$$\max_{\mu} \sum_{i \in V} \sum_{x_i} \theta_i(x_i) \mu_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \mu_{ij}(x_i, x_j)$$

- What is the dimension of  $\mu$ , if binary variables?
- Are these two problems equivalent?

$$\max_{\mu} \sum_{i \in V} \sum_{x_i} \theta_i(x_i) \mu_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \mu_{ij}(x_i, x_j)$$

- For every "cluster" of variables to choose a local assignment

$$\mu_i(x_i) \in \{0, 1\} \quad \forall i \in V, x_i$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in V$$

$$\mu_{ij}(x_i, x_j) \in \{0, 1\} \quad \forall i, j \in E, x_i, x_j$$

$$\sum_{x_i, x_j} \mu_{ij}(x_i, x_j) = 1 \quad \forall i, j \in E$$

- Enforce that these local assignments are globally consistent

$$\mu_i(x_i) = \sum_{x_j} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_i$$

$$\mu_j(x_j) = \sum_{x_i} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_j$$

# MAP as an integer linear program (ILP)

$$\max_{\mu} \sum_{i \in V} \sum_{x_i} \theta_i(x_i) \mu_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \mu_{ij}(x_i, x_j)$$

subject to:

$$\mu_i(x_i) \in \{0, 1\} \quad \forall i \in V, x_i$$

$$\mu_{ij}(x_i, x_j) \in \{0, 1\} \quad \forall i, j \in E, x_i, x_j$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in V$$

$$\mu_i(x_i) = \sum_{x_j} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_i$$

$$\mu_j(x_j) = \sum_{x_i} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_j$$

- Many extremely good off-the-shelf solvers, such as CPLEX and Gurobi
- But it might be too slow...

# Linear Programming Relaxation for MAP

$$MAP(\theta) = \max_{\mu} \sum_{i \in V} \sum_{x_i} \theta_i(x_i) \mu_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \mu_{ij}(x_i, x_j)$$

subject to:

$$\begin{aligned} \mu_i(x_i) &\in \{0, 1\} && \forall i \in V, x_i \\ \mu_{ij}(x_i, x_j) &\in \{0, 1\} && \forall i, j \in E, x_i, x_j \\ \sum_{x_i} \mu_i(x_i) &= 1 && \forall i \in V \\ \mu_i(x_i) &= \sum_{x_j} \mu_{ij}(x_i, x_j) && \forall ij \in E, x_i \\ \mu_j(x_j) &= \sum_{x_i} \mu_{ij}(x_i, x_j) && \forall ij \in E, x_j \end{aligned}$$

- Relax integrality constraints, allowing the variables to be between 0 and 1

$$\mu_i(x_i) \in [0, 1] \quad \forall i \in V, x_i \quad \mu_{ij}(x_i, x_j) \in [0, 1] \quad \forall ij \in E, x_i, x_j$$

# Linear Programming Relaxation for MAP

$$LP(\theta) = \max_{\mu} \sum_{i \in V} \sum_{x_i} \theta_i(x_i) \mu_i(x_i) + \sum_{ij} (x_i, x_j) \mu_{ij}(x_i, x_j)$$

$$\mu_i(x_i) \in [0, 1] \quad \forall i \in V, x_i$$

$$\mu_{ij}(x_i, x_j) \in [0, 1] \quad \forall i, j \in E, x_i, x_j$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in V$$

$$\mu_i(x_i) = \sum_{x_j} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_i$$

$$\mu_j(x_j) = \sum_{x_i} \mu_{ij}(x_i, x_j) \quad \forall ij \in E, x_j$$

- Linear programs can be solved relatively efficient via Simplex method, interior point, ellipsoid algorithm
- Since the LP relaxation maximizes over a larger set of solutions, its value can only be higher

$$MAP(\theta) \leq LP(\theta)$$

- LP relaxation is tight for tree-structured MRFs
- Faster algorithms by deriving the dual (dual variables represent messages)
- Zero limit temperature of the variational inference for Marginals

# Loopy Belief Propagation (Max Product)

- Introducing Lagrange multipliers and solving we get (see Murphy 22.3.5.4)

$$M_{i \rightarrow j}(x_i) \propto \max_{x_j} \left[ \exp\{\theta_{ij}(x_i, x_j) + \theta_j(x_j)\} \prod_{u \in N(j) \setminus i} M_{u \rightarrow j}(x_j) \right]$$

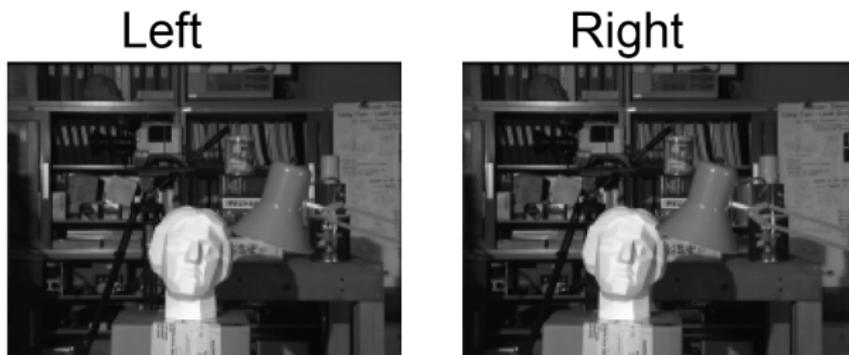
- Thus we pass messages for a fixed number of iterations, or until the messages do not change too much
- We decode the local scoring functions by

$$\mu_s(x_s) \propto \exp(\theta_s(x_s)) \prod_{t \in N(s)} M_{t \rightarrow s}(x_s)$$

- We then compute the maximal value of  $\mu_s(x_s)$
- What if two solutions that have the same score?

# Stereo Estimation

- Tsukuba images from Middlebury stereo database



- MRF for each pixel, with states the disparity
- Our unary is the matching term

$$\theta_i(d_i) = |L(x + d_i, y) - R(x, y)|$$

where pixel  $p_i = (x, y)$

- The pairwise factor  $\theta_{ij}$  between neighboring pixels favor smoothness

# Stereo Estimation

- If we only use the unary terms. How would you do inference in this case?



- If full graphical model

left,



right,



up,



down sweeps



[Credit: Coughlan BP Tutorial]

Subsequent iterations:

2



3



4



5



... 20



Note:

Little change after first few iterations.

Model can be improved to give better results  
-- this is just a simple example to illustrate BP.

[Credit: Coughlan BP Tutorial]