Approximate MAP Inference
Belief Propagation

• Efficient method for inference on a tree

• Represent the variable elimination process as a collection of messages passed between nodes in the tree
  – The messages keep track of the potential functions produced throughout the elimination process
Belief Propagation

- \( p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j) \)

\[
m_{i \rightarrow j}(x_j) = \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{k \rightarrow i}(x_i)
\]

where \( N(i) \) is the set of neighbors of node \( i \) in the graph

- Messages are passed in two phases: from the leaves up to the root and then from the root down to the leaves
MAP Inference

- Compute the most likely assignment under the (conditional) joint distribution

\[ x^* = \arg \max_x p(x) \]

- Can encode 3-SAT, maximum independent set problem, etc. as a MAP inference problem
Max-Product

- \( p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j) \)

\[
m_{i \to j}(x_j) = \max_{x_i} \left[ \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{k \to i}(x_i) \right]
\]

- Guaranteed to produced the correct answer on a tree
Max-Product

- To construct the maximizing assignment, we look at the max-marginal produced by the algorithm

\[ \mu_i(x_i) = \frac{1}{Z} \phi_i(x_i) \prod_{k \in N(i)} m_{k \rightarrow i}(x_i) \]

- Last time, we argued that, on a tree,

\[ \mu_i(x_i) = \max_{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n} p(x_1, \ldots, x_n) \]
Reparameterization

- The messages passed in max-product can be used to construct a reparameterization of the joint distribution

\[
p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j)
\]

and

\[
p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i \in V} \left[ \phi_i(x_i) \prod_{k \in N(i)} m_{k \rightarrow i}(x_i) \right] \prod_{(i,j) \in E} \frac{\psi_{ij}(x_i, x_j)}{m_{i \rightarrow j}(x_j)m_{j \rightarrow i}(x_i)}
\]
Reparameterization

\[ p(x_1, ..., x_n) = \frac{1}{Z} \prod_{i \in V} \left[ \phi_i(x_i) \prod_{k \in N(i)} m_{k \rightarrow i}(x_i) \right] \prod_{(i,j) \in E} \frac{\psi_{ij}(x_i, x_j)}{m_{i \rightarrow j}(x_j)m_{j \rightarrow i}(x_i)} \]

- Reparameterizations do not change the partition function, the MAP solution, or the factorization of the joint distribution
  - They just push "weight" around between the different factors
- Other reparameterizations are possible/useful
Tree Reparameterization

- On a tree, this reparameterization takes a special form

\[ p(x_1, \ldots, x_n) = \frac{1}{Z'} \prod_{i \in V} \mu_i(x_i) \prod_{(i,j) \in E} \frac{\mu_{ij}(x_i, x_j)}{\mu_i(x_i)\mu_j(x_j)} \]

- \( \mu_i \) is the max-marginal distribution of the \( i^{th} \) variable and \( \mu_{ij} \) is the max-marginal distribution for the edge \( (i, j) \in E \)

- How to express \( \mu_{ij} \) as a function of the messages and the potential functions?
MAP in General MRFs

• While max-product solves the MAP problem on trees, the MAP problem in MRFs is, in general, intractable
  – Don’t expect to be able to solve the problem exactly
  – Will settle for “good” approximations
  – Can use max-product messages as a starting point

• This is an active area of research
  – Advances are constantly being made
Upper Bounds

\[
\max_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \leq \frac{1}{Z} \prod_{i \in V} \max_{x_i} \phi_i(x_i) \prod_{(i,j) \in E} \max_{x_i,x_j} \psi_{ij}(x_i, x_j)
\]

- This provides an upper bound on the optimization problem
  - Do other reparameterizations provide better bounds?
Duality

\[ L(m) = \frac{1}{Z} \prod_{i \in V} \max_{x_i} \left[ \phi_i(x_i) \prod_{k \in N(i)} m_{k \rightarrow i}(x_i) \right] \prod_{(i,j) \in E} \max_{x_i, x_j} \left[ \frac{\psi_{ij}(x_i, x_j)}{m_{i \rightarrow j}(x_j)m_{j \rightarrow i}(x_i)} \right] \]

- We construct a dual optimization problem

\[ \min_{m} L(m) \geq \max_{x} p(x) \]

- The dual problem is log-convex in the messages

\[ L(m)^{\delta} L(m')^{1-\delta} \geq L(\delta m + (1 - \delta)m') \]

Equivalently, \( \log L(m) \) is a convex function
Optimizing the Dual

• Minimizing $L(m)$

  – Block coordinate descent: improve the bound by changing only a small subset of the messages at a time (usually look like message-passing algorithms)

  – Subgradient descent: variant of gradient descent for non-differentiable functions

  – Many more methods...
Max-Sum Diffusion

- Can improve the bound iteratively by looking at only the pieces of the objective function involving the variable $x_i$ and forcing agreement

- That is, for all $j \in N(i)$, update $m_{ji}(x_i)$ so that

$$\max_{x_j} \left[ \frac{\psi_{ij}(x_i, x_j)}{m_{i\to j}(x_j) m_{j\to i}(x_i)} \right] = \phi_i(x_i) \prod_{k \in N(i)} m_{k\to i}(x_i)$$

- Pick a new $i \in V$ and iterate this process

- This can only improve the bound but is not guaranteed to minimize it (coordinate descent methods can “get stuck”)