Probabilistic and Bayesian Machine Learning

Day 4: Expectation and Belief Propagation

Yee Whye Teh

ywteh@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit
University College London

http://www.gatsby.ucl.ac.uk/∼ywteh/teaching/probmodels

The Other KL

Variational methods find $Q = \text{argmin}\,\textbf{KL}[Q\|Q(\mathbf{Y}|\mathbf{X})]$:

- guaranteed convergence;
- maximising lower bound may increase log likelihood;

What about the reversed KL ($Q = \text{argmin}\,\textbf{KL}[P(\mathbf{Y}|\mathbf{X})\|Q]$)?

For a factored approximation the marginals are **correct**:

$$\begin{aligned}
\underset{q_i}{\text{argmin}}\,\textbf{KL}\Big[P(\mathbf{Y}|\mathbf{X})\Big\|\prod Q_j(\mathbf{Y}_j)\Big] &= \underset{Q_i}{\text{argmin}} - \int d\mathbf{Y}\; P(\mathbf{Y}|\mathbf{X}) \log \prod_j Q_j(\mathbf{Y}_j) \\
&= \underset{Q_i}{\text{argmin}} - \sum_j \int d\mathbf{Y}\; P(\mathbf{Y}|\mathbf{X}) \log Q_j(\mathbf{Y}_j) \\
&= \underset{Q_i}{\text{argmin}} - \int d\mathbf{Y}_i\; P(\mathbf{Y}_i|\mathbf{X}) \log Q_i(\mathbf{Y}_i) \\
&= P(\mathbf{Y}_i|\mathbf{X})
\end{aligned}$$

and the marginals are what we need for learning.

But (perversely) this means optimizing this KL is intractrable...

Expectation Propagation

The posterior distribution we need to approximate is often a (normalised) product of factors:

$$P(\mathbf{Y}|\mathbf{X}) \propto \prod_j f_i(\mathbf{Y}_{C_j})$$

We wish to approximate this by a product of *simpler* terms: $$Q(\mathbf{Y}) := \prod_j \tilde{f}_j(\mathbf{Y}_{C_j})$$

$$\min_{\{\tilde{f}_j(\mathbf{Y}_{C_j}\}} \textbf{KL}\left[\prod_i f_j(\mathbf{Y}_{C_j}) \middle\| \prod_j \tilde{f}_j(\mathbf{Y}_{C_j})\right]$$ (intractable)

$$\min_{\tilde{f}_i(\mathbf{Y}_{C_i})} \textbf{KL}\left[f_i(\mathbf{Y}_{C_i}) \middle\| \tilde{f}_i(\mathbf{Y}_{C_i})\right]$$ (simple, non-iterative, inaccurate)

$$\min_{\tilde{f}_i(\mathbf{Y}_{C_i})} \textbf{KL}\left[f_i(\mathbf{Y}_{C_i})\prod_{j\neq i} \tilde{f}_j(\mathbf{Y}_{C_j}) \middle\| \tilde{f}_i(\mathbf{Y}_{C_i})\prod_{j\neq i} \tilde{f}_j(\mathbf{Y}_{C_j})\right]$$ (simple, iterative, accurate) ← EP

Expectation Propagation

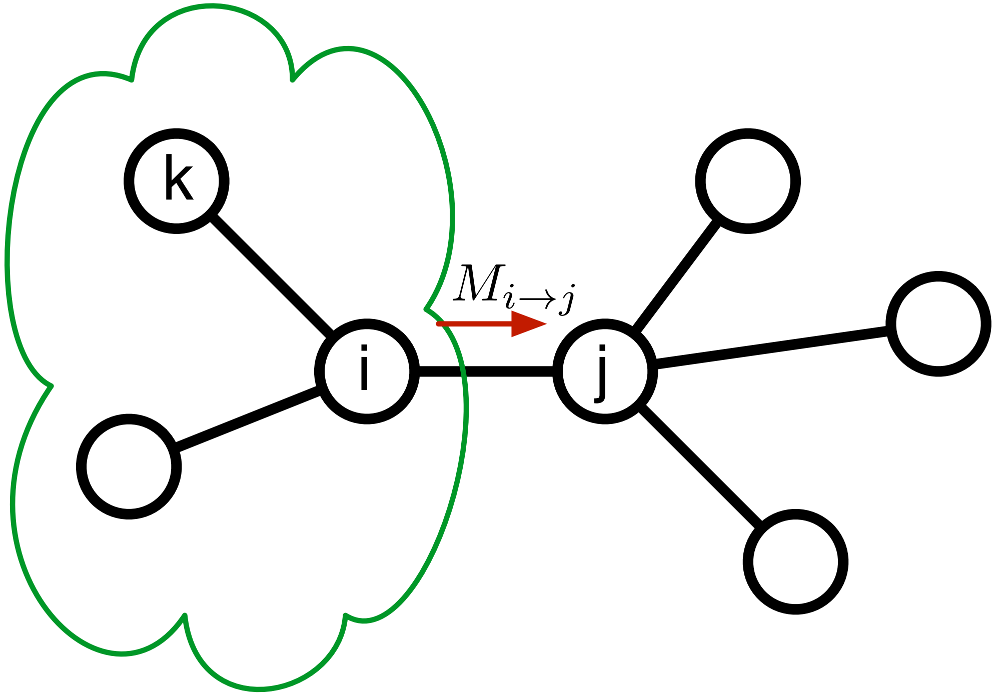
```
Input $\{f_i(\mathbf{Y}_{C_i})\}$
Initialize $\tilde{f}_i(\mathbf{Y}_{C_i}) = 1$, $Q(\mathbf{Y}) = \prod_i \tilde{f}_i(\mathbf{Y}_{C_i})$
repeat
  for each factor $i$ do
    Deletion: $Q_{\neg i}(\mathbf{Y}) \leftarrow \frac{Q(\mathbf{Y})}{\tilde{f}_i(\mathbf{Y}_{C_i})} = \prod_{j \neq i} \tilde{f}_j(\mathbf{Y}_{C_j})$
    Projection: $\tilde{f}_i^{\text{new}}(\mathbf{Y}_{C_i}) \leftarrow \underset{f_i'(\mathbf{Y}_{C_i})}{\text{argmin}}\ \textbf{KL}[f_i(\mathbf{Y}_{C_i})Q_{\neg i}(\mathbf{Y}) \| f_i'(\mathbf{Y}_{C_i})Q_{\neg i}(\mathbf{Y})]$
    Inclusion: $Q(\mathbf{Y}) \leftarrow \tilde{f}_i^{\text{new}}(\mathbf{Y}_{C_i})\, Q_{\neg i}(\mathbf{Y})$
  end for
until convergence
```

- KL minimisation (projection) only depends on $Q_{\neg i}(\mathbf{Y})$ marginalised to $\mathbf{Y}_{C_i}$.
- If $\tilde{f}_i(\mathbf{Y})$ in exponential family, then the projection step is **moment matching**.
- Update order need not be sequential.
- Minimizes the opposite KL to variational methods.
- Loopy belief propagation and assumed density filtering are special cases.
- No convergence guarantee (although convergent forms can be developed).
- The names (deletion, projection, inclusion) are not the same as in (Minka, 2001).

Recap: Belief Propagation on Undirected Trees

Joint distribution of undirected tree:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} f_i(X_i) \prod_{\text{edges } (ij)} f_{ij}(X_i, X_j)$$



Recursively compute messages:

$$M_{i\to j}(X_j) := \sum_{X_i} f_{ij}(X_i, X_j) f_i(X_i) \prod_{k \in \text{ne}(i)\setminus j} M_{k\to i}(X_i)$$

Marginal distributions:

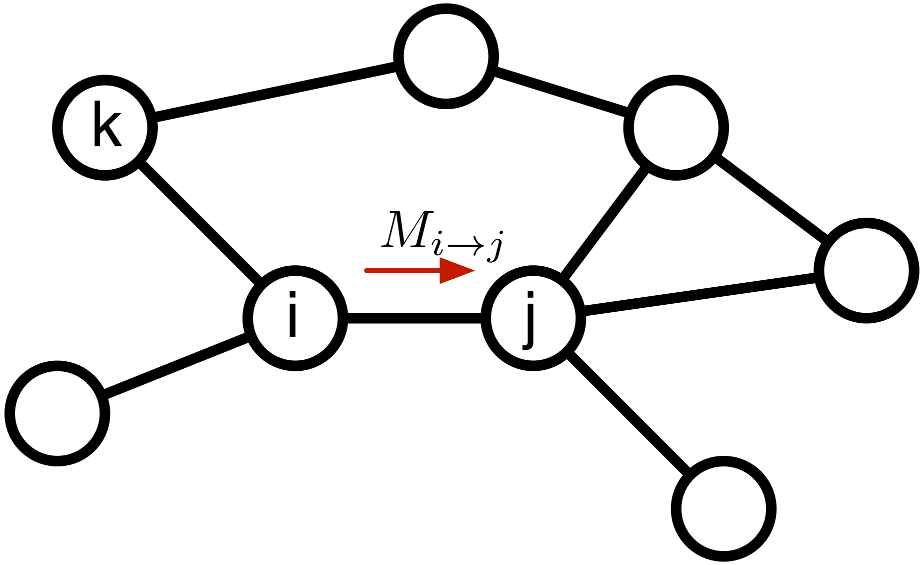
$$p(X_i) \propto f_i(X_i) \prod_{k \in \text{ne}(i)} M_{k\to i}(X_i)$$

$$p(X_i, X_j) \propto f_{ij}(X_i, X_j) f_i(X_i) f_j(X_j) \prod_{k \in \text{ne}(i)\setminus j} M_{k\to i}(X_i) \prod_{l \in \text{ne}(j)\setminus i} M_{l\to j}(X_j)$$

Loopy Belief Propagation

Joint distribution of undirected graph:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} f_i(X_i) \prod_{\text{edges } (ij)} f_{ij}(X_i, X_j)$$



Recursively compute messages **(and hope that updates converge)**:

$$M_{i \to j}(X_j) := \sum_{X_i} f_{ij}(X_i, X_j) f_i(X_i) \prod_{k \in \text{ne}(i) \setminus j} M_{k \to i}(X_i)$$

Approximate marginal distributions:

$$p(X_i) \approx b_i(X_i) \propto f_i(X_i) \prod_{k \in \text{ne}(i)} M_{k \to i}(X_i)$$

$$p(X_i, X_j) \approx b_{ij}(X_i, X_j) \propto f_{ij}(X_i, X_j) f_i(X_i) f_j(X_j) \prod_{k \in \text{ne}(i) \setminus j} M_{k \to i}(X_i) \prod_{l \in \text{ne}(j) \setminus i} M_{l \to j}(X_j)$$

Practical Considerations

- **Convergence**: Loopy BP is not guaranteed to converge for most graphs.
  - Trees: BP will converge.
  - Single loop: BP will converge for graphs containing at most one loop.
  - Weak interactions: BP will converge for graphs with weak enough interactions.
  - Long loops: BP more likely to converge for graphs with long (weakly interacting) loops.
  - Gaussian networks: Means correct, variances many converge under some conditions.
- **Damping**: Popular approach to encourage convergence.

$$M^{\text{new}}_{i\to j}(X_j) := (1-\alpha)M^{\text{old}}_{i\to j}(X_j) + \alpha \sum_{X_i} f_{ij}(X_i, X_j) f_i(X_i) \prod_{k\in \text{ne}(i)\backslash j} M_{k\to i}(X_i)$$

- **Other graphical models**: equivalent formulations for DAG and factor graphs.

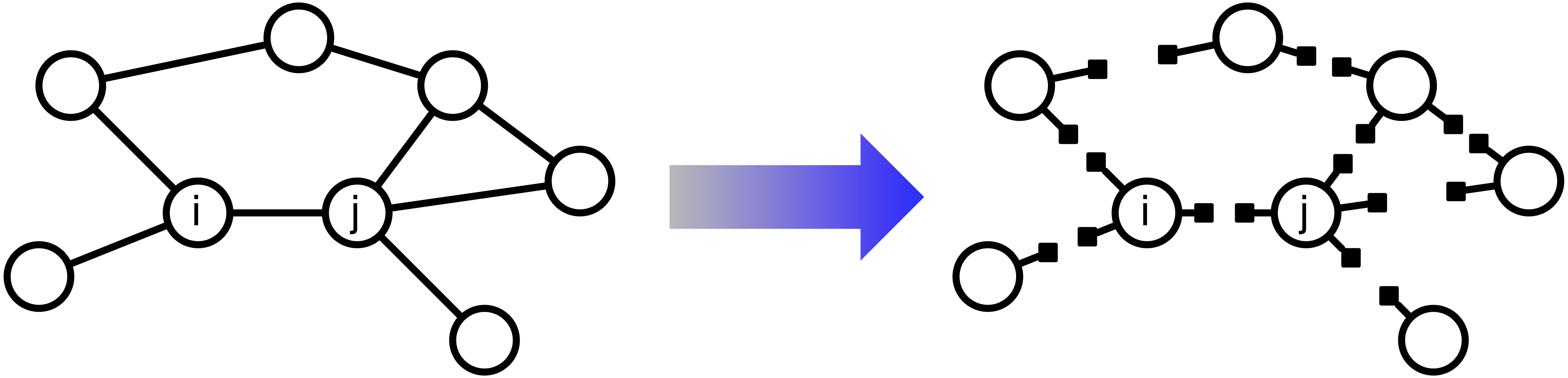
Different Perspectives on Loopy Belief Propagation

- Expectation propagation.

- Tree-based Reparametrization.

- Bethe free energy.

Loopy BP as Expectation Propagation



Approximate each factor f_{ij} describing interaction between i and j as:

$$f_{ij}(X_i, X_j) \approx \tilde{f}_{ij}(X_i, X_j) = M_{i \to j}(X_j) M_{j \to i}(X_i)$$

The full joint distribution is thus approximated by a factorized distribution:

$$p(\mathbf{X}) \approx \frac{1}{Z} \prod_{\text{nodes } i} f_i(X_i) \prod_{\text{edges } (ij)} \tilde{f}_{ij}(X_i, X_j) = \frac{1}{Z} \prod_{\text{nodes } i} f_i(X_i) \prod_{j \in \text{ne}(i)} M_{j \to i}(X_i) = \prod_{\text{nodes } i} b_i(X_i)$$

Loopy BP as Expectation Propagation

Each EP update to $\tilde{f}_{ij}$ is as follows:

- “Corrected” distribution is:

$$f_{ij}(X_i, X_j)q_{\neg ij}(\mathbf{X}) = f_{ij}(X_i, X_j)f_i(X_i)f_j(X_j) \prod_{k \in \text{ne}(i)\backslash j} M_{k\to i}(X_i) \prod_{l \in \text{ne}(j)\backslash i} M_{l\to j}(X_j)$$

$$\prod_{s \neq i,j} f_s(X_s) \prod_{t \in \text{ne}(s)} M_{t\to s}(X_s)$$

- Moments are just marginal distributions on i and j.
- Thus optimal $\tilde{f}_{ij}(X_i, X_j)$ minimizing

$$\textbf{KL}[f_{ij}(X_i, X_j)q_{\neg ij}(\mathbf{X}) \| \tilde{f}_{ij}(X_i, X_j)q_{\neg ij}(\mathbf{X})]$$

is given by:

$$f_j(X_j)M_{i\to j}(X_j) \prod_{l \in \text{ne}(j)\backslash i} M_{l\to j}(X_j) = \sum_{X_i} f_{ij}(X_i, X_j)f_i(X_i)f_j(X_j) \prod_{k \in \text{ne}(i)\backslash j} M_{k\to i}(X_i) \prod_{l \in \text{ne}(j)\backslash i} M_{l\to j}(X_j)$$

$$M_{i\to j}(X_j) = \sum_{X_i} f_{ij}(X_i, X_j)f_i(X_i) \prod_{k \in \text{ne}(i)\backslash j} M_{k\to i}(X_i)$$

Similarly for $M_{j\to i}(X_i)$.

Loopy BP as Tree-based Reparametrization

Many ways of parametrizing tree-structured distributions.

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} f_i(X_i) \prod_{\text{edges } (ij)} f_{ij}(X_i, X_j) \qquad \text{undirected tree} \qquad (1)$$

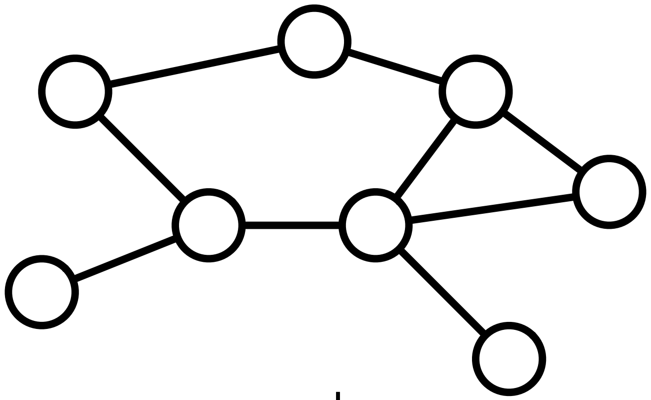
$$= p(X_r) \prod_{i \neq r} p(X_i | X_{\text{pa}(i)}) \qquad \text{directed (rooted) tree} \qquad (2)$$

$$= \prod_{\text{nodes } i} p(X_i) \prod_{\text{edges } (ij)} \frac{p(X_i, X_j)}{p(X_i)p(X_j)} \qquad \text{locally consistent marginals} \qquad (3)$$

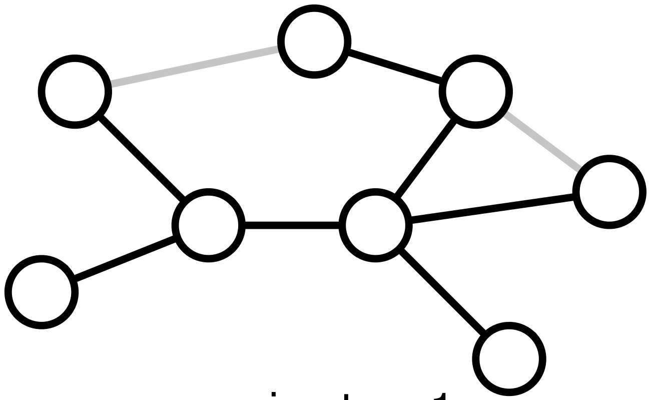
Undirected tree representation is redundant—multiplying a factor $f_{ij}(X_i, X_j)$ by $g(X_i)$, and dividing $f_i(X_i)$ by the same $g(X_i)$ does not change the distribution.

BP on tree can be understood as reparametrizing (1) by locally consistent factors. This results in (3), from which the local marginals can be read off.

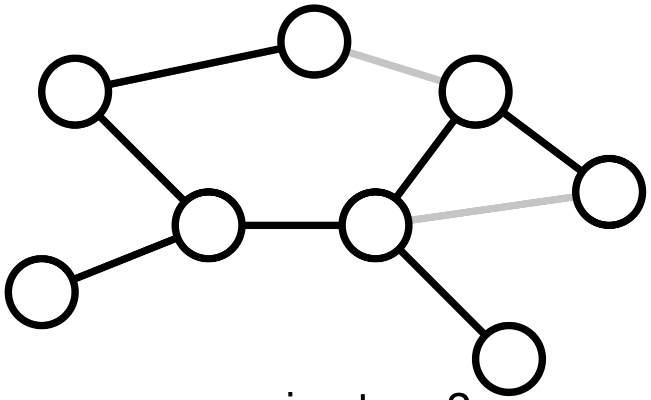
Loopy BP as Tree-based Reparametrization



graph



spanning tree 1



spanning tree 2

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} f_i^0(X_i) \prod_{\text{edges } (ij)} f_{ij}^0(X_i, X_j)$$

$$= \frac{1}{Z} \prod_{\text{nodes } i \in T_1} f_i^0(X_i) \prod_{\text{edges } (ij) \in T_1} f_{ij}^0(X_i, X_j) \prod_{\text{edges } (ij) \notin T_1} f_{ij}^0(X_i, X_j)$$

$$= \frac{1}{Z} \prod_{\text{nodes } i \in T_1} f_i^1(X_i) \prod_{\text{edges } (ij) \in T_1} f_{ij}^1(X_i, X_j) \prod_{\text{edges } (ij) \notin T_1} f_{ij}^1(X_i, X_j)$$

where $f_i^1(X_i) = p^{T_1}(X_i)$, $f_{ij}^1(X_i, X_j) = \frac{p^{T_1}(X_i,X_j)}{p^{T_1}(X_i)p^{T_1}(X_j)}$, $f_{ij}^1 = f_{ij}^0$.

$$= \frac{1}{Z} \prod_{\text{nodes } i \in T_2} f_i^1(X_i) \prod_{\text{edges } (ij) \in T_2} f_{ij}^1(X_i, X_j) \prod_{\text{edges } (ij) \notin T_2} f_{ij}^1(X_i, X_j)$$

$\ldots$

Loopy BP as Tree-based Reparametrization

At convergence, loopy BP has reparametrized the joint distribution as:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} f_i^\infty(X_i) \prod_{\text{edges } (ij)} f_{ij}^\infty(X_i, X_j)$$

where for any tree T embedded in the graph,

$$f_i^\infty(X_i) = p^T(X_i)$$
$$f_{ij}^\infty(X_i, X_j) = \frac{p^T(X_i, X_j)}{p^T(X_i)p^T(X_j)}$$

In particular, all local marginals of all trees are locally consistent with each other:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{\text{nodes } i} b_i(X_i) \prod_{\text{edges } (ij)} \frac{b_{ij}(X_i, X_j)}{b_i(X_i)b_j(X_j)}$$

Loopy BP as Optimizing Bethe Free Energy

$$p(\mathbf{X}) = \frac{1}{Z} \prod_i f_i(X_i) \prod_{(ij)} f_{ij}(X_i, X_j)$$

Loopy BP can be derived as fixed point equations for finding stationary points of an objective function called the Bethe free energy.

The Bethe free energy is not optimized wrt a full distribution over $\mathbf{X}$, rather over locally consistent pseudomarginals or beliefs $b_i \geq 0$ and $b_{ij} \geq 0$:

$$\sum_{X_i} b_i(X_i) = 1 \qquad \forall i$$

$$\sum_{X_j} b_{ij}(X_i, X_j) = b_i(X_i) \qquad \forall i, j \in \text{ne}(i)$$

Loopy BP as Optimizing Bethe Free Energy

$$\mathcal{F}_{\text{bethe}}(b) = \mathcal{E}_{\text{bethe}}(b) + \mathcal{H}_{\text{bethe}}(b)$$

The Bethe average energy is “exact”:

$$\mathcal{E}_{\text{bethe}}(b) = \sum_i \sum_{X_i} b_i(X_i) \log f_i(X_i) + \sum_{(ij)} \sum_{X_i, X_j} b_{ij}(X_i, X_j) \log f_{ij}(X_i, X_j)$$

While the Bethe entropy is approximate:

$$\mathcal{H}_{\text{bethe}}(b) = -\sum_i \sum_{X_i} b_i(X_i) \log b_i(X_i) - \sum_{(ij)} \sum_{X_i, X_j} b_{ij}(X_i, X_j) \log \frac{b_{ij}(X_i, X_j)}{b_i(X_i) b_j(X_j)}$$

Factors in denominator are to account for overcount of entropy on edges, so that the Bethe entropy is exact on trees.

Message updates in loopy BP can now derived by finding the stationary points of the Lagrangian (with Lagrange multipliers included to enforce local consistency). Messages are related to the Lagrange multipliers.

Loopy BP as Optimizing Bethe Free Energy

- Fixed points of loopy BP are exactly the stationary points of the Bethe free energy.

- Stable fixed points of loopy BP are local maximum of Bethe free enegy (note we used inverted notion of free energy to be consistent with the variational free energy).

- For binary attractive networks, Bethe free energy at fixed points of loopy BP forms lower bound on log partition function $\log Z$.

Loopy BP vs Variational Approximation

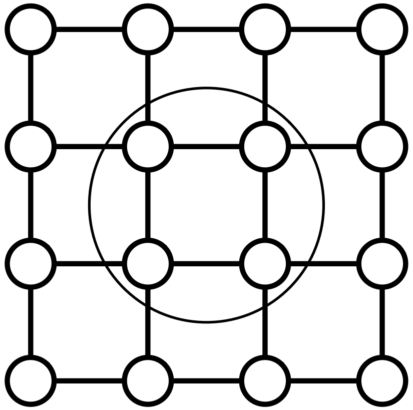
- Beliefs b_i and b_{ij} in loopy BP are only locally consistent pseudomarginals and do not necessarily form a full joint distribution.

- Bethe free energy accounts for interactions between different sites, while variational free energy assumes independence.

- The loop series or Plefka expansion of the log partition function Z: the variational free energy forms the first order terms, while Bethe free energy contains higher order terms (involving generalized loops).

- Loopy BP tends to be signficantly more accurate whenever it converges.

Extensions and Variations



- Generalized BP: group variables together to treat their interactions exactly.

- Convergent alternatives: Fixed points of loopy BP are stationary points of the Bethe free enegy. We can derive algorithms **minimizing** the Bethe free energy thus are guaranteed to converge.

- Convex alternatives: We can derive convex cousins of Bethe free energy. These give rise to algorithms that will converge to the unique global minimum.

- Treatment of loopy Viterbi or max-product algorithms is different.

A Convex Perspective

An exponential family distribution is parametrized by a natural parameter vector θ and equivalent by its mean parameter vector μ.

$$P(X|\theta) = \exp\left(\theta^\top \mathbf{T}(X) - \Phi(\theta)\right)$$

where $\Phi(\theta)$ is the log partition function

$$\Phi(\theta) = \log Z = \log \sum_x \exp\left(\theta^\top \mathbf{T}(x)\right)$$

$\Phi(\theta)$ plays an important role in the characterization of the exponential family. It is a cumulant generating function for the distribution:

$$\begin{aligned}\nabla\Phi(\theta) &= \mathbf{E}_\theta[\mathbf{T}(X)] = \mu(\theta) = \mu \\ \nabla^2\Phi(\theta) &= \mathbf{V}_\theta[\mathbf{T}(X)]\end{aligned}$$

The second derivative is positive semi-definite, so $\Phi(\theta)$ is convex in θ.

A Convex Perspective

The log partition function and the negative entropy are intimately related. We express the negative entropy as a function of the mean parameter:

$$\begin{aligned}\Psi(\mu) &= \mathbf{E}_\theta[\log P(X|\theta)] = \theta^\top \mu - \Phi(\theta) \\ \theta^\top \mu &= \Phi(\theta) + \Psi(\mu)\end{aligned}$$

The KL divergence between two exponential family distributions $p(X|\theta')$ and $p(X|\theta)$ is:

$$\begin{aligned}\mathbf{KL}(P(X|\theta)\|P(X|\theta')) =& \mathbf{KL}(\theta\|\theta') = \mathbf{E}_\theta[\log P(X|\theta) - \log P(X|\theta')] \\ =& \Psi(\mu) - (\theta')^\top \mu + \Phi(\theta') \geq 0 \\ \Psi(\mu) \geq& (\theta')^\top \mu - \Phi(\theta')\end{aligned}$$

For any pair of mean and natural parameter vectors.
Because the minimum of the KL divergence is zero, and attained at $\theta = \theta'$, we have:

$$\Psi(\mu) = \sup_{\theta'} (\theta')^\top \mu - \Phi(\theta')$$

The construction on the RHS is called the convex dual of $\Phi(\theta)$. For continuous convex functions, the dual of the dual is the original function, thus:

$$\Phi(\theta) = \sup_{\mu'} \theta^\top \mu' - \Psi(\mu')$$

The Marginal Polytope

$$\Phi(\theta) = \sup_{\mu'} \theta^\top \mu' - \Psi(\mu')$$

The supremum is only over mean parameters that can in fact be expressed as means of the sufficient statistics function:

$$\mathcal{M} = \{\mu' | \mu' = \mathbf{E}_{P'(X)}[\mathbf{T}(X)] \text{ for some distribution } P'(X)\}$$

$\mathcal{M}$ is a convex set. If X is discrete, it is a polytope called the marginal polytope.

- One view of inference is the computation of the log partition function via the maximization over μ', as well as of the maximizing mean parameters.
- There are two difficulties with the computation: optimizing over $\mathcal{M}$ is intractable, and computing the negative entropy Ψ is intractable for many models of interest.
- Many propagation algorithms can be viewed as explicit approximations to $\mathcal{M}$ and Ψ.

End Notes

Minka (2001).
A Family of Algorithms for Approximate Bayesian Inference. MIT.

R. J. McEliece, D. J. C. MacKay and J. F. Cheng (1998).
Turbo decoding as an instance of Pearl's belief propagation algorithm. IEEE Journal on Selected Areas in Communication, 16(2):140-152.

F. Kschischang and B. Frey. (1998).
Iterative decoding of compound codes by probability propagation in graphical models. IEEE Journal on Selected Areas in Communication, 16(2):219-230.

J. S. Yedidia, W. T. Freeman and Y. Weiss (2005).
Constructing free energy approximations and generalized belief propagation algorithms. IEEE Transactions on Information Theory, 51:2282-2313.

A unifying viewpoint:

M. J. Wainwright and M. I. Jordan (2008).
Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, 1:1-305.

End Notes