Belief Propagation and Bethe approximation part I

Caterina De Bacco and Isabel Valera

1 Introduction: $TrueSkill^{TM}$ Bayesian rating system

Consider the problem of rating chess players. The outcome of a game between k teams is $\mathbf{r} = (r_1, \dots, r_k)$, $r_i \in \{1, \dots, k\}$, where r_i is the rank of each team and $r_i = 1$ if i is the winner (best rank is the smallest one). Each team j is made of a subset $A_j \subset \{1, \dots, n\}$ of n players.

Let's assume that the probability $P(\mathbf{r}|\mathbf{s}, A)$ of a given outcome $\mathbf{r}$ depends on a set of hidden variables, the skills s_i of each player and the team assignments A.

Objective: infer the hidden variables $\mathbf{s}$ given the data $\mathbf{r}$ and A.

We can solve this inference problem by using a Bayesian approach as in Herbrich *et al.* (2007) and writing the posterior:

$$P(\mathbf{s}|\mathbf{r}, A) = \frac{P(\mathbf{r}|\mathbf{s}, A)\, P(\mathbf{s})}{P(\mathbf{r}|A)} \qquad \text{Posterior of the skills} \tag{1}$$

$$P(\mathbf{s}) = \prod_{i=1}^{n} \mathcal{N}(s_i|\mu_i, \sigma_i^2) \qquad \text{Prior over the skills} \quad . \tag{2}$$

We further assume that the each player has a performance centered around its skill s_i:

$$p_i \sim \mathcal{N}(p_i; s_i, \beta^2) \quad . \tag{3}$$

A team's j performance is simply the sum over the players' performances:

$$t_j = \sum_{i \in A_j} p_i \quad . \tag{4}$$

Ordering the teams in ascending performance $r_1 \leq r_2 \leq \cdots \leq r_k$ and disregarding draws, the probability of an outcome is:

$$P(\mathbf{r}|\{t_1, \dots, t_k\}) = P(t_{r_1} > t_{r_2} > \cdots > t_{r_k}) \tag{5}$$

this is the likelihood of the data in the numerator of (1). If there is a draw, then the winning outcome $r_j < r_{j+1}$ requires $t_j > t_{j+1} + \epsilon$ and the draw outcome $r_j = r_{j+1}$ requires $|t_j - t_{j+1}| \leq \epsilon$, where $\epsilon > 0$ is the draw margin.

Example. Consider a game with 3 teams, 4 players and assignments $A_1 = \{1\}$, $A_2 = \{2, 3\}$ and $A_3 = \{4\}$. Let's assume that team 1 wins, i.e. $r_1 = 1$ and the other two draw, i.e. $r_2 = r_3 = 2$. Then the joint distribution $P(\mathbf{s}, \mathbf{p}, \mathbf{t}|\mathbf{r}, A)$ is given by the factor graph in Figure 1.

Goal: model the marginal $P(s_i|\mathbf{r}, A)$ of the posterior $P(\mathbf{s}|\mathbf{r}, A)$. In other words, we want to infer the distribution of each of the single skills. The posterior is obtained by integrating out the individuals' and teams' performances:

$$P(\mathbf{s}|\mathbf{r}, A) = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} P(\mathbf{s}, \mathbf{p}, \mathbf{t}|\mathbf{r}, A)\, d\mathbf{p}\, d\mathbf{t} \quad . \tag{6}$$

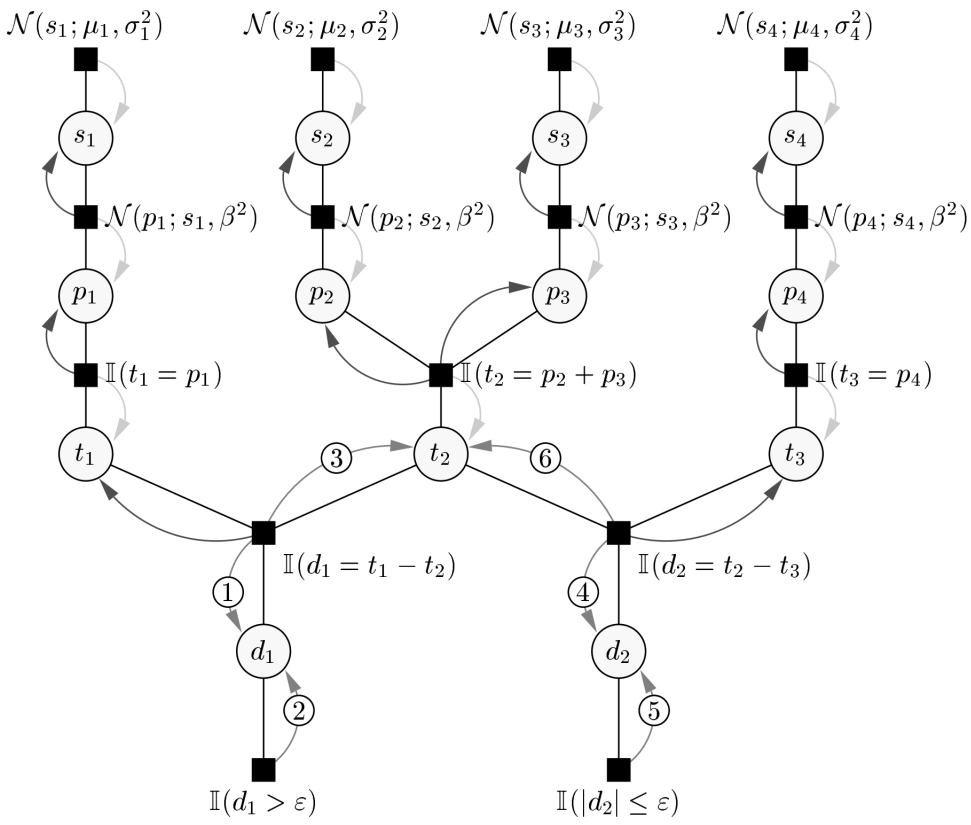

Figure 1: An example TrueSkill factor graph. There are four types of variables: s_i for the skills of all players, p_i for the performances of all players, t_i for the performances of all teams and d_i for the team performance difference. The first row of factors encode the (product) prior; the product of the remaining factors characterizes the likelihood for the game outcome Team 1 > Team 2 = Team 3. The arrows indicate the optimal message passing schedule: first, all light arrow messages are updated from top to bottom. In the following, the schedule over the team performance (difference) nodes are iterated in the order of the numbers. Finally, the posterior over the skills is computed by updating all the dark arrow messages from bottom to top. Figure taken from Herbrich *et al.* (2007).

Not only this is hard to calculate by itself, it might be even harder to then marginalize over the $\mathbf{s} \setminus j$ to get the marginal $P(s_i|\mathbf{r},A)$!
So what do we do?
We could use the tools learned so far, but they might give bad results ... we can do better.
Hold on and will find out by the end of this lecture...

2 The Bethe Approximation: Variational approach

So far, we approximated the joint probability distribution $P(\mathbf{s})$ of a system of N variables with a factorized probability distribution $Q(\mathbf{s})$, a factorization of only one-variable function:

$$Q(\mathbf{s}) = \prod_{i=1}^{N} Q_i(s_i) \quad . \tag{7}$$

This was the main idea of the Mean Field variational approach (which was then further corrected using TAP).

The next natural step to improve this approximation, which was valid only in some restricted scenarios (e.g. weak couplings), is to consider also two-variable functions $b_{ij}(s_i, s_j)$ inside the factorization. We then consider a variational distribution of the form:

$$Q_{Bethe}(\mathbf{s}) = \prod_{ij} b_{ij}(s_i, s_j) \prod_{i=1}^{N} b_i(s_i)^{1-d_i} \quad , \tag{8}$$

where d_i is the degree of node i, i.e. the number of nodes with whom i is interacting.

As before, the goal is to find the best set of b_{ij} and b_i such that it minimizes the KL divergence $KL(Q||P)$. This is equivalent to minimizing the variational free energy:

$$G_{Bethe} := F[Q_{Bethe}] = E[Q_{Bethe}] - \frac{1}{\beta} S[Q_{Bethe}] \quad , \tag{9}$$

as defined in Lecture 7.

Before calculating an expression for G_{Bethe}, let's assume that the exact joint has a particular shape:

$$P(\mathbf{s}) = \prod_{ij} P_{ij}(s_i, s_j) \prod_{i=1}^{N} P_i(s_i)^{1-d_i} \quad , \tag{10}$$

where d_i is the degree of node i, i.e. the number of variables that i interacts with. This term is there to ensure that $P(\mathbf{s})$ is correctly normalized.
$P_{ij}(s_i, s_j)$ and $P_i(s_i)$ are two-variable and one-variable *marginals* respectively. This means that they are normalized to 1, and they agree, by definition of *marginal*, to a set of **consistency** equations of the type:

$$P_{ij}(s_i, s_j) = \sum_{\mathbf{s} \setminus \{i,j\}} P(\mathbf{s}) \tag{11}$$

$$P_i(s_i) = \sum_{s_j} P_{ij}(s_i, s_j) \quad . \tag{12}$$

Consider a generic Hamiltonian for the exact model:

$$H(\mathbf{s}) = -\sum_{i<j} J_{ij} s_i s_j - \sum_i h_i s_i \quad , \tag{13}$$

from this we can derive the exact joint distribution $P(\mathbf{s}) = \frac{1}{Z} e^{-\beta H(\mathbf{s})}$ as usual.

Obs1: the expression $P(\mathbf{s})$ as in Equation (10) is an exact representation of $P(\mathbf{s}) = \frac{1}{Z} e^{-\beta H(\mathbf{s})}$ for tree structures, i.e. when there are no loops. It is a good approximation of that for *locally* tree-like structures, i.e. networks with no short loops, or, equivalently, with correlations between pairs of variables decaying fast enough.
Obs2: Equation (10) is similar to Q_{Bethe}, except the important difference that for Q_{Bethe} we did not assume that Q_{ij} and Q_i are consistent. The b_i and b_{ij} can be interpreted as *beliefs*, i.e. approximations for the marginals of the exact distribution.

Keeping in mind Equation (10), we now give the computations of the entropy and the average energy of P. First let's compute the entropy $S[P]$:

$$S[P] = -\sum_{\mathbf{s}} P(\mathbf{s}) \log(P(\mathbf{s})) \tag{14}$$

$$= -\mathbb{E}_P \Big[\sum_{i<j} \log(P_{ij}(s_i, s_j)) + \sum_i (1 - d_i) \log(P_i(s_i)) \Big] \tag{15}$$

$$= -\sum_{i<j} \sum_{s_i, s_j} P_{ij}(s_i, s_j) \log(P_{ij}(s_i, s_j)) - \sum_i (1 - d_i) \sum_{s_i} P_i(s_i) \log(P_i(s_i)) \quad . \tag{16}$$

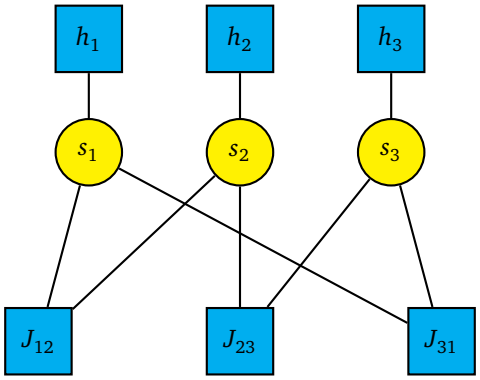

Figure 2: Factor graph example for a pairwise model with 3 variables. Function nodes a are the interaction couplings $J_{ij}(s_i, s_j)$ and external fields h_i.

Second, let's compute the internal energy $E[P]$:

$$
\begin{aligned}
E[P] &= \sum_{\mathbf{s}} P(\mathbf{s}) H(\mathbf{s}) && (17)\\
&= \mathbb{E}_P\Big[-\sum_{i<j} J_{ij} s_i s_j - \sum_i h_i s_i \Big] && (18)\\
&= -\sum_{i<j} \sum_{s_i,s_j} P_{ij}(s_i,s_j) J_{ij} s_i s_j - \sum_i \sum_{s_i} P_i(s_i) h_i s_i && (19)\\
&= -\sum_{i<j} \sum_{s_i,s_j} P_{ij}(s_i,s_j) \left[J_{ij} s_i s_j + h_i s_i + h_j s_j \right] - \sum_i (1-d_i) \sum_{s_i} P_i(s_i) h_i s_i && (20)\\
&= \sum_{i<j} \sum_{s_i,s_j} P_{ij}(s_i,s_j) E_{ij}(s_i,s_j) + \sum_i (1-d_i) \sum_{s_i} P_i(s_i) E_i(s_i) \quad , && (21)
\end{aligned}
$$

where we defined:

$$
\begin{aligned}
E_{ij}(s_i,s_j) &:= -(J_{ij} s_i s_j + h_i s_i + h_j s_j) && (22)\\
E_i(s_i) &:= -h_i s_i \quad . && (23)
\end{aligned}
$$

We developed the last step in order to write the internal energy in a more convenient way, as it will become clear later. The first term of eq. (21) is the average energy of each link, and the second term is the correction for the fact that the evidence is counted $d_i - 1$ times too many.

We now derive these functions for the above factorization (8).

$$
\begin{aligned}
G_{bethe} &= E[Q_{Bethe}] - \frac{1}{\beta} S[Q_{Bethe}] && (24)\\
&= \sum_{ij} \sum_{s_i,s_j} b_{ij}(s_i,s_j) \Big[\frac{1}{\beta} \log(b_{ij}(s_i,s_j)) + E_{ij}(s_i,s_j) \Big] + \sum_i (1-d_i) \sum_{s_i} b_i(s_i) \Big[\frac{1}{\beta} \log(b_i(s_i)) + E_i(s_i) \Big] \quad .
\end{aligned}
$$

2.1 Locally consistent marginals

Now, in order to define precisely the Bethe free energy G_{bethe}, we must consider a space of "possible" marginals, as the $P_i(s_i)$ and $P_{ij}(s_i, s_J)$ are. The starting choice is to restrict to the *locally consistent marginals*. For a factor graph $G(V, F, E)$ (see Figure 2 for a simple example) these are denoted as $LOC(G)$ and they are a collection of distributions $b_i(\cdot)$ over $\mathscr{X}$ for each variable node $i \in V$ and $b_a(\cdot)$ over $\mathscr{X}^{|\partial a|}$ for each functional node $a \in F$.

Since we want them to be distributions, they are non-negative, i.e. $b_i(s_i) \geq 0$ and $b_a(s_{\partial a}) \geq 0$ and they must satisfy the normalization conditions:

$$\sum_{s_i} b_i(s_i) = 1 \quad \forall i \in V \tag{25}$$

$$\sum_{s_{\partial a}} b_a(s_{\partial a}) = 1 \quad \forall a \in F \quad . \tag{26}$$

To be *locally consistent*, the must satisfy the following consistency condition for marginalization:

$$\sum_{s_{\partial a \setminus i}} b_a(s_{\partial a}) = b_i(s_i) \qquad \forall a \in F \, , \forall i \in \partial a \, . \tag{27}$$

Obs1: the marginals $P_i(s_i)$ and $P_{ij}(x_i, x_j)$ of any probability distribution are locally consistent. However, the converse is not true: one can find a set of locally consistent marginals that do not correspond to any probability distribution. To emphasize this point, locally consistent marginals are sometimes called **beliefs**.

Example. Consider the factor model of figure 3 on binary variables (s_1, s_2, s_3), $s_i \in \{0, 1\}$. They are locally consistent (exercise: check this), but they do not represent the marginals of any probability distribution. In other words, the beliefs are "unrealizable". Why?
Any joint probability function for this example is a $P(s_1, s_2, s_3)$ over 8 possible configuration. If we calculate, for instance, $\sum_{s_3 \in \{0,1\}} P(s_1 = 0, s_2 = 1, s_3) = b_{12}(0, 1) = 0.01$.
This implies that $P(0, 1, 0) \leq 0.01$ and $P(0, 1, 1) \leq 0.01$. Similarly for $P(1, 0, 0) \leq 0.01$ and $P(1, 0, 1) \leq 0.01$.
Repeating for the term $b_{23}(s_2, s_3)$ yields: $P(0, 0, 1) \leq 0.01$ and $P(1, 0, 1) \leq 0.01$; $P(0, 1, 0) \leq 0.01$ and $P(1, 1, 0) \leq 0.01$.
Finally for $b_{31}(s_1, s_3)$ we get: $P(0, 0, 0) \leq 0.01$ and $P(0, 0, 1) \leq 0.01$; $P(1, 1, 0) \leq 0.01$ and $P(1, 1, 1) \leq 0.01$.
This means that for every one among the 8 possible configurations $P(s_1, s_2, s_3) \leq 0.01$, which means that $\sum_{\mathbf{s}} P(\mathbf{s}) \leq 0.08 \neq 1$. This answers to our question.

Obs2: in general, the b_i and b_{ij} are unrealizable unless the network is a tree.

b

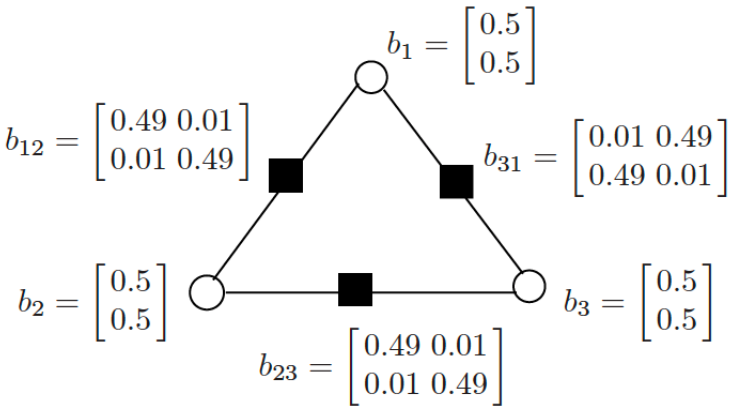

Figure 3: A set of LOC(G) that cannot arise as marginals of any global probability distribution. Figure taken from Mezard and Montanari (2009).

For the pairwise model considered above, we get that the constraints on the beliefs are:

$$\sum_{s_i} b_i(s_i) = 1 \quad \forall i \in V \tag{28}$$

$$\sum_{s_i,s_j} b_{ij}(s_i,s_j) = 1 \quad \forall ij \tag{29}$$

$$\sum_{s_j} b_{ij}(s_i,s_j) = b_i(s_i) \quad \forall s_i \tag{30}$$

$$\sum_{s_i} b_{ij}(s_i,s_j) = b_j(s_j) \quad \forall s_j \quad . \tag{31}$$

The minimization of the Eq. (24) becomes a constrained minimization problem that can be solved by introducing Lagrange multipliers γ_i and γ_{ij} which enforce the constraints:

$$L := G_{bethe} + \sum_i \gamma_i \Big(1 - \sum_{s_i} b_i(s_i)\Big) + \sum_{ij} \gamma_{ij} \Big(1 - \sum_{s_i,s_j} b_{ij}(s_i,s_j)\Big) \tag{32}$$

$$+ \sum_{ij} \sum_{s_j} \lambda_{ji}(s_j) \Big(b_i(s_i) - \sum_{s_j} b_{ij}(s_i,s_j)\Big) + \sum_{ij} \sum_{s_i} \lambda_{ij}(s_i) \Big(b_j(s_j) - \sum_{s_i} b_{ij}(s_i,s_j)\Big) \quad , \tag{33}$$

which we aim at minimizing.
This means solving the following equations:

$$\frac{\partial L}{\partial b_i(s_i)} = (1-d_i)\frac{1}{\beta}[1 + \log(b_i(s_i)] + (1-d_i)E_i(s_i) - \gamma_i + \sum_j \lambda_{ji}(s_i) \tag{34}$$

$$\frac{\partial L}{\partial b_i(s_i)} \equiv 0 \implies b_i(s_i) = \exp\left[\beta \frac{-(1-d_i)E_i(s_i) + \gamma_i - \sum_j \lambda_{ji}(s_i)}{(1-d_i)} - 1\right]. \tag{35}$$

Adding a normalization constant for enforcing the normalization condition and keeping only the Lagrange multipliers that depends on both indexes, we obtain:

$$b_i(s_i) = \frac{1}{Z_i} \exp\left[-\beta\, E_i(s_i) + \beta\, \frac{\sum_j \lambda_{ji}(s_i)}{(d_i - 1)}\right]. \tag{36}$$

With the same procedure for the derivative over $b_{ij}(s_i,s_j)$ we obtain:

$$b_{ij}(s_i,s_j) = \frac{1}{Z_{ij}} \exp\left\{-\beta\left[E_{ij}(s_i,s_j) - \lambda_{ji}(s_i) - \lambda_{ij}(s_j)\right]\right\}. \tag{37}$$

The equations (36) and (37) depend on the Lagrange multipliers. To find an expression for them, we would need to enforce the consistency equations (30) and (31), but this increases the computational time for solving this problem.

Question: how do we minimize the Bethe free energy efficiently?
In order to answer this question, we present an algorithmic tool (*Belief Propagation algorithm*) that provides a possible solution.

3 Belief Propagation

Let's forget for a moment the problem we were addressing, i.e. deriving good approximations for the joint distribution $P(\mathbf{s})$ from a factorized variational distribution $Q(\mathbf{s})$.

Consider now the problem of computing marginals of a graphical model with N variables $\mathbf{x} = (x_1, \ldots, x_N)$ taking values in a finite alphabet $\mathcal{X}$.

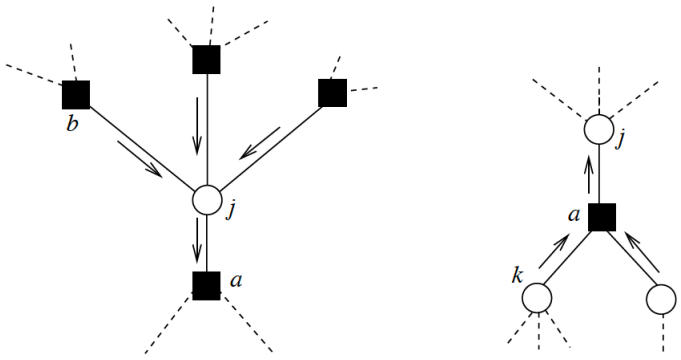

Figure 4: Example of a factor graph. Circles represent variable nodes and squares are factor nodes. Arrows represent messages illustrating the Belief Propagation equations (40) and (41) of belief propagation. Figure taken from Mezard and Montanari (2009).

Obs1: the naive algorithm, which sums over all configurations, takes a time of order $|\mathscr{X}|^N$.
Obs2: however, the complexity can be reduced dramatically when the underlying factor graph has some special structure. One extreme case is that of tree factor graphs. On trees, marginals can be computed in a number of operations which grows linearly with N.

This can be done through a 'dynamic programming' procedure that recursively sums over all variables, starting from the leaves and progressing towards the root of the tree. Remarkably, such a recursive procedure can be recast as a distributed 'message-passing' algorithm. Message-passing algorithms operate on 'messages' associated with edges of the factor graph, and update them recursively through local computations done at the vertices of the graph. *Belief Propagation (BP)* is a message-passing algorithm and it yields exact marginals on trees. The various algorithms in the message-passage family differ in the precise form of the update equations.

3.1 Belief Propagation on tree graphs

Let consider now a graphical model such that the associated factor graph is a tree (tree-graphical model). A simple example is given in Figure 4. The model describes N random variables $\mathbf{x} = (x_1, \dots, x_N)$ taking values in a finite alphabet $\mathscr{X}$, whose joint probability distribution has the form:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{a=1}^{M} \psi_a(\mathbf{x}_{\partial a}) \quad , \tag{38}$$

where $\mathbf{x}_{\partial a} = \{x_i | i \in \partial a\}$. The set $\partial a \subseteq [N]$ contains all variables involved in constraint a. We can think about ψ_a as a function which takes in input the variables in the set $\mathbf{x}_{\partial a}$. We shall always use indices $i, j, k, \dots$ for the *variable* nodes (represented as circles in factor graphs) and $a, b, c, \dots$ for the *function* nodes (represented as squares in factor graphs). The set of indices ∂i involves all function nodes a connected to i. With this framework, when the factor graph has no loops, the BP solves efficiently problems like computing the marginal distribution of one variable, or the joint distribution of a small subset of variables.

Example: for pairwise interactions, the joint probability can be written as:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{i,j} \psi_{ij}(x_i, x_j) \prod_i \phi_i(x_i) \quad , \tag{39}$$

where $\psi_{ij}(x_i, x_j)$ represent the pairwise interactions and $\phi_i(x_i)$ are local field acting on one variable.

The basic variables on which BP acts are "messages" associated with edges in the factor graph (see Figure 4). For each edge (i,a) there exist, at the t-th iteration, two messages $\nu_{i\to a}^{(t)}$ (from variable to function nodes) and $\hat{\nu}_{a\to i}^{(t)}$ (from function to variable nodes). Messages take values in the space of probability distributions over the single-variable space $\mathscr{X}$, e.g. $\nu_{i\to a}^{(t)} = \{\nu_{i\to a}^{(t)}(x_i) : x_i \in \mathscr{X}\}$, with $\nu_{i\to a}^{(t)}(x_i) \geq 0$ and $\sum_{x_i} \nu_{i\to a}^{(t)}(x_i) = 1$. As a remark on the notation, we use the index i for the variable node, while x_i indicates the state of the variable node. The message $\nu_{i\to a}^{(t)}$ is the "belief" of node i of what its state should be if function node a were to be removed, i.e. is the opinion of node i accounting for the part of the network on the other side of a.

Objective: the BP algorithm aims at computing marginal distributions. This is done efficiently by finding fixed-point values of a set of equations. It works by iteratively updating messages through local computation at the nodes of the factor graph.

Obs1: by local we mean that a given node updates the outgoing messages on the basis of *incoming* ones at previous iterations, which for sparse networks involves only a small fraction of the overall set of nodes.

The update equations for **belief propagation**, or **sum-product**, are:

$$\nu_{i\to a}^{(t+1)}(x_i) \cong \prod_{b\in\partial i\setminus a} \hat{\nu}_{b\to i}^{(t)}(x_i) \tag{40}$$

$$\hat{\nu}_{a\to i}^{(t)}(x_i) \cong \sum_{\mathbf{x}_{\partial a}\setminus i} \psi_a(\mathbf{x}_{\partial a}) \prod_{j\in\partial a\setminus i} \nu_{j\to a}^{(t)}(x_j). \tag{41}$$

The BP algorithm is called also sum-product because of the equation (41), where the sum represents the marginalization over all variables in $\partial a \setminus i$.

For pairwise models as in equation (39), we do not need two messages because equation (40) can be collapsed into equation (41). Also, function nodes (square black nodes) can be removed as they only involve two variables. The BP equations than simplify to:

$$\nu_{i\to j}^{(t+1)}(x_i) \cong \prod_{k\in\partial i\setminus j} \sum_{x_k} \psi_{ik}(x_i,x_k)\nu_{k\to i}^{(t)}(x_k) \quad . \tag{42}$$

3.1.1 Belief Propagation toy example

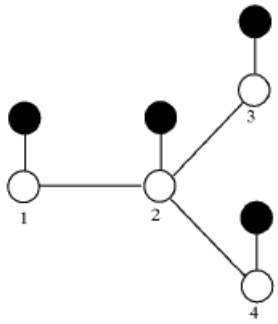

Figure 5: Example of BP algorithm on a pairwise model. Black circles represent local fields and white nodes are variables. Figure taken from Yedidia *et al.* (2003).

We will not give a proof here but rather an example to convince you that they make sense. Consider the graphical model of Figure 5 involving 4 variables and pairwise interactions with local fields. Let's first compute the belief about the marginal probability at node 1. This is a leaf, so assuming that we knew the opinion coming from node 2, i.e. message $\hat{v}^{(t)}_{2\to 1}(x_1)$, we would have a marginal:

$$b_1^{t+1}(x_1) = \frac{1}{Z_1}\phi_1(x_1)\,\hat{v}^{(t)}_{2\to 1}(x_1) \quad , \tag{43}$$

where Z_1 is a normalization. Let's now further unpack the message coming from node 2, using the information that it used to estimate it coming from the other side of the network:

$$\hat{v}^{(t)}_{2\to 1}(x_1) = \sum_{x_2}\phi_2(x_2)\,\psi_{12}(x_1,x_2)\,\hat{v}^{(t)}_{3\to 2}(x_2)\,\hat{v}^{(t)}_{4\to 2}(x_2) \quad , \tag{44}$$

and continuing like that, noticing that both node 3 and 4 are leaves:

$$\hat{v}^{(t)}_{3\to 2}(x_2) = \sum_{x_3}\phi_3(x_3)\psi_{32}(x_2,x_3) \tag{45}$$

$$\hat{v}^{(t)}_{4\to 2}(x_2) = \sum_{x_4}\phi_4(x_4)\psi_{42}(x_2,x_4) \quad . \tag{46}$$

Putting all together and reorganizing, we obtain:

$$b_1^{t+1}(x_1) = \frac{1}{Z_1}\sum_{x_2,x_3,x_4} P(\mathbf{x}) \quad , \tag{47}$$

which is exactly the marginal distribution on node 1! For tree networks, this message-passing routine gives indeed exact estimates of the marginals.

Obs2: When $\partial i \setminus a$ is empty, $v_{i\to a}(x_i)$ is the uniform distribution, i.e. $v^{(t+1)}_{i\to a}(x_i) = \frac{1}{|\mathcal{X}|}$. This is also the initial condition.
Obs3: Similarly, if $\partial a \setminus i$ is empty, then $\hat{v}_{a\to i}(x_i) = \psi_a(x_i)$.

3.2 Theoretical guarantees

A BP fixed point is a set of t-independent messages $v^{(t)}_{i\to a} = v_{i\to a}$, $\hat{v}^{(t)}_{a\to i} = \hat{v}_{a\to i}$ which satisfy equations (40) and (41). From these, one obtains $2|\mathcal{E}|$ equations, one equation for each edge of the factor graph. We shall often refer to these fixed-point conditions as the **BP equations**.

Question: how are there messages related to the exact marginals?

The answer is given by the following important theoretical results about the BP algorithm on tree graphs.

Theorem: BP is exact on trees.
Consider a tree-graphical model with diameter t_* (which means that t_* is the maximum distance between any two variable nodes). Then:

1. *Irrespective of the initial condition, the BP update equations (40) and (41) **converge** after at most t_* iterations. In other words, for any edge (ia), and any $t > t_*$, $v^{(t)}_{i\to a} = v^{*}_{i\to a}$, $\hat{v}^{(t)}_{a\to i} = \hat{v}^{*}_{a\to i}$.*
2. *The fixed-point messages provide the **exact marginals**: for any variable node i, and any $t > t_*$, $v^{(t)}_i(x_i) = P(x_i)$.*

This means that after t iterations, one can estimate the marginal distribution $P(x_i)$ of a variable i using the set of *all* incoming messages. The BP estimate of the marginal is:

$$P(x_i) = v^{(t)}_i(x_i) \cong \prod_{a\in\partial i}\hat{v}^{(t-1)}_{a\to i}(x_i) \quad , \tag{48}$$

where you should notice that the main difference between (48) and (40) is in the terms included in the product.

Question: what about other types of marginals?

The use of BP is not limited to computing one-variable marginals, but it can be used also for joint distributions. Suppose now we want the joint probability distribution $P(x_i, x_j)$ of two variables x_i and x_j. Since BP already enables to compute the marginal $P(x_i)$, we can write $P(x_i, x_j) = P(x_j|x_i)P(x_i)$ and the problem is equivalent to computing the conditional distribution $P(x_j|x_i)$. Given a model that factorizes as in eq. (38), the conditional distribution of $\mathbf{x} = (x_1, \dots, x_N)$ given $x_i = x$ takes the form

$$P(x_j|x_i = x) \cong \prod_{a=1}^{M} \psi_a(\mathbf{x}_{\partial a}) \mathbb{I}(x_i = x). \tag{49}$$

In other words, it is sufficient to add to the original graph a new function node of degree 1 connected to variable node i, which fixes $x_i = x$. One can then run BP on the modified factor graph and obtain estimates $\nu_j^{(t)}(x_j|x_i = x)$ for the conditional marginal of x_j. This can be generalized to any subset of variables, thanks to the fact that the marginal distribution of any number of variables admits an explicit expression in terms of messages for tree-graphical models. This reduces the computational time.

Let F_R be a subset of function nodes, let V_R be the subset of variable nodes adjacent to F_R, let R be the induced subgraph, and let $\mathbf{x}_R$ be the corresponding variables. Without loss of generality, we shall assume R to be connected. Further, we denote by ∂R the subset of function nodes that are not in F_R but are adjacent to a variable node in V_R. Then, for $a \in \partial R$, there exists a unique $i \in \partial a \bigcap V_R$, which we denote by $i(a)$. The joint distribution of variables in R is

$$P(\mathbf{x}_R) = \frac{1}{Z_R} \prod_{a \in F_R} \psi_a(\mathbf{x}_{\partial a}) \prod_{a \in \partial R} \hat{\nu}^*_{a \to i(a)}(x_{i(a)}), \tag{50}$$

where $\hat{\nu}^*_{a \to i}(\cdot)$ are the fixed-point BP messages.

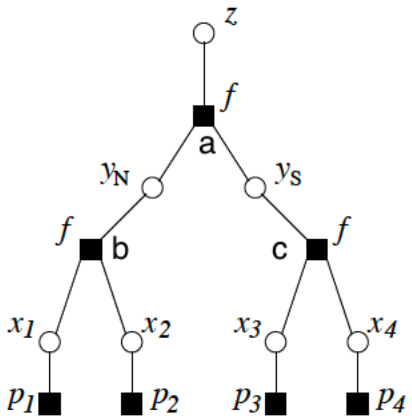

Figure 6: Factor graph of a toy Election model. Figure taken from Mezard and Montanari (2009).

Example. Let consider the factor graph in Figure 6, and let assume $F_R = \{a\}$. Then, $V_R = \{z, y_N, y_S\}$ and the induced subgraph R is given by the three circles z, y_N and y_S, connected through the factor node a. The subset $\partial R = \{b, c\}$, $i(b) = y_N$, and $i(c) = y_S$. The joint distribution of the induced graph is:

$$P(z, y_N, y_S) = \frac{1}{Z_R} \psi_a(z, y_N, y_S)\, \hat{\nu}^*_{b \to y_N}(y_N)\, \hat{\nu}^*_{c \to y_S}(y_S) \tag{51}$$

More generally, with equations (48) and (50) we are able to compute marginal and joint distributions on a tree-graphical model.

Question: what happens if we iterate BP for structures with loops, for instance the model of figure 3? BP might not converge. This is an approximation, which can be quite bad for many short loops structures. It can be improved, e.g. with Kikuchi approximation, but in general there will be no guarantee that the converged messages lead to a realizable distribution.
Exercise: check if BP messages converge for the model of figure 3 and in case to what do they converge.

A main reference for this lecture is Chapter 14 of Mezard and Montanari (2009).
The TrueSkill model is presented in Herbrich *et al.* (2007).

References

R. Herbrich, T. Minka, and T. Graepel, in *Advances in neural information processing systems* (2007) pp. 569–576.

M. Mezard and A. Montanari, *Information, physics, and computation* (Oxford University Press, 2009).

J. S. Yedidia, W. T. Freeman, and Y. Weiss, Exploring artificial intelligence in the new millennium **8**, 236 (2003).