Belief Propagation and Bethe approximation part II

Caterina De Bacco and Isabel Valera

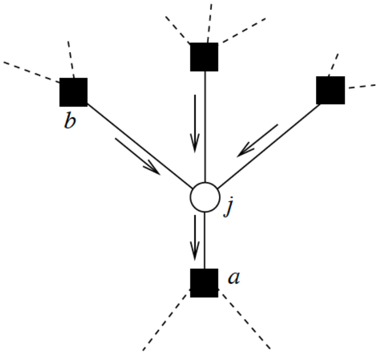
1 Belief Propagation

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 $\mathcal{X}$.

Obs1: the naive algorithm, which sums over all configurations, takes a time of order $|\mathcal{X}|^N$.
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.



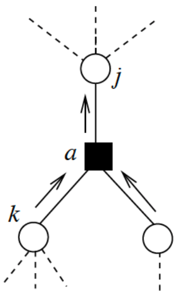


Figure 1: Example of a factor graph. Circles represent variable nodes and squares are factor nodes. Arrows represent messages illustrating the Belief Propagation equations (3) and (4) of belief propagation. Figure taken from Mezard and Montanari (2009).

1.1 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 1. The model describes N random variables $\mathbf{x} = (x_1, \dots, x_N)$ taking values in a finite alphabet $\mathcal{X}$, whose joint probability distribution has the form:

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

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:

$$P(\mathbf{x}) = \frac{1}{Z} \prod_{i,j} \psi_{ij}(x_i, x_j) \prod_{i} \phi_i(x_i) \quad , \tag{2}$$

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 1). 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 $\mathcal{X}$, e.g. $\nu_{i\to a}^{(t)} = \{\nu_{i\to a}^{(t)}(x_i) : x_i \in \mathcal{X}\}$, 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:

$$\nu_{i\to a}^{(t+1)}(x_i) \cong \prod_{b\in\partial i \setminus a} \hat{\nu}_{b\to i}^{(t)}(x_i) \tag{3}$$

$$\hat{\nu}_{a\to i}^{(t)}(x_i) \cong \sum_{\mathbf{x}_{\partial a}\setminus i} \psi_a(\mathbf{x}_{\partial a}) \prod_{j\in\partial a \setminus i} \nu_{j\to a}^{(t)}(x_j). \tag{4}$$

The BP algorithm is called also sum-product because of the equation (4), where the sum represents the marginalization over all variables in $\partial a \setminus i$.

For pairwise models as in equation (2), we do not need two messages because equation (3) can be collapsed into equation (4). Also, function nodes (square black nodes) can be removed as they only involve two variables. The BP equations than simplify to:

$$\nu_{i\to j}^{(t+1)}(x_i) \cong \prod_{k\in\partial i \setminus j} \sum_{x_k} \psi_{ik}(x_i, x_k) \nu_{k\to i}^{(t)}(x_k) \quad . \tag{5}$$

1.1.1 Belief Propagation toy example

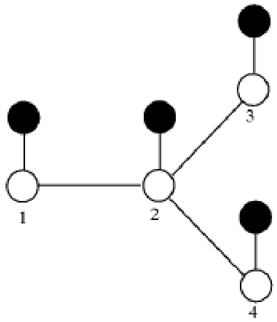


Figure 2: Example of BP algorithm on a pairwise model. Black circles represent local fields and white nodes are variables. Figure taken from Yedidia *et al.* (2003).

We will not give a proof here but rather an example to convince you that they make sense. Consider the graphical model of Figure 2 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 $\hat{\nu}^{(t)}_{2\to 1}(x_1)$, we would have a marginal:

$$b_1^{t+1}(x_1) = \frac{1}{Z_1}\phi_1(x_1)\,\hat{\nu}^{(t)}_{2\to 1}(x_1) \quad , \tag{6}$$

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:

$$\hat{\nu}^{(t)}_{2\to 1}(x_1) = \sum_{x_2}\phi_2(x_2)\,\psi_{12}(x_1,x_2)\,\hat{\nu}^{(t)}_{3\to 2}(x_2)\,\hat{\nu}^{(t)}_{4\to 2}(x_2) \quad , \tag{7}$$

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

$$\begin{aligned} \hat{\nu}^{(t)}_{3\to 2}(x_2) &= \sum_{x_3}\phi_3(x_3)\psi_{32}(x_2,x_3) & (8)\\ \hat{\nu}^{(t)}_{4\to 2}(x_2) &= \sum_{x_4}\phi_4(x_4)\psi_{42}(x_2,x_4) \quad . & (9) \end{aligned}$$

Putting all together and reorganizing, we obtain:

$$b_1^{t+1}(x_1) = \frac{1}{Z_1}\sum_{x_2,x_3,x_4} P(\mathbf{x}) \quad , \tag{10}$$

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

1.2 Theoretical guarantees

A BP fixed point is a set of t-independent messages $\nu^{(t)}_{i\to a} = \nu_{i\to a}$, $\hat{\nu}^{(t)}_{a\to i} = \hat{\nu}_{a\to i}$ which satisfy equations (3) and (4). 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 (3) and (4)* ***converge*** *after at most t_* iterations. In other words, for any edge (ia), and any $t > t_*$, $\nu^{(t)}_{i\to a} = \nu^{*}_{i\to a}$, $\hat{\nu}^{(t)}_{a\to i} = \hat{\nu}^{*}_{a\to i}$.*
2. *The fixed-point messages provide the* ***exact marginals****: for any variable node i, and any $t > t_*$, $\nu^{(t)}_{i}(x_i) = P(x_i)$.*

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:

$$P(x_i) = \nu^{(t)}_{i}(x_i) \cong \prod_{a\in\partial i} \hat{\nu}^{(t-1)}_{a\to i}(x_i) \quad , \tag{11}$$

where you should notice that the main difference between (11) and (3) 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. (1), the conditional distribution of $\mathbf{x} = (x_1, \dots, x_N)$ given $x_i = x$ takes the form

$$P(x_j|x_i = x) \cong \prod_{a=1}^{M} \psi_a(\mathbf{x}_{\partial a})\mathbb{I}(x_i = x). \tag{12}$$

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 $\nu^{(t)}_{j}(x_j|x_i = x)$ 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

$$P(\mathbf{x}_R) = \frac{1}{Z_R} \prod_{a\in F_R} \psi_a(\mathbf{x}_{\partial a}) \prod_{a\in\partial R} \hat{\nu}^{*}_{a\to i(a)}(x_{i(a)}), \tag{13}$$

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

Example. Let consider the factor graph in Figure 3, 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:

$$P(z, y_N, y_S) = \frac{1}{Z_R} \psi_a(z, y_N, y_S)\, \hat{\nu}^{*}_{b\to y_N}(y_N)\, \hat{\nu}^{*}_{c\to y_S}(y_S) \quad . \tag{14}$$

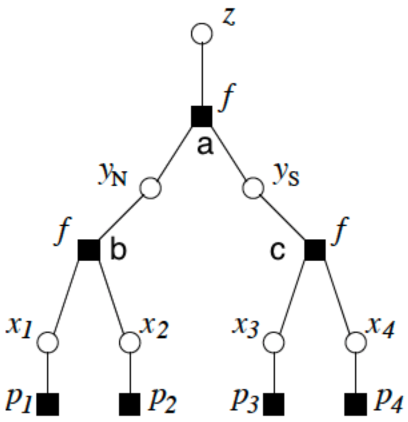


Figure 3: Factor graph of a toy Election model. Figure taken from Mezard and Montanari (2009) example 9.1. This represents a toy example of an election: a country elects its president from two candidates A and B according to the system of Figure 3. The country is divided into 4 regions $\{1,2,3,4\}$, grouped into two states: North (region 1 and 2) and South (region 3 and 4). Each of these choose their favorite candidate according to the popular vote: we denote this by $x_i \in \{A, B\}$, where $i = 1,2,3,4$. Then a north and south candidate are chosen according to this rule: if the preferences of region 1 and 2 agree, then y_N takes this value, otherwise coin flip. Same for 3,4 to chose y_S. Finally, the president z is based on a choice between y_N and y_S according to the same rule. Assume that a pooling agency estimates $p_i(x_i)$ from each region. The goal is to estimate $p(z)$.

More generally, with equations (11) and (13) 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 of Lecture 9?
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 of Lecture 9 and in case to what do they converge.

2 Belief Propagation and the Bethe Approximation

So far in this lecture we saw: i) the variational approach for representing a joint distribution with one-point and two-point marginals; ii) the BP algorithm to estimate marginals on factor graphs. We have treated them separately as they can be formulated without knowledge of the other. However, at a first glance, you could tell that they both aim at estimating marginals from intractable joint probabilities.

Question: are these two approaches related? The answer is **Yes**! And we will now see how.

Obs1: recall that the theory of the variational approach in terms of minimizing the Gibbs free energy was theoretically grounded. However it came with a computational cost, as it was expensive to perform the actual estimation of the beliefs.
Obs2: recall that BP was presented as an efficient algorithm that recovers exact marginals in trees.

Idea: if we are able to connect the two, we can link a sound theory like with an efficient algorithmic implementation!

In fact, now we will show that *assuming $\psi_a(\mathbf{x}_{\partial a}) > 0$ for every a and $\mathbf{x}_{\partial a}$, the stationary points of the Bethe free energy G_{bethe} are in one-to-one correspondence with the fixed points of the BP algorithm.*

In the settings seen so far in the previous lectures, we have that the functions in eq. (1) take the form:

$$\psi_a(\mathbf{x}_{\partial a}) = e^{-\beta H_a(\mathbf{x}_{\partial a})} \quad , \tag{15}$$

which means $H_a(\mathbf{x}_{\partial a}) = -\log(\psi_a(\mathbf{x}_{\partial a}))$, fixing $\beta = 1$.

Proof (for the interested reader). In order to prove that correspondence between stationary points of G_{Bethe} and fixed points of BP, we make a correspondence between the Lagrange multipliers and the messages, by defining:

$$\nu_{i\to(ij)}(s_i) \cong \exp\big[-\lambda_{ij}(s_i)\big] \Leftrightarrow \lambda_{ij}(s_i) \cong -\log(\nu_{i\to(ij)}(s_i)) \tag{16}$$

For the examples of the pairwise graphical model of Lecture 9 we get:

$$\begin{aligned} \hat{\nu}_{(ij)\to i}(s_i) &\cong \sum_{s_j} \psi_{ij}(s_i, s_j) e^{-\lambda_{ji}(s_j)} \\ &\cong \sum_{s_j} \psi_{ij}(s_i, s_j)\, \nu_{j\to(ij)}(s_j) \quad . \end{aligned} \tag{17}$$

Substituting the previous relationships to the stationary conditions of the Bethe variational approximation and considering $\beta = 1$ and $E_i(s_i) = 0$ (i.e. no external field), we can rewrite them as function of the BP messages:

$$b_i(s_i) \cong \prod_{j\in\partial i} \nu_{i\to(ij)}(s_i)^{\frac{1}{d_i-1}} \tag{18}$$

$$b_{ij}(s_i, s_j) \cong \psi_{ij}(s_i, s_j)\nu_{i\to(ij)}(s_i)\nu_{j\to(ij)}(s_j) \implies \sum_{s_j} b_{ij}(s_i, s_j) = \nu_{i\to(ij)}(s_i)\hat{\nu}_{(ij)\to i}(s_i) \equiv b_i(s_i). \tag{19}$$

This implies:

$$\prod_{j\in\partial i} \nu_{i\to(ij)}(s_i)^{\frac{1}{d_i-1}} \cong \nu_{i\to(ij)}(s_i)\hat{\nu}_{(ij)\to i}(s_i). \tag{20}$$

Taking the product of these equalities for $k \in \partial i \setminus j$, and eliminating $\prod_{k\in\partial i\setminus j} \nu_{i\to(ik)}(s_i)$ from the resulting equation (which is possible if $\psi_{ij}(s_i, s_j) > 0$), we get

$$\nu_{i\to(ij)}(s_i) \cong \prod_{k\in\partial i\setminus j} \hat{\nu}_{(ik)\to i}(s_i). \tag{21}$$

At this point, we recognize in equations (21) and (17) the fixed-point condition for BP (as in equations (3) and (4)). It means, that the fixed point of BP, if achieved, is the stationary point of the Bethe free energy. QED.

2.1 Solving TrueSkill inference problem

We can now answer to the initial problem about rating chess players. Look at the factor graph in Figure 4, note that this has no loops and recall that we are interested in calculating marginals $P(s_i|\mathbf{r}, A)$ of the Posterior. Now we know how to do this.

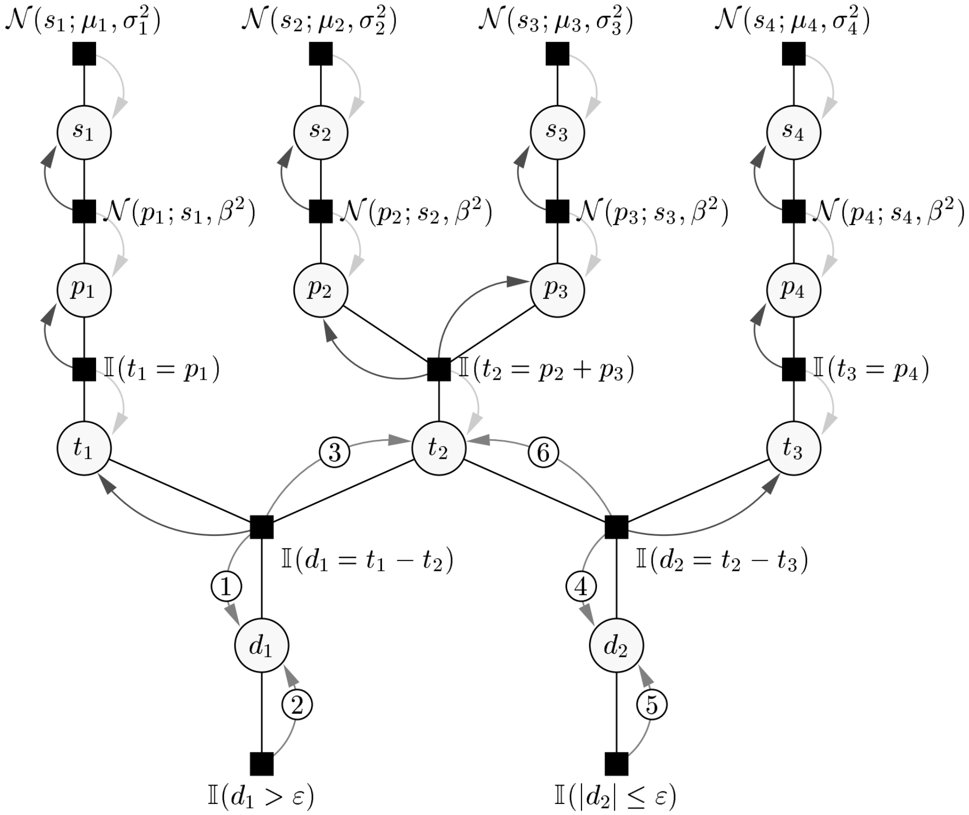

$\mathcal{N}(s_1;\mu_1,\sigma_1^2)$
$\mathcal{N}(s_2;\mu_2,\sigma_2^2)$
$\mathcal{N}(s_3;\mu_3,\sigma_3^2)$
$\mathcal{N}(s_4;\mu_4,\sigma_4^2)$
s_1
s_2
s_3
s_4
$\mathcal{N}(p_1;s_1,\beta^2)$
$\mathcal{N}(p_2;s_2,\beta^2)$
$\mathcal{N}(p_3;s_3,\beta^2)$
$\mathcal{N}(p_4;s_4,\beta^2)$
p_1
p_2
p_3
p_4
$\mathbb{I}(t_1=p_1)$
$\mathbb{I}(t_2=p_2+p_3)$
$\mathbb{I}(t_3=p_4)$
t_1
t_2
t_3
3
6
$\mathbb{I}(d_1=t_1-t_2)$
$\mathbb{I}(d_2=t_2-t_3)$
1
4
d_1
d_2
2
5
$\mathbb{I}(d_1>\varepsilon)$
$\mathbb{I}(|d_2|\leq\varepsilon)$

Figure 4: 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).

$$m_{v_k\to f}(v_k) = \prod_{\hat{f}\in F_{v_k}\setminus f} m_{\hat{f}\to v_k}(v_k) \tag{22}$$

$$m_{f\to v_j}(v_j) = \int\cdots\int f(\mathbf{v})\prod_{i\neq j} m_{v_i\to f}(v_i)\,d\mathbf{v}_{\setminus j} \tag{23}$$

$$P(v_k) = \prod_{f\in F_{v_k}} m_{f\to v_k}(v_k) \quad . \tag{24}$$

The actual computation can be done by considering that most of the messages are 1-dim Gaussians, except those of the indicator functions of the score differences d. The full picture is given by Figure 5. For instance, for the messages coming from the factor nodes of the priors (the upper ones), use the parametrization of a Gaussian using precision $\pi := \sigma^{-2}$ and precision adjusted mean $\tau := \pi\mu$, such that we can use the expression:

$$\mathcal{N}(s_i;\tau_1,\pi_1)\,\mathcal{N}(s_i;\tau_2,\pi_2) = \mathcal{N}(s_i;\tau_1+\tau_2,\pi_1+\pi_2) \tag{25}$$

2.2 Bethe approximation and BP: summary

- Bethe approximation poses a variational distribution using a factorized function with one-variable and two-variable marginals.
- Belief Propagation is an algorithm to approximate marginals in an efficient way. It is exact on tree graphical models and a good approximation for locally tree-like structures.
- On trees, i.e. graphs with no loops, the fixed-point solution of BP coincides with the distribution minimizing of the Bethe free energy.
- For structures with loops, one can introduce "high-order" beliefs, this variant is the so called *cluster variational method* or Kikuchi approximation.

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

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).

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

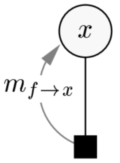
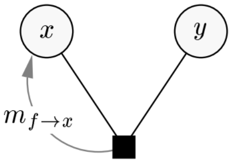
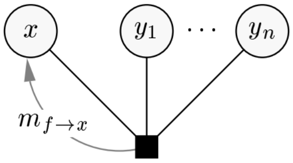
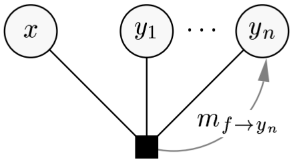
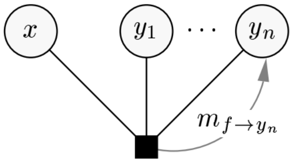
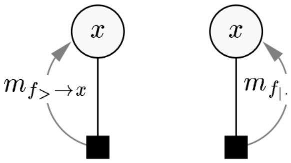
Factor	Update equation
 $\mathcal{N}(x; m, v^2)$	$\pi_x^{\mathrm{new}} \leftarrow \pi_x + \frac{1}{v^2}$ $\tau_x^{\mathrm{new}} \leftarrow \tau_x + \frac{m}{v^2}$
 $\mathcal{N}(x; y, c^2)$	$\pi_{f \to x}^{\mathrm{new}} \leftarrow a \left(\pi_y - \pi_{f \to y}\right)$ $\tau_{f \to x}^{\mathrm{new}} \leftarrow a \left(\tau_y - \tau_{f \to y}\right)$ $a := \left(1 + c^2 \left(\pi_y - \pi_{f \to y}\right)\right)^{-1}$ $m_{f \to y}$ follows from $\mathcal{N}\left(x; y, c^2\right) = \mathcal{N}\left(y; x, c^2\right)$.
 $\mathbb{I}(x = \mathbf{a}^\top \mathbf{y})$	$\pi_{f \to x}^{\mathrm{new}} \leftarrow \left(\sum_{j=1}^{n} \frac{a_j^2}{\pi_{y_j} - \pi_{f \to y_j}}\right)^{-1}$ $\tau_{f \to x}^{\mathrm{new}} \leftarrow \pi_{f \to x}^{\mathrm{new}} \cdot \left(\sum_{j=1}^{n} a_j \cdot \frac{\tau_{y_j} - \tau_{f \to y_j}}{\pi_{y_j} - \pi_{f \to y_j}}\right)$
 $\mathbb{I}(x = \mathbf{b}^\top \mathbf{y})$	 $\mathbb{I}(y_n = \mathbf{a}^\top [y_1, \cdots, y_{n-1}, x])$ $\mathbf{a} = \frac{1}{b_n} \cdot \begin{bmatrix} -b_1 \\ \vdots \\ -b_{n-1} \\ +1 \end{bmatrix}$
 $\mathbb{I}(x > \varepsilon)$ $\mathbb{I}(\lvert x \rvert \leq \varepsilon)$	$\pi_x^{\mathrm{new}} \leftarrow \frac{c}{1 - W_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}$ $\tau_x^{\mathrm{new}} \leftarrow \frac{d + \sqrt{c} \cdot V_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}{1 - W_f\left(d/\sqrt{c}, \varepsilon\sqrt{c}\right)}$ $c := \pi_x - \pi_{f \to x}\,, \quad d := \tau_x - \tau_{f \to x}$

Figure 5: The final BP messages of TrueSkill. 1-dim Gaussian messages are represented by their natural parameters precision $\pi = \sigma^{-2}$ and precision adjusted mean $\tau = \pi\mu$. Figure taken from Herbrich *et al.* (2007).