Contents

1 Tutorial 7: Bethe approximation and BP

1.1 Exercise 1: graph coloring and BP

Coloring is a classical problem of graph theory. Given a (unweighted, undirected) graph $G(V, E)$ a coloring $M \subseteq V$ is an assignment of labels, called colors, to the vertices of a graph such that no two adjacent vertices share the same color.
The problem can be tackled using a set of spins $\{s_i\}$ taking values in the set $s_i \in \{1, ..., q\}$ for each $i = 1, ..., N = |V|$. In particular, we can represent the spins configurations considering the relaxed probability distribution (derived in tutorial 6):

$$P(\mathbf{s}) = \frac{1}{Z} \prod_{(ij)\in\tilde{F}} \psi_{ij}(s_i, s_j) = \frac{1}{Z} \prod_{(ij)\in E} \mathrm{e}^{-\beta \mathbb{I}(s_i = s_j)} \quad ,$$

with $\tilde{F}$ set of factor nodes representing the interaction between spins in the coloring problem.
In this tutorial, we want to complete the following tasks:

Part I

1. Using BP to model marginals of the coloring assignment, denote as:
   - $\nu_{s_i}^{(ij)\to i}$ the messages from function node (ij) to variable node i.
   - $\chi_{s_i}^{i\to(ij)}$ the message from variable node i to function node (ij).

   Note that they are both functions of the state s_i of the variable node i.
   Write BP equations for this model.
2. Find a fixed point of these equations. (PS: recall that there might be more than one fixed point)
   Hint: what would a random guess do?
3. Write the equation for the one-point marginal $P(s_i)$ and the two-point marginal $P(s_i, s_j)$ obtained from BP.

1. According to BP rule we have:

$$\begin{aligned}
\chi_{s_j}^{j\to(ij)} &= \frac{1}{\tilde{Z}^{j\to(ij)}} \prod_{k\in\partial j\setminus i} \nu_{s_j}^{(kj)\to j} \\
\nu_{s_i}^{(ij)\to i} &= \frac{1}{\tilde{Z}^{(ij)\to i}} \sum_{s_j} \psi_{ij}(s_i, s_j)\, \chi_{s_j}^{j\to(ij)} \\
&= \frac{1}{\tilde{Z}^{(ij)\to i}} \left[\psi_{ij}(s_i, s_i)\, \chi_{s_i}^{j\to(ij)} + \sum_{s_j\neq s_i} \psi_{ij}(s_i, s_j)\, \chi_{s_j}^{j\to(ij)} \right] \\
&= \frac{\mathrm{e}^{-\beta}\chi_{s_i}^{j\to(ij)} + \sum_{s_j\neq s_i} \chi_{s_j}^{j\to(ij)}}{\tilde{Z}^{(ij)\to i}} \\
&= \frac{1 - \left(1 - \mathrm{e}^{-\beta}\right)\chi_{s_i}^{j\to(ij)}}{\tilde{Z}^{(ij)\to i}}
\end{aligned}$$

where $\tilde{Z}^{j\to(ij)}$ and $\tilde{Z}^{(ij)\to i}$ are the normalization constant of $\chi_{s_j}^{j\to(ij)}$ and $\nu_{s_i}^{(ij)\to i}$ respectively.

In the last passage of the chain of identities above we invoked the normalization of the messages $\chi_{s_i}^{j\to(ij)}$, i.e. $\sum_{s_j}\chi_{s_j}^{j\to(ij)}=1,\forall i\in V,\,(ij)\in F$.
Hence, it is sufficient to only use one set of BP messages, here we choose χ's. This simplifies the equations and results in:

$$\begin{aligned}\chi_{s_j}^{j\to(ij)} &\cong \prod_{k\in\partial j\setminus i}\left[1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}\right]\\ &= \frac{1}{\tilde{Z}^{j\to i}}\prod_{k\in\partial j\setminus i}\left[1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}\right]\end{aligned}$$

Observations:

- The quantity $1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}$ is the probability that neighbor k is fine with j taking color s_j.
- The quantity $\prod_{k\in\partial j\setminus i}\left[1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}\right]$ is the probability that all the neighbors are fine that j taking color s_j (if i was excluded and (ij) erased).

2. A fixed point is the uniform probability $\chi_{s_j}^{k\to(kj)}=\frac{1}{q}$. In fact, substituting inside

$$\chi_{s_j}^{j\to(ij)} = \frac{1}{\tilde{Z}^{j\to i}}\prod_{k\in\partial j\setminus i}\left[1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}\right]$$

and calculating the normalization function explicitly, we get:

$$\frac{\left[1-\left(1-\mathrm{e}^{-\beta}\right)\frac{1}{q}\right]^{d_j-1}}{q\left[1-\left(1-\mathrm{e}^{-\beta}\right)\frac{1}{q}\right]^{d_j-1}}=\frac{1}{q}$$

3. The marginals are:

$$\begin{aligned}\chi_{s_j} &= \frac{1}{Z^{(i)}}\prod_{k\in\partial j}\left[1-\left(1-\mathrm{e}^{-\beta}\right)\chi_{s_j}^{k\to(kj)}\right]\\ \chi_{s_j,s_j} &= \frac{1}{Z^{(ij)}}\psi_{ij}\left(s_i,s_j\right)\chi_{s_j}^{j\to(ij)}\chi_{s_i}^{i\to(ij)}\end{aligned}$$

which are valid at convergence.

Part II

Now, we want to code belief propagation for Erdös-Rényi graphs coloring for $\beta \to \infty$.

1. Initialize BP close to be uniform fixed point, i.e. $1/q + \varepsilon_s^{j\to i}$ and iterate the equations until convergence. Define converge as the time when the

$$\frac{1}{2qM} \sum_{(ij)\in E} \sum_s \left| \left(\chi_s^{i\to(ij)}(t+1) - \chi_s^{i\to(ij)}(t) \right) \right| < \tilde{\varepsilon}$$

   with suitably chosen small $\tilde{\varepsilon} > 0$.
2. Check how the behavior depends on the order of update, i.e. compare what happens if you update all messages at once or randomly one by one.
3. For parameters where the update converges, plot the convergence time as a function of the average degree c. Do this enlarging the graphs as it is feasible with your code.
4. Assign one color to each node at convergence, based on the argmax of the one-point marginals. Count how many violations you get over $N_{\text{real}} = 100$ random initializations of the graph and plot them as a function of c.
5. Check how the convergence behavior depends on the initialization.
   What if initial messages are random?
   What if they are all points towards the first color?

1. In the code, we generate an Erdös-Rényi graph with $N = 100$ nodes and average degree $c = 5$. We fix the number of colors $q = 3$ and $\beta = 2$. We use the directed version of the generated graph, so that each message is saved as an attribute of a single edge. The BP messages are initialized as $1/q + \varepsilon_s^{j\to i}$ where $\varepsilon_s^{j\to i} \sim$ Uniform $[-\alpha/q, \alpha/q]$, with $\alpha = 0.1$. Since the entries of the vector with the messages represent probabilities, they have to be normalized so that they sum to 1. The stopping threshold $\tilde{\varepsilon}$ is chosen to be 10^{-4}, and in the Jupyter this is codified by the variable `abs_tol`. As a reminder, $|E|$ is the number of edges of the original graph, so we have to divide by 2 the number of edges of the directed graph.

2. If we update BP messages parallelly, BP will not converge in `max_it` $= 1000$ iterations, but if we update BP messages randomly one by one, BP converges in around 30 iterations. Figure 1 shows the error (`err`) as function of the number of iterations for the parallel and the random update. The parallel version updates all messages at once and doesn't use the improvements for computing the messages at time $(t + 1)$. The plot on the left reflects this phenomena showing how `err` doesn't decrease as the number of iterations increases. On the other hand, the randomized version includes the updated version of some messages for computing the update of the ramaining ones. This accelerate the convergence. In the code, we consider a permutation of the edges in order to start the randomization from different messages.

3. We use $q = 3, \beta = 2$ and $N \in [50, 100, 200]$. We choose 50 values of $c \in [0.1, 7]$ and we draw 5 random graphs for each c. The update of BP is randomly one by one and for each c we save the median of the converge number of iterations. Figure 2 shows the median convergence time of these five trials as a function of the average degree for all choices of N. As the average degree increases, the problem becomes more difficult to solve and the BP doesn't converge. This is reflected by the jump around $c = 6$. Remember that we are using 3 colors, and when the average degree is higher then q, the constraint of having neighbours with a different colors is much more difficult to satisfy. Moreover, as N increases, the jump appears earlier because the problem is more difficult.

4. We use $q = 3, \beta = 2, N = 500$ and we choose 30 values of $c \in [2, 7]$. For each c we draw 5 random graphs and we update BP randomly one by one. A violation represents the case when two neighbours have the same color, and the total number of errors is normalized over the number of edges of the undirected graph for allowing the comparison between graphs with different number of edges. We save the mean and the standard deviation of the fraction

of violations and Figure 3 shows the result. As expected, the fraction of violations increases as the average degree increases, because the number of neighbours becomes bigger than the number of colors.

5. We fix $N = 1000, c = 5, q = 3$ and $\beta = 2$. We use the generated graph with three different initializations: small perturbation around the fixed-solution $1/q$, uniformly random and point mass on first color. Figure 5 shows the err as a function of the number of iterations for all cases. If we start around the fix point $1/q$, BP converges faster and `err` is quite small. The random initialization is the slowest to converge and the first-color curve has the biggest jump.

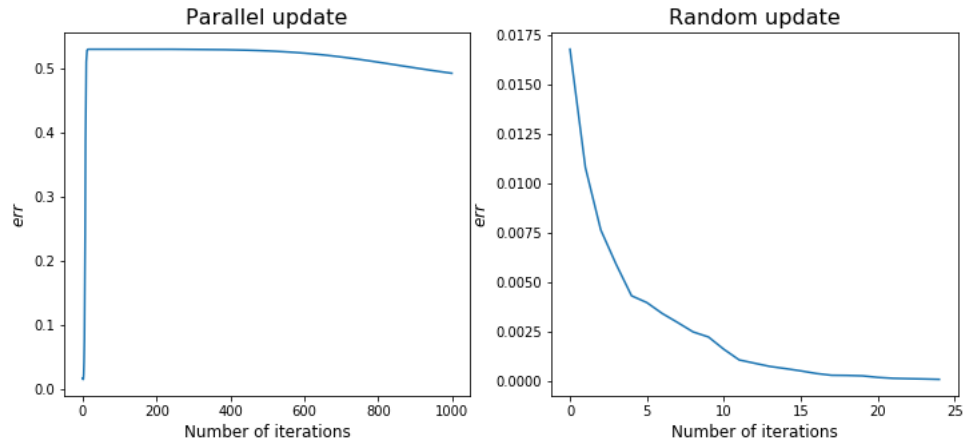

Figure 1: error value under different BP update for a graph with $N = 100$ and $c = 5; q = 3$ and $\beta = 2$.

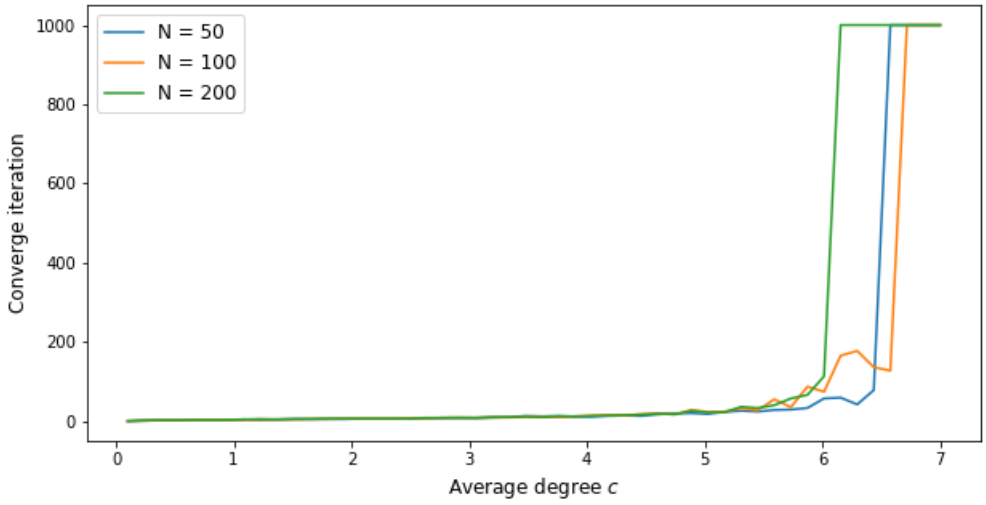

Figure 2: Convergence time vs. c for $N = 50, 100, 200, q = 3$ and $\beta = 2$.

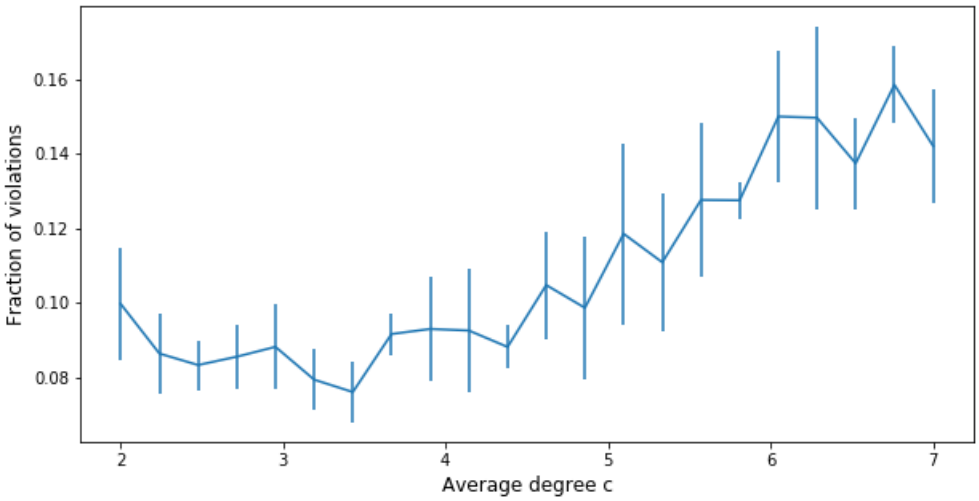

Figure 3: Fraction of violations vs. c for $N = 500, q = 3$ and $\beta = 2$. We plot the mean over 5 independent realizations and the bars represent the standard deviation.

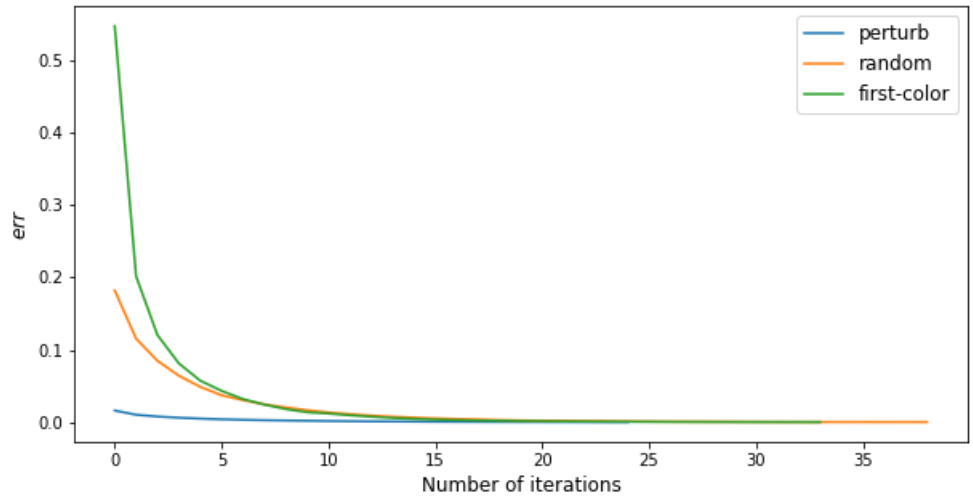

Figure 4: error value of BP with three different initializations.