ISSN 1342-2812

Research Reports on Mathematical and Computing Sciences

Belief Propagation and Spectral Methods

Masaki Yamamoto and Osamu Watanabe

Nov. 2007, C–248

Department of
Mathematical and
Computing Sciences
Tokyo Institute of Technology

SERIES C: Computer Science

Belief Propagation and Spectral Methods[*]

Masaki Yamamoto[†] Osamu Watanabe[‡]

Research Report C-248

Abstract

We investigate an algorithm derived based on the belief propagation method of Pearl [11] applied to the (Min-)Bisection problem under the standard planted solution model (or more precisely the Most Likely Partition problem under the same planted solution model). We first point out that the algorithm (without thresholding) is nothing but the standard power method for computing an eigenvector with the largest eigenvalue used by some spectral method for the Bisection problem. We then show that the thresholding helps to improve an approximate solution (by the spectral method) to the exact solution. Through our analysis, we prove that, at least for the Bisection problem, the belief propagation can be regarded as a unified spectral type algorithm for obtaining the exact solution with high probability.

1 Introduction

The purpose of this work is to give some theoretical analysis to an algorithm designed by the belief propagation method, or more specifically, "Pearl's belief propagation" [11]. Such an algorithm will be called *the belief propagation* in the following.

Roughly speaking, the belief propagation is a way to compute the marginal probability of a state at each node in a given Bayesian network. It computes iteratively the "belief" of each state; that is, it updates these beliefs in parallel at every round until all the beliefs get converged. It is shown [11] that this process converges in a constant number of rounds and that the correct marginal probabilities can be computed from the obtained beliefs *provided* that the Bayesian network has no cycle. Such a convergence cannot be guaranteed in general Bayesian networks. Nevertheless, researchers have started using the belief propagation for various problems that can be modeled by Bayesian networks with cycles, and surprisingly, it has been shown experimentally that the belief propagation works in many cases; see, e.g., [9, 10, 8] for decoding problems and, e.g., [3] for the SAT and related problems. For such success reports of the belief propagation on cyclic Bayesian networks, several attempts have been also made to give some theoretical justification to such applications. For example, Luby etal [8] proved a certain average-case performance of a message passing algorithm that is similar to but simpler than the belief propagation when applied to the LDPC decoding problem; Feige, Mossel, and Vilenchik [6] analyzed a message passing algorithm as

[*]This research is supported in part by Grant-in-Aids for Scientific Research on Priority Areas "New Horizons in Computing" and "Deepening and Expansion of Statistical Mechanical Informatics", and by JSPS Global COE program "Computationism As Foundations of Sciences".

[†]Grad. School of Informatics, Kyoto Univ., Kyoto 606-8501, Japan (masaki.yamamoto@kuis.kyoto-u.ac.jp)

[‡]Dept. of Math. and Comput. Sci., Tokyo Inst. of Tech., Tokyo, 152-8552, Japan (watanabe@is.titech.ac.jp)

a model of "survey propagation", an extension of the belief propagation, and give some theoretical justification that it works for the 3CNF SAT problem on average. It has been still open, however, to prove that the original belief propagation indeed works for some problem.

Here we consider the (Min-)Bisection problem and investigate the average-case performance of the belief propagation applied to this problem. For the distribution for discussing the average-case analysis, we consider the standard planted solution model [7]. Onsjö and Watanabe [12] derived an algorithm — `pseudo-bp` (the one given in Figure 1) — for the Bisection problem (under the planted solution model) following the belief propagation recipe of [10]. Though some modification has been made for simplifying the algorithm, detail parameter values have been kept from the original belief propagation (see the next section and Appendix for the detail). While they gave some very preliminary analysis on the performance of `pseudo-bp`, the full and general analysis has been left open. Here we first point out that their derived algorithm (if we ignore thresholding) is indeed the power method for computing an eigenvector with the largest eigenvalue. Thus, by using a somewhat standard spectral method [1, 4], we can show that the algorithm (without the thresholding) computes a good approximation of the target portion with high probability. While it is sometimes hard to analyze the nonlinear computation with the thresholding, we can again use some technique from [1, 4] to prove that the thresholding indeed helps to clean up the approximate solution to get the exact solution. It should be mentioned that for the sake of analysis the algorithm is further modified to a simpler two phase algorithm (see the next section), we may expect that its execution is similar to `pseudo-bp` if an appropriate initial value is used. Thus we may claim that this is the first concrete example such that the belief propagation applied to some nontrivial problem is justified theoretically.

From the view point of spectral methods, it has been open to develop a simple method for obtaining exact solutions after obtaining approximate solutions by a spectral method. Here we show, on this simple example, that a message passing algorithm with a certain thresholding can be used as an unified algorithm for obtaining exact solutions.

Problem and Average-Case Scenario

For our example problem, we consider here the following simple (*Min-*)*Bisection problem*: Given an undirected graph $G = (V, E)$ over $2n$ vertices, find a minimum bisection of G, that is, a partition $[V_L, V_R]$ of V such that $|V_L| = |V_R|$ and $|(V_L \times V_R) \cap E|$ is minimum. Throughout this paper, we consider graphs with $2n$ vertices and we use $V = \{1, \ldots, 2n\}$ to denote the vertex set.

Although this problem is NP-hard, since the seminal work of Boppana [2], many researchers have shown algorithms that perform well on this problem *on average*, and the planted solution model of [7] has been often used for defining an input instance distribution. Here we follow this model and, for given probability parameters $r < p$, consider the following distribution of input graphs for the problem: Given n for specifying the set V of $2n$ vertices, and given equal size partition $[V_L, V_R]$ of V, we generate, for every pair (i, j), an edge (i, j) with probability p if i and j are in the same partition, and with probability r if i and j are in the different partition. A self-loop (i, i) is also generated with probability p, for simplifying our presentation. The given partition $[V_L, V_R]$ is called a *planted solution*. We assume that input graphs are generated in this way from some planted solution, and in the following, we use $\mathcal{G}_n(p, n)[V_L, V_R]$ to denote the probability distribution of generated graphs (of $2n$ vertices). Without losing generality, we may fix V_L and V_R to $\{1, \ldots, n\}$ and $\{n + 1, \ldots, 2n\}$, in which case the distribution is simply denoted by $\mathcal{G}_n(p, r)$ or $\mathcal{G}_n$.

Under this planted solution model, the Bisection problem can be regarded as the following more natural statistical inference problem: Given an undirected graph $G = (V, E)$ over $2n$ vertices and parameters p and r, find a partition $[V_+, V_-]$ that gives the max. probability of generating G

under the planted solution model. Precisely, the partition maximizing the following probability:

$$\Pr\{G|V_+, V_-\} \;=\; \prod_{(i,j)\in E} p^{[a_i=a_j]} r^{[a_i\neq a_j]} \cdot \prod_{(i,j)\notin E} (1-p)^{[a_i=a_j]}(1-r)^{[a_i\neq a_j]}, \tag{1}$$

where $a_i = +1$ (resp., -1) if vertex i belongs to V_+ (resp., V_-), and $[\cdots]$ takes 1 if $\cdots$ holds and 0 otherwise. Intuitively, this max. probability partition is the most likely partition for the observed G. Thus, we may call this the *Most Likely Partitioning problem* (in short, the MLP problem). Note that for the MLP problem we do not require that a solution partition is of equal size. We may also consider the parameterless version where neither p nor r is given; we will also show a simple way of adjusting our algorithm for solving this general problem.

It has been shown, see, e.g., [4] that if $p - r$ is large enough (e.g., $p - r = \Omega(\log n/n)$), then the planted solution is the unique optimal solution for the Bisection problem with high probability. Also it is not hard to show [13] a similar claim for the MLP problem; for example, if $p > 4r$ and $p = \Omega(\log n/n)$, then the planted solution (which is an equal size partition) is the unique optimal solution for the MLP problem with high probability. Thus, under a certain range of parameters p and r, two problems are asking for essentially the same solution.

2 Algorithms and Our Result

We state algorithms that we analyze and give the precise statement of our analysis. Here and in the following analysis, we consider the MLP problem (for given p and r) where graphs are generated from the fixed planted solution $V_L = \{1, \ldots, n\}$ and $V_R = \{n+1, \ldots, 2n\}$. We use $+1$ and -1 (either computed or target) classification labels. The target label of each vertex i is denoted as $\boldsymbol{e}(i)$, where we fix labels as $\boldsymbol{e}(1) = \cdots = \boldsymbol{e}(n) = +1$ and $\boldsymbol{e}(n+1) = \cdots = \boldsymbol{e}(2n) = -1$. Clearly, algorithms does not know these labels and it is their task to compute these labels. But without losing generality, we may assume that the label of vertex 1 is fixed to $+1$, and algorithms can make use of this knowledge.

First we state in Figure 1 the algorithm that is derived by the belief propagation method (see Appendix for the derivation of the algorithm).

We explain the outline of the algorithm. The algorithm computes "belief" for each vertex in V, whose sign means ± 1 classification label of the vertex and whose absolute value means the confidence or the belief for this classification. Initially, all beliefs are set 0 except for b_1 that is assigned some large positive value, meaning that we *know* that $a(1) = +1$. Then beliefs are updated in rounds. Let us see this belief update focusing on some vertex i. The updating formula can be interpreted as follows: at each round (i) its current belief b_i is propagated to all the other vertices j, which we call a "message" from i to j, (ii) a message to j has the same sign as b_i for $j \in N_i$ and has the opposite sign for $j \notin N_i$, and (iii) the new belief b_i is computed by summing up those messages to b_i. This update process is executed *in parallel* on all vertices. The algorithm terminates when all beliefs get stabilized. We may consider several different conditions for stabilization, but here we simply consider the situation when the algorithm is executed for enough number (specified by $T_{\max}$) of updating rounds.

We remark on two simplifications introduced when deriving our algorithm from the original belief propagation. First one is on the updating rule. In the belief propagation, for computing a message from vertex i to vertex j, a message coming from j at the previous round is not used. That is, at each vertex i, slightly different beliefs b_{ij} are computed for every $j \in V$, where b_{ij} is computed by messages from vertices other than j and it is used to compute the next round message to j (from i). But, for the efficiency and the simplicity of the algorithm, we ignore this difference, thereby

```
procedure pseudo-bp ($G$, $p$, $r$);
begin
  set all $b_i$ to 0;
  $b_1 \leftarrow +\infty$;
  repeat $T_{\max}$ times do {
    for each $i \in V$ do in parallel {
      $b_i \leftarrow \sum_{j \in N_i} f_+(b_j) - \sum_{j \notin N_i} f_-(b_j)$;
    }
    if all $b_i$'s get stabilized then break;
  }
  output $(+1, \mathrm{sg}(b_2), ..., \mathrm{sg}(b_{2n}))$;
end-procedure
```

parameters & functions (for $r < p < 0.5$)

$$h_- = \frac{p+r}{2}, \qquad h_+ = \frac{2-(p+r)}{2},$$

$$c_- = \frac{1-p}{1-r}, \qquad c_+ = \frac{p}{r},$$

$$\theta_- = \frac{4h_-h_+^2(-\ln c_-)}{(p-r)^2}, \quad \theta_+ = \frac{4h_+h_-^2 \ln c_+}{(p-r)^2},$$

$$f_\pm(x) = \begin{cases} h_\pm \cdot \theta_\pm, & \text{if } \theta_\pm < x, \\ h_\pm \cdot x, & \text{if } -\theta_\pm \le x \le \theta_\pm, \\ -h_\pm \cdot \theta_\pm, & \text{if } x < -\theta_\pm, \end{cases}$$

(By, e.g., $f_\pm$, we mean f_+ and f_- resp.)

$\mathrm{sg}(z)$ = the sign of z, and

N_i = the set of v_i's neighbors.

Figure 1: The belief propagation based algorithm for the MLP problem

deriving our simplified updating rule. We should mention here that this simplification may not be appropriate for some applications; but at least for the MLP problem (or the Bisection problem) no essential difference can be found by this simplification from our preliminary computer experiments. The second one is a minor simplification for the definitions of functions f_+ and f_-. Originally these are sigmoid functions, but we approximate them to linear functions with thresholds.

Now we simplify this algorithm slightly more for the sake of our analysis. First we ignore the difference between two threshold values θ_+ and θ_-; let θ for this unified threshold. Then the updating formula of `pseudo-bp` can be simplified as

$$b_i = \sum_{j \in N_i} f_+(b_j) - \sum_{j \notin N_i} f_-(b_j) = \sum_{j \in N_i} h_+ \cdot b'_j - \sum_{j \notin N_i} h_- \cdot b'_j, \tag{2}$$

where b'_i is $\min(b_i, \theta)$ if b_i is positive and $\max(b_i, -\theta)$ if b_i is negative. These bounded beliefs b'_i can be computed by thresholding updated beliefs b_i at the end of each round. It turns out that the actual value of θ is not essential for our analysis so long as it is large enough relative to the initial value of b_1. In the following we simply assume that θ is some large constant and we set $b_1 = 1$ initially.

Next we assume that the thresholding is not applied for some number of initial rounds. By executing the algorithm we observe that most beliefs take similar (absolute) values after enough number of rounds. In particular, the following phenomenon is usually observed: Almost no belief reaches the threshold θ or $-\theta$ up to some Tth round, and then at the Tth round, many beliefs suddenly exceed the threshold. Since we have not been able to prove such a sharp behavior, for the sake of our analysis, we consider the modification of the algorithm so that the thresholding is not applied for first T_0 rounds for a given parameter T_0.

We analyze the algorithm obtained by these two simplifications; let us refer this algorithm as `pseudo-bp-for-analysis`. Our main technical result is to show the following performance of `pseudo-bp-for-analysis` under the planted solution model. (Throughout this paper, we say that an event occurs *with high probability* (abbreviated as **whp**) if the probability that the event occurs approaches to one as $n \to \infty$.)

Theorem 2.1. *For some sufficiently large constant $c > 0$, let p and r be any parameters such that $p, r \ge c \log n/n$ and $p/r \ge c$, and let n be any sufficiently large number. Let $d = pn$. Consider the* *execution of* `pseudo-bp-for-analysis` *on $G \in \mathcal{G}_n(p,r)$ with $T_0 = \Omega(\log n/\log d)$ and $T_{\max} = \log n$. Then it yields the planted solution of G* ***whp****.*

Remark 2.1. *From technical reason, we need to assume that $(*)$ $|\widehat{\boldsymbol{e}}_1(i) - \boldsymbol{e}_1(i)| \le \epsilon/(8\sqrt{n})$ holds on vertex $i = 1$ for some ϵ (see the next section for the notations). While this bound may not hold on vertex 1, we can surely find such a vertex i for which $(*)$ holds from the first $\ell = o(n)$ vertices. Thus, very precisely speaking, we need to execute* `pseudo-bp-for-analysis` *from $\ell = o(n)$ different initial values.*

Remark 2.2. *Though we assume here that r also satisfies $r \ge c\log n/n$. But the case where $r < c\log n/n$ can be handled by a similar and in fact a rather easier way. Thus, this condition is not necessary (see Remark 3.1).*

3 The analysis for our algorithm

In this section, we prove Theorem 2.1. We first prepare some notations used throughout this section. Fix parameters p and r satisfying the condition of the theorem, and let $d = pn$ and $d' = rn$. We use a vector $\boldsymbol{b}$ to denote the value of beliefs $(b_1, \ldots, b_{2n})$ computed in the algorithm; in particular, let $\boldsymbol{b}^{(k)}$ be the beliefs after the kth round. For each vertex i, let $\boldsymbol{b}(i)$ denote its ith component. Recall that $\boldsymbol{e}(i) \in \{+1, -1\}$ is the label of the planted solution. Now we state the following two lemmas from which the theorem follows.

Lemma 3.1. *Let $G \in \mathcal{G}_n(p,r)$ be a random graph for which $(*)$ of Remark 2.1 holds with $i = 1$. Consider the execution of* `pseudo-bp-for-analysis` *on G for the first T_0 rounds, i.e., the rounds where no thresholding is applied. Then* ***whp*** *the following holds: The number of vertices i such that $|\boldsymbol{b}^{(T_0)}(i)| < \theta$ or the sign of $\boldsymbol{b}^{(T_0)}(i)$ disagrees with $\boldsymbol{e}(i)$ is $o(n)$.*

Lemma 3.2. *Let $G \in \mathcal{G}_n(p,r)$ be a random graph for which the previous lemma holds. Then at the end of the execution of* `pseudo-bp-for-analysis` *after $T_{\max}$ rounds, the following holds* ***whp****: The sign of the obtained beliefs $\boldsymbol{b}^{(T_{\max})}$ completely agree with $\boldsymbol{e}$.*

3.1 Spectral arguments

We first prove Lemma 3.1. We fix any random $G \in \mathcal{G}_n(p,r)$ for which $(*)$ of Remark 2.1 holds with $i = 1$, and let $\widehat{A}$ be the adjacency matrix of G.

First consider the belief updating formula (2). Since no thresholding is applied, we have $b_j = b'_j$ for all $j \in V$; hence (2) is restated as follows:

$$b_i \;=\; \sum_{(i,j)\in E} h_+ \cdot b_j - \sum_{(i,j)\notin E} h_- \cdot b_j, \;=\; \sum_j a_{ij} \cdot h_+ \cdot b_j - (1 - a_{ij}) \cdot h_- \cdot b_j,$$

where a_{ij} is the value of $\widehat{A}$ at the (i,j) entry. Thus, with all 1 matrix J, we can state the computation of the kth round (i.e., the computation of $\boldsymbol{b}^{(k)}$ from $\boldsymbol{b}^{(k-1)}$ as follows:

$$\boldsymbol{b}^{(k)} \;=\; h_+\widehat{A}\boldsymbol{b}^{(k-1)} + h_-(J - \widehat{A})\boldsymbol{b}^{(k-1)} \;=\; \left(h_+ - h_-)\widehat{A} + h_-J\right)\boldsymbol{b}^{(k-1)} \;=\; \left(\widehat{A} - \frac{p+r}{2}J\right)\boldsymbol{b}^{(k-1)}.$$

Thus, for any k, $1 \le k \le T_0$, the beliefs after the kth round is computed as

$$\boldsymbol{b}^{(k)} = \widehat{U}^k \boldsymbol{b}^{(0)}, \quad \text{where } \widehat{U} \overset{\text{def}}{=} \widehat{A} - \frac{p+r}{2}J,$$

and this is nothing but the standard power method for computing the eigenvector of $\widehat{U}$ with the largest eigenvalue. On the other hand, Coja-Oghlan [4] made a detail spectral analysis of $\widehat{U}$ and we borrow his analysis here (see Section 3.3).

Let $\widehat{\lambda}_1 \geq \cdots \geq \widehat{\lambda}_{2n}$ be eigenvalues of $\widehat{U}$, and $\widehat{\boldsymbol{e}}_1, \ldots, \widehat{\boldsymbol{e}}_{2n}$ be the corresponding orthonormal eigenvectors of $\widehat{U}$. Note first that (i) $\mathrm{E}[\widehat{U}] = U$, (ii) the largest eigenvalue of U is $\lambda_1 = 2(p-r)n$, and (iii) its corresponding unit eigenvector is $\boldsymbol{e}_1$, where

$$U \;=\; \frac{1}{2}\begin{pmatrix} p-r & r-p \\ r-p & p-r \end{pmatrix} \otimes J_n, \quad \text{and} \quad \boldsymbol{e}_1 \;=\; \frac{1}{\sqrt{2n}}\left(\begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes 1^n - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes 1^n \right).$$

Note that $\boldsymbol{e}_1$ is parallel to the planted label vector $\boldsymbol{e}$. Thus, roughly speaking, we can expect with high probability that $\widehat{\lambda}_1 \approx \lambda_1$ and $\widehat{\boldsymbol{e}}_1 \approx \boldsymbol{e}_1$; hence, $\boldsymbol{b}^{(k)}$ that approximates $\widehat{\boldsymbol{e}}_1$ can be used to detect the planted labels. We make this argument precise below.

Recall that the initial belief vector is $\boldsymbol{b}^{(0)} = (1, 0, \ldots, 0) \in R^{2n}$. We express it as a linear combination of orthonormal eigenvectors of $\widehat{U}$, that is,

$$\boldsymbol{b}^{(0)} \;=\; c_1\widehat{\boldsymbol{e}}_1 + \cdots + c_{2n}\widehat{\boldsymbol{e}}_{2n}, \quad \text{where } c_i \;=\; (\boldsymbol{b}^{(0)}, \widehat{\boldsymbol{e}}_i).$$

Then we have

$$\boldsymbol{b}^{(k)} \;=\; \widehat{U}^k \boldsymbol{b}^{(0)} \;=\; \widehat{\lambda}_1^k \left(c_1\widehat{\boldsymbol{e}}_1 + \left(\frac{\widehat{\lambda}_2}{\widehat{\lambda}_1}\right)^k c_2\widehat{\boldsymbol{e}}_2 + \cdots + \left(\frac{\widehat{\lambda}_{2n}}{\widehat{\lambda}_1}\right)^k c_{2n}\widehat{\boldsymbol{e}}_{2n} \right). \tag{3}$$

Now for some sufficiently small constant ϵ (which will be specified later), define $S \subset V$ be a set of vertices i such that $|\widehat{\boldsymbol{e}}_1(i) - \boldsymbol{e}_1(i)| \geq \epsilon/(8\sqrt{n})$. That is, $V \setminus S$ is the set of vertices i such that $\widehat{\boldsymbol{e}}_1(i)$ is close to $\boldsymbol{e}_1(i)$ $(= \pm 1/\sqrt{2n})$. From Corollary 3.7 we have $|S| = o(n)$ **whp**. Note that what we assumed at Remark 2.1 is that $1 \notin S$. Then noting $c_1 = (\boldsymbol{b}^{(0)}, \widehat{\boldsymbol{e}}_1) = \widehat{\boldsymbol{e}}_1(1)$, we can easily see that this assumption implies the following claim.

Claim 1. *For all $1 \leq i \leq 2n$ such that $i \notin S$, we have*

$$\begin{array}{lccccc} i \in V_L: & \dfrac{1}{(2+\epsilon)n} & \leq & c_1\widehat{\boldsymbol{e}}_1(i) & \leq & \dfrac{1}{(2-\epsilon)n} \\ i \in V_R: & -\dfrac{1}{(2-\epsilon)n} & \leq & c_1\widehat{\boldsymbol{e}}_1(i) & \leq & -\dfrac{1}{(2+\epsilon)n}. \end{array}$$

This claim estimates the first term of (3). On the other hand, as shown by the next claim, the other terms can be bounded small.

Claim 2. *For any $k = \Omega(\log n/\log d)$ (its constant depends on ϵ), the following holds **whp**: For all $j \in V$,*

$$\left| \sum_{i=2}^{2n} \left(\frac{\widehat{\lambda}_i}{\widehat{\lambda}_1}\right)^k c_i\widehat{\boldsymbol{e}}_i(j) \right| \;\leq\; \frac{\epsilon}{n}.$$

Proof. It suffices to bound $\sum_{i=2}^{2n} |\widehat{\lambda}_i/\widehat{\lambda}_1|^k$ because $|c_i| \leq 1$ due to $\sum_{i=1}^{2n} c_i^2 = 1$ and $|\widehat{\boldsymbol{e}}_i(j)| \leq 1$ due to $\|\widehat{\boldsymbol{e}}_i\| = 1$. By using the constant $c > 0$ of Corollary 3.6, **whp** we have $|\widehat{\lambda}_i/\widehat{\lambda}_1| \leq c/\sqrt{d}$ for all i, $2 \leq i \leq 2n$. Then (if these bounds hold) for any $k \geq c' \log n/\log d$ (where c' is some appropriate constant) we have $\sum_{i=2}^{2n} |\widehat{\lambda}_i/\widehat{\lambda}_1|^k \leq 2n/(\sqrt{d}/c)^k \leq \epsilon/n$. $\square$

Now fix $\epsilon = 1/4$ and let $k \geq c'' \log n/\log d$ for some c'' chosen appropriately depending on ϵ and θ. Consider the situation such that (i) $|S| = o(n)$, (ii) the bound of Claim 2, and (iii) $\widehat{\lambda}_1 \geq d/2$

(see Corollary 3.6), all hold. Then for any $i \notin S$, we have the following,

$$\begin{array}{lrcl} i \in V_L : & \boldsymbol{b}^{(k)}(i) & \geq & \widehat{\lambda}_1^k \left(\frac{1}{(2+\epsilon)n} - \frac{\epsilon}{n} \right) & \geq & \theta, \\ i \in V_R : & \boldsymbol{b}^{(k)}(i) & \leq & -\widehat{\lambda}_1^k \left(\frac{1}{(2+\epsilon)n} - \frac{\epsilon}{n} \right) & \leq & -\theta. \end{array}$$

The lemma is immediate from this observation.

3.2 Counting arguments

We prove Lemma 3.2. For our argument, the following lemma, which is similar to the one used in [1], plays an important role. (When using this lemma for the later analysis, we use parameters n and p that are used for specifying the distribution $\mathcal{G}_n(p,r)$.)

Lemma 3.3. *Let $G' = (V', E')$ be a random graph with n vertices whose undirected edges are given independently with probability p, and let $d = pn$. Then we have the following with probability $1 - n^{-\Omega(d)}$: There are no two subsets S_1 and S_2 of V' such that*

$$\begin{array}{ll} \text{(i)} & |S_1| \leq n/1000 \;\text{ and }\; |S_2| = |S_1|/2, \;\text{ and} \\ \text{(ii)} & \text{every vertex of } S_2 \text{ has at least } 0.1d \text{ neighbors in } S_1. \end{array} \tag{4}$$

Proof. The proof is similar to the one in [1], and we only state its outline.

Fix arbitrarily S_1 and S_2 such that $|S_1| \leq n/1000$ and $|S_2| = |S_1|/2$. Let $|S_2| = s$, hence we have $|S_1| = 2s$. Suppose that $|S_1 \cap S_2| = t$ where $0 \leq t \leq s$. Observe that if every vertex of S_2 has at least $0.1d$ neighbors in S_1, then we have $|(S_1 \times S_2) \cap E'| \geq 0.1d|S_2| - \alpha$ for some α, $0 \leq \alpha \leq 0.1d|S_2|/2$. Thus, the probability that S_1 and S_2 satisfy (4) is

$$\binom{2s^2 - (t+1)t/2}{0.1ds - \alpha} \left(\frac{d}{n}\right)^{0.1ds-\alpha} \leq \left(\frac{40es}{n}\right)^{0.05ds}.$$

Then by using the union bound, the probability that such S_1 and S_2 exist is bounded by $n^{-\Omega(d)}$. □

We also need the following lemma, which is obtained easily by the Chernoff bound.

Lemma 3.4. *Consider a random graph G following $\mathcal{G}_n(p,r)$. Then we have the following with probability $1 - O(n)2^{-\Omega(d')}$:*

For every $i \in V$, the number of edges to vertices in the same planted class is between $0.9d$ and $1.1d$, and to vertices in the different planted class is between $0.9d'$ and $1.1d'$. (5)

Remark 3.1. *Note that if r is too small, then the bound like $1 - O(n)2^{-\Omega(d')}$ does not make sense; thus, we need that $r \geq c \log n/n$ for sufficiently large c. But on the other hand, this lemma is only the point where we need this condition, and the lemma and the following argument can be modified easily for the case where r is not large enough.*

Now consider any G for which Lemma 3.1 holds, and the execution of the algorithm after the T_0th round. For any $k \geq 0$ (letting $k' = T_0 + k$), let $S_L^{(k)}$ (resp. $S_R^{(k)}$) be the set of vertices $i \in V_L$ (resp. $i \in V_R$) such that the value of its belief after the k'th round satisfies either $|\boldsymbol{b}^{(k')}(i)| < \theta$, or $|\boldsymbol{b}^{(k')}(i)| = \theta$ and $\mathrm{sg}(\boldsymbol{b}^{(k')}(i)) \neq \boldsymbol{e}(i)$. Then the following claim holds for $S_L^{(k)}$ and $S_R^{(k)}$.

Claim 3. *For any $k \geq 0$, suppose that $|S_L^{(k)}| = o(n)$ and $|S_R^{(k)}| = o(n)$. Then, we have the following with probability $1 - O(n)2^{-\Omega(d')}$:*

Every $i \in S_L^{(k+1)}$ has at least $0.1d$ neighbors in $S_L^{(k)}$, and the corresponding statement also holds for $S_R^{(k+1)}$. (6)

Proof. We assume that (5) holds, and this implies (6). Then the lemma follows from Lemma 3.4.

For showing (6), we argue as follows: consider any $i \in V_L$ and assume that the number of neighbors j of i such that $j \in S_L^{(k)}$ is less than $0.1d$. From which we show that $i \notin S_L^{(k+1)}$.

Estimate the value of the belief of i after the $(k'+1)$th round (where $k' = T_0 + k$). Let $b_L^{(+)}(i) \geq 0$ and $b_L^{(-)}(i) \leq 0$ (resp. $b_R^{(+)}(i) \geq 0$ and $b_R^{(-)}(i) \leq 0$) be the values of the positive and negative beliefs that the vertex i receives from vertices of V_L (resp. V_R) at the k'th round. Let $s_L = |S_L^{(k)}|$ and $s_R = |S_R^{(k)}|$. Note that any $i \in V_L \setminus S_L^{(k)}$ (resp., $i \in V_R \setminus S_R^{(k)}$) satisfies $\boldsymbol{b}^{(k')}(i) = \theta$ (resp., $\boldsymbol{b}^{(k')}(i) = -\theta$), and that any $i \in S_L^{(k)} \cup S_R^{(k)}$ satisfies either $|\boldsymbol{b}^{(k')}(i)| < \theta$, or $|\boldsymbol{b}^{(k')}(i)| = \theta$ and $\mathrm{sg}(\boldsymbol{b}^{(k')})(i) \neq \boldsymbol{e}(i)$. Then by using (5) and our assumption on i, we have the following:

$$\begin{aligned}
\left|b_L^{(+)}(i)\right| &\geq (0.9d - 0.1d)(1-(p+r)/2)\theta \\
\left|b_L^{(-)}(i)\right| &\leq 0.1d(1-(p+r)/2)|-\theta| + n((p+r)/2)\theta, \\
\left|b_R^{(+)}(i)\right| &\geq (n - s_R - 1.1d')((p+r)/2)|-\theta|, \quad \text{and} \\
\left|b_R^{(-)}(i)\right| &\leq 1.1d'(1-(p+r)/2)|-\theta| + s_R((p+r)/2)\theta.
\end{aligned}$$

Here by using $p+r \leq 3/2$, which is implied by $p/r \gg 1$, we estimate the total beliefs sent to i after the $(k'+1)$th round, thereby deriving the following bound for the belief of the next round (before the thresholding).

$$\begin{aligned}
\boldsymbol{b}^{(k'+1)}(i) &= b_L^{(+)}(i) + b_R^{(+)}(i) - b_L^{(-)}(i) - b_R^{(-)}(i) \\
&\geq (0.7d(1-(p+r)/2) - 1.1d'(1-(p+r)) - s_R(p+r))\theta \\
&\geq ((0.7d - 2.2d')(1-(p+r)/2) - o(d))\theta \quad (\because\ s_R = o(n)) \\
&\geq (0.1d - o(d))\theta \quad (\because\ p+r \leq 3/2,\ d/d' \gg 1).
\end{aligned}$$

This value is larger than θ since $d \geq \log n$; hence we can conclude that $i \notin S_L^{(k+1)}$. The same argument also holds for $i \in V_R$. □

Now we are ready to prove Lemma 3.2. Consider first the (T_0+1)th round. Since we consider a graph G for which Lemma 3.1 holds, we have $|S_L^{(0)} \cup S_R^{(0)}| = o(n)$. Below we show that the size of $S_L^{(1)}$ (resp., $S_R^{(1)}$) becomes half **whp**, and for this purpose, consider the probability $\Pr\{\,|S_L^{(1)}| \geq |S_L^{(0)}|/2\,\}$.

Let (6) denote the event that (6) of Claim 3 holds. Then we have

$$\begin{aligned}
\Pr\{\,|S_L^{(2)}| \geq |S_L^{(1)}|/2\,\} &= \Pr\{\,\neg(6) \wedge\ |S_L^{(2)}| \geq |S_L^{(1)}|/2\,\} + \Pr\{\,(6) \wedge\ |S_L^{(2)}| \geq |S_L^{(1)}|/2\,\} \\
&\leq \Pr\{\,\neg(6)\,\} + \Pr\{\,(6) \wedge\ |S_L^{(2)}| \geq |S_L^{(1)}|/2\,\}.
\end{aligned}$$

Since $|S_L^{(0)} \cup S_R^{(0)}| = o(n)$, by Claim 3, we have $\Pr\{\,\neg(6)\,\} \leq O(n)2^{-\Omega(d')}$. On the other hand, by Lemma 3.3, regarding $S_L^{(1)}$ as S_1 in the lemma and any size $|S_L^{(1)}|/2$ subset of $S_L^{(2)}$ as S_2, we have

$$\Pr\{\,(6) \wedge\ |S_L^{(2)}| \geq |S_L^{(1)}|/2\,\} = n^{-\Omega(d)}.$$

Thus,

$$\Pr\{\,|S_L^{(1)}| \geq |S_L^{(0)}|/2\,\} = O(n)2^{-\Omega(d')} + n^{-\Omega(d)} = 2^{-\Omega(d')}.$$

The same argument also holds for $S_R^{(0)}$ and $S_R^{(1)}$. Therefore, we can conclude that both $|S_L^{(1)}| < |S_L^{(0)}|/2$ and $|S_R^{(1)}| < |S_R^{(0)}|/2$ hold with probability $1 - 2^{-\Omega(d')}$.

Now by induction, we also have the same bound for $S_L^{(k)}$ and $S_R^{(k)}$ for any $k \geq 1$, and by using the union bound, the probability that this halving event occurs for enough number of rounds ($\leq \log n$ rounds) is bounded by $1 - (\log n)2^{-\Omega(d')} = 1 - o(1)$. This implies the lemma.

3.3 Bounds on eigenvalues of $\widehat{U}$

In [4], a detail spectral analysis of $\widehat{U}$ is given, and we use its results, in particular, Lemma 23 of [4]. In our context, the lemma is stated as follows.

Lemma 3.5. *For some constant $c_0 > 0$, the following properties for $\widehat{U}$ holds **whp**:*

$$\begin{array}{ll} \text{(i)} & \forall \boldsymbol{v} \in R^{2n} : \|\boldsymbol{v}\| = 1 \wedge (\boldsymbol{v}, \boldsymbol{e}_1) = 0 \ \left[\ \|\widehat{U}\boldsymbol{v}\| \le c_0\sqrt{pn}\ \right], \quad \text{and} \\ \text{(ii)} & \|(p-r)n\boldsymbol{e}_1 - \widehat{U}\boldsymbol{e}_1\| \le c_0\sqrt{pn}. \end{array}$$

By using this lemma, we derive some bounds used in our analysis.

Corollary 3.6. *For some constant $c_1 > 0$, we have **whp** that $\widehat{\lambda}_1 \ge d/2$, and $\max\{|\widehat{\lambda}_2|, |\widehat{\lambda}_{2n}|\} \le c_1\sqrt{d}$.*

Proof. We make use of Rayleigh quotient principle, or Courant-Fischer min-max theorem (see, e.g., [5]). Note first that by the symmetry of $\widehat{U}$, we have $|\boldsymbol{v}^{\mathrm{t}}\widehat{U}\boldsymbol{v}| = \|\widehat{U}\boldsymbol{v}\|$ for any unit vector $\boldsymbol{v}$. For the bound for $\widehat{\lambda}_1$, we first derive the following from the second item of Lemma 3.5.

$$(p-r)n - \|\widehat{U}\boldsymbol{e}_1\| \ \le\ c_0\sqrt{pn} \ =\ c_0\sqrt{d}.$$

Then by our assumption of $p/r \gg 1$, we have

$$\widehat{\lambda}_1 \ \ge\ |\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1| \ =\ \|\widehat{U}\boldsymbol{e}_1\| \ \ge\ (p-r)n - c_0\sqrt{d} \ \ge\ d/2.$$

An upper bound for $\widehat{\lambda}_2$ can be obtained from the first item Lemma 3.5; that is, $\widehat{\lambda}_2 \le c_0\sqrt{pn} = c_0\sqrt{d}$. For bounding $\widehat{\lambda}_{2n}$, consider any unit vector $\boldsymbol{v}$, and we express $\boldsymbol{v}$ as $\boldsymbol{v} = \alpha\boldsymbol{e}_1 + \beta\boldsymbol{w}$, where $\boldsymbol{w}$ is a unit vector perpendicular to $\boldsymbol{e}_1$. (Note that $\alpha^2 + \beta^2 = 1$ because $\boldsymbol{v}$ is a unit vector.) We first bound $\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1$. Since $\widehat{U} = \widehat{A} - ((p+r)/2)J$, we have

$$\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1 \ =\ \boldsymbol{e}_1^{\mathrm{t}}\widehat{A}\boldsymbol{e}_1 - \frac{p+r}{2}\boldsymbol{e}_1^{\mathrm{t}}J\boldsymbol{e}_1 \ =\ \boldsymbol{e}_1^{\mathrm{t}}\widehat{A}\boldsymbol{e}_1.$$

By using Chernoff bound, we can easily show that $\boldsymbol{e}_1^{\mathrm{t}}\widehat{A}\boldsymbol{e}_1$ is $(1\pm\epsilon)(p-r)n$ for any $\epsilon > 0$ **whp**. Thus, we have $\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1 > 0$, and hence the following holds **whp**:

$$\begin{array}{rcl} \boldsymbol{v}^{\mathrm{t}}\widehat{U}\boldsymbol{v} & = & \alpha^2\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1 + 2\alpha\beta\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{w} + \beta^2\boldsymbol{w}^{\mathrm{t}}\widehat{U}\boldsymbol{w} \\ & \ge & 2\alpha\beta\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{w} + \beta^2\boldsymbol{w}^{\mathrm{t}}\widehat{U}\boldsymbol{w} \quad (\because\ \boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1 > 0 \ \textbf{whp}) \\ & \ge & -2\|\widehat{U}\boldsymbol{w}\| - \|\widehat{U}\boldsymbol{w}\| \\ & \ge & -3c_0\sqrt{pn}. \end{array}$$

Therefore, we have $\widehat{\lambda}_{2n} \ge -3c_0\sqrt{d}$. □

Corollary 3.7. *For any $\epsilon > 0$, we have **whp** that the number of vertices i such that $|\widehat{\boldsymbol{e}}_1(i) - \boldsymbol{e}_1(i)| \ge \epsilon/\sqrt{n}$ is $o(n)$.*

Proof. We express $\widehat{\boldsymbol{e}}_1$, the unit eigenvector corresponding to the largest eigenvalues of $\widehat{U}$, as $\widehat{\boldsymbol{e}}_1 = \alpha\boldsymbol{e}_1 + \beta\boldsymbol{w}$, where $\boldsymbol{w}$ is a unit vector perpendicular to $\boldsymbol{e}_1$. (Note that $\alpha^2 + \beta^2 = 1$ because $\boldsymbol{v}$ is a unit vector.) We may assume that $\alpha \ge 0$. The second item of Lemma 3.5 is equivalent to

$$(p-r)n - \|\widehat{U}\boldsymbol{e}_1\| \ \le\ c_0\sqrt{pn}, \quad \text{and} \tag{7}$$

$$\|\widehat{U}\boldsymbol{e}_1\| - (p-r)n \ \le\ c_0\sqrt{pn}. \tag{8}$$

Thus, we have the following **whp**:

$$\begin{array}{rcl}(p-r)n - c_0\sqrt{d} & \le & \|\widehat{U}\boldsymbol{e}_1\| \quad (\because\ (7)) \\ & = & \|\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1\| \ \le\ \|\widehat{\boldsymbol{e}}_1^{\mathrm{t}}\widehat{U}\widehat{\boldsymbol{e}}_1\| \\ & = & \alpha^2\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1 + 2\alpha\beta\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{w} + \beta^2\boldsymbol{w}^{\mathrm{t}}\widehat{U}\boldsymbol{w} \\ & \le & \alpha^2|\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1| + 2|\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{w}| + |\boldsymbol{w}^{\mathrm{t}}\widehat{U}\boldsymbol{w}| \\ & \le & \alpha^2|\boldsymbol{e}_1^{\mathrm{t}}\widehat{U}\boldsymbol{e}_1| + 3\|\widehat{U}\boldsymbol{w}\| \\ & \le & \alpha^2\|\widehat{U}\boldsymbol{e}_1\| + 3c_0\sqrt{d} \quad (\because \text{ the first item of Lemma 3.5}) \\ & \le & \alpha^2(p-r)n + 4c_0\sqrt{d}. \quad (\because\ (8))\end{array}$$

The above inequality implies $1-\alpha^2 = O(\sqrt{d}/d)$; hence $1-\alpha = O(\sqrt{d}/d)$ and $|\beta| = O(d^{1/4}/d^{1/2})$. Let ℓ be the number of vertices i such that $|\widehat{\boldsymbol{e}}_1(i) - \boldsymbol{e}_1(i)| \ge \epsilon/\sqrt{n}$. Then we have

$$\sqrt{\frac{\ell}{n/\epsilon^2}} \ \le\ \|\widehat{\boldsymbol{e}}_1 - \boldsymbol{e}_1\| \ =\ \|\beta\boldsymbol{w} - (1-\alpha)\boldsymbol{e}_1\| \ \le\ (1-\alpha)+\beta \ =\ O\left(\sqrt{\sqrt{d}/d}\right).$$

Therefore, we can conclude that for any $\epsilon > 0$, we have $\ell/n \le \sqrt{d}/(\epsilon^2 d)$, that is, $\ell = o(n)$ **whp**. $\square$

4 Concluding Remarks

We analyzed an algorithm for the Most Likely Partitioning problem based on the belief propagation. Since the algorithm uses the information on the parameters p and r, some consideration is necessary for using it to the Bisection problem (under the planted solution model). Note here that, besides the two threshold parameters, what we need for the algorithm is the value $p+r$, and that, as our analysis indicates, the algorithm works so long as thresholds are large enough. On the other hand, under the assumption that the graph is generated from some solution of the Bisection problem, $p+r$ can be estimated with reasonable accuracy from the total number of edges of a given graph G. Thus, we can use our algorithm for the Bisection problem as well. More generally, the robustness to the probability parameter estimation is left open.

Acknowledgments

The relation between our algorithm based on the belief propagation (i.e., `pseudo-bp`) and a spectral method has been pointed out first by Professor Amin Coja-Oghlan to the second author. Without this advice this research would not have existed, and the authors would like to express deep appreciation to his advice.

References

[1] N. Alon and N. Kahale, A spectral technique for coloring random 3-colorable graphs, *SIAM J. Comput.* 26(6), 1733–1748, 1997.

[2] R.B. Boppana, Eigenvalues and graph bisection: an average-case analysis, in *Proc. Symposium on Foundations of Computer Science*, 280–285, 1987.

[3] A. Braunstein, M. Mezard, and R. Zecchina, Survey propagation: an algorithm for satisfiability, *Random Struct. and Algorithms* 27(2), 201–226, 2005.

[4] Amin Coja-Oghlan, A spectral heuristic for bisecting random graphs, *Random Struct. and Algorithms* 29(3), 351–398, 2006.

[5] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, 1997.

[6] U. Feige, E. Mossel, and D. Vilenchik, Complete convergence of message passing algorithms for some satisfiability problems, in *Proc. RANDOM and APPROX 2006.*

[7] M. Jerrum and G. Sorkin, The Metropolis algorithm for graph bisection, *Discrete Appl. Math* 82(1-3), 155–175, 1998.

[8] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, Improved low-density parity-check codes using irregular graphs, *IEEE Trans. on Information Theory*, 47(2), 585–598, 2001.

[9] D. MacKay, Good error-correcting codes based on very sparse matrices, *IEEE Trans. Inform. Theory*, **IT-45**(2), 399–431, 1999.

[10] R. McEliece, D. MacKay, and J. Cheng, Turbo decoding as an instance of Pearl's "Belief Propagation" algorithm, in *IEEE J. on Selected Areas in Comm.* 16(2), 1998.

[11] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers Inc., 1988.

[12] M. Onsjö and O. Watanabe, A simple message passing algorithm for graph partition problem, in *Proc. 17th Int'l Sympos. on Algorithms and Computation* (ISAAC'06), LNCS 4288, 507–516, 2006. (A longer version has been submitted for the publication.)

[13] M. Onsjö and O. Watanabe, Finding most likely solutions, submitted.

Appendix: Derivation of Algorithm `pseudo-bp`

The algorithm stated in Figure 1 is obtained from the standard belief propagation algorithm for the MLP problem. Although this derivation has been stated in [12] (journal submission version), we state it (and its slight modification) here for the sake of completeness. Here we show its derivation and explain the points that differ from the belief propagation.

Let $G = (V, E)$ be an input graph with $2n$ vertices; let $V = \{1, ..., 2n\}$. Our task is to compute, for given probability parameters p and r, a partition maximizing $\Pr[G|V_+, V_-]$ defined by (1). For this, we use the belief propagation computing the following marginal probabilities $P(i)$ for each $i \in V$.

$$P(i) \;=\; \Pr[\, i \in V_+ \,|\, G \,] \;=\; \sum_{\substack{(V_+, V_-)\\ \text{s.t. } i \in V_+}} \Pr[\,(V_+, V_-)\,|\, G\,]$$

Then we simply assign $a_i = +1$ (i.e., $i \in V_+$) if and only if $P(i) > 0.5$.

We first explain this algorithm. Below we follow [10] for notions and notations on the belief propagation. (Although we will not explain the precise meaning of such notations, it is not essential for our derivation.) For any $i, j \in V$, we let $e_{ij} = +1$ if there exists an edge between i and j in E, and $e_{ij} = -1$ otherwise. A Bayesian network for G is a graph consisting of nodes $\{N_i\}_{1 \le i \le 2n}$ corresponding to vertices in V nodes $\{Z_{ij}\}_{1 \le i < j \le 2n}$ corresponding to all unordered pairs in $V \times V$. The belief propagation updates beliefs on these nodes by exchanging messages between them. But since those messages are quite simple in our case, we can simplify this scheme so that messages are exchanged between nodes corresponding to vertices in V. For each pair of vertices $i, j \in V$, where $i \ne j$, two messages $\pi_{ij}(-1)$ and $\pi_{ij}(+1)$ sent from node N_i to N_j are computed as follows from messages π_{ki} that node N_i received at the previous round. (In the following, the domain of the subscript k (sometimes j) of $\prod$ or $\sum$ is $\{1, ..., 2n\} - \{i\}$.)

$$\pi_{ij}(x) \;=\; \alpha q_i(x) \prod_{k:k \ne j} (\delta_{ij}(r)\pi_{ki}(-x) + \delta_{ij}(p)\pi_{ki}(x)), \tag{9}$$

Here q_i, α, and δ_{ij} have the following meaning: $q_i(x)$ is a priori probability of $a_i = x$ (in our case, $q_i(+1) = q_i(-1) = 1/2$ except for the vertex 1); α is a normalization factor to keep $\pi_{ij}(+1) + \pi_{ij}(-1) = 1$; and $\delta_{ij}(y) = y$ if $e_{ij} = +1$, and $\delta_{ij}(y) = 1 - y$ otherwise. Notice here that for computing a message π_{ij} from node N_i to node N_j, the previous value of π_{ji}, i.e., a message from node N_j, is not used. This is the point we will relax later in our modification. A belief Bel_i at node N_i, intuitively the belief for $a_i = +1$, is then computed as follows:

$$Bel_i \;=\; \frac{\prod_j \pi_{ji}(+1)}{\prod_j \pi_{ji}(+1) + \prod_j \pi_{ji}(-1)}.$$

Now we make several simplifications for our problem. First in order to reduce the number of variables, we use $m_{ij} = \pi_{ij}(+1)/\pi_{ij}(-1)$ and $B_i = \prod_j m_{ji}$; also let $\rho_i = q_i(+1)/q_i(-1)$. Note that we can now consider $a_i = +1$ if $B_i > 1$ and $a_i = -1$ if $B_i < 1$. The following updating rule is obtained from (9).

$$m_{ij} \;=\; \rho_i \prod_{k:k \ne j} f_{e_{ij}}(m_{ki}),$$

where $f_{e_{ij}}(x)$ is defined by

$$f_{e_{ij}}(x) \;=\; \frac{c_{ij}x + 1}{x + c_{ij}}, \qquad c_{ij} \;=\; \begin{cases} c_+ = \dfrac{p}{r}, & \text{if } e_{ij} = +1, \text{ and} \\ c_- = \dfrac{1-p}{1-r}, & \text{if } e_{ij} = -1. \end{cases} \tag{10}$$

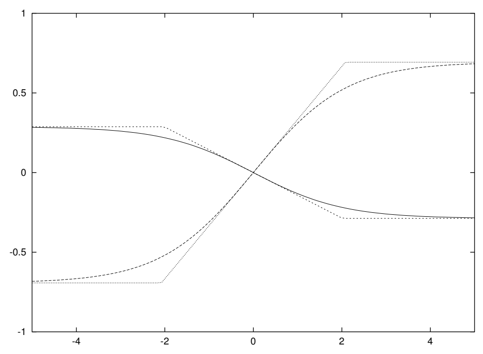

Figure 2: lex_+ and lex_- and their approximations $\widetilde{\text{lex}}_+$ and $\widetilde{\text{lex}}_-$ (for $p = 0.4$ and $r = 0.2$)

At this point, we introduce one a priori knowledge. Without losing generality, we may fix the classification of vertex 1 and assume that $1 \in V_+$ and $a_1 = +1$. This means that $q_1(+1) = 1$ and $q_1(-1) = 0$, implying that $\rho_1 = +\infty$ and $m_{1j} = +\infty$. For the other i's, we have $q_i(+1) = q_i(-1) = 0.5$, and hence, $\rho_i = 1$. Thus, we have the following simplified rule.

$$m_{1j} \; = \; +\infty, \quad \text{and} \quad m_{ij} \; = \; \prod_{k:k\neq j} f_{e_{ij}}(m_{ki}).$$

Here we may define $f_{e_{ij}}(+\infty) = c_{ij}$.

Let us convert this updating rule to additive one. For this purpose, we introduce $\ell_{ij} = \ln(m_{ij})$ and a function lex defined by

$$\text{lex}_{e_{ij}}(x) = \ln(f_{e_{ij}}(e^x)). \tag{11}$$

Then, for all $i, j \in V$, where $i \neq 1$ and $i \neq j$, we have

$$\ell_{ij} \; = \; \sum_{k:k\neq j} \text{lex}_{e_{ki}}(\ell_{ki}), \tag{12}$$

Note that $\ell_{1j} = +\infty$. The logarithmic belief $\ln(B_i)$ is computed as $\sum_{j\in V} \ell_{ij}$, and a_i is determined whether it is positive or negative.

Up to this point, our modification does not change the essential meaning of the belief propagation. Now we introduce two approximations for simplifying the updating rule. As shown in Figure 2, both functions lex_+ and lex_- can be approximated well by some linear functions with thresholds. More specifically, we consider the following functions for approximating lex_σ, $\sigma \in \{+, -\}$.

$$\widetilde{\text{lex}}_\sigma(x) \; = \; \begin{cases} h'_\sigma \cdot \theta'_\sigma, & \text{if } \theta'_\sigma < x, \\ \sigma h'_\sigma \cdot x, & \text{if } -\theta'_\sigma \le x \le \theta'_\sigma, \text{ and} \\ -h'_\sigma \cdot \theta'_\sigma, & \text{if } x < -\theta'_\sigma, \end{cases} \tag{13}$$

where h'_σ (i.e., h'_+ and h'_-) and θ'_σ (i.e., θ'_+ and θ'_-) are determined as follows, by some simple calculation from (10), (11), and (12). (Here we assume that $r < p < 0.5$.)

$$h'_+ \; = \; \left|\frac{c_+ - 1}{c_+ + 1}\right| \; = \; \frac{p-r}{p+r}, \;\; h'_- \; = \; \left|\frac{c_- - 1}{c_- + 1}\right| \; = \; \frac{p-r}{2-(p+r)}, \;\; \theta'_- \; = \; \frac{-\ln c_-}{h'_-}, \;\; \theta'_+ \; = \; \frac{\ln c_+}{h'_+}.$$

A (linearized version of) belief propagation algorithm is to compute messages by (12) by using $\widetilde{\text{lex}}_\pm$ for $\text{lex}_\pm$. (Yet simpler parameters h_+, h_-, θ_+, and θ_- of Figure 1 are obtained by multiplying $(2-(p+r))(p+r)/(2(p-r))$ to all the above parameters.)

Finally we introduce the second approximation. When computing a message m_{ij} from node N_i to node N_j, the previous value of m_{ji}, a message that N_i received from N_j, is excluded. But our preliminary experiments show that the behavior of the algorithm becomes more stable if m_{ij} is computed by using all previous messages coming to N_i. Thus, we modify the algorithm so that ℓ_{ij} is computed by using ℓ_{ki} for all k. Then there is no distinction between messages to N_j and to $N_{j'}$, and we only need to consider the following quantity:

$$b_i \;=\; \sum_j \widetilde{\mathrm{lex}}_{e_{ij}}(b_j), \tag{14}$$

which we may interpret as a message from vertex i to any other vertex in V. Furthermore, we may now consider it also as a quantity corresponding to $\ln(B_i)$, which we will call a *pseudo belief*. It is easy to see that our base algorithm `pseudo-bp` computes this pseudo belief by using the updating formula (14).