Percolation and matrix spectrum through NIB message passing

Pedro Hack[1,2]

[1] German Aerospace Center, Institute of Communication and Navigation, Müncherstraße 20, 82234 Weßling, Germany
[2] Technical University of Munich, School of CIT, Department of Computer Science, Boltzmannstraße 3, 85748 Garching, Germany
pedro.hack@dlr.de

Abstract. Given its computational efficiency and versatility, belief propagation is the most prominent message passing method in several applications. In order to diminish the damaging effect of loops on its accuracy, the first explicit version of generalized belief propagation for networks, the KCN-method, was recently introduced. This approach was developed in the context of two problems: percolation and the calculation of matrix spectra. The KCN-method was then extended in order to deal with graphical models' inference on networks. It was in this scenario where an improvement on the KCN-method, the NIB-method, was conceived. We show here that this improvement can also be achieved in the original applications of the KCN-method.

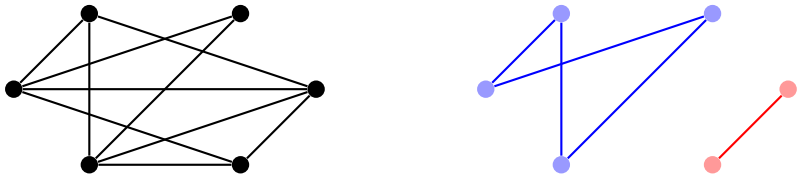
Keywords: percolation, matrix spectrum, belief propagation

1 Introduction

Message passing schemes have been shown to be key in order to address problems in several areas which are based on graphs and hypergraphs. This includes statistical mechanics, general constraint satisfaction problems, disease spread and even quantum error correction [13, 9, 7].

Given its low time complexity, the most widely used message passing algorithm is **belief propagation** (BP). Despite its advantages, BP is known to suffer from accuracy losses when dealing with short loops. As a result, **generalized belief propagation** (GBP) [17] was introduced. While providing a basis for improving on BP, the main issue with generalized belief propagation was its lack of concreteness. In fact, the first explicit and general instance of GBP appeared only around two decades after the introduction of GBP [2]. We refer to this instance as the **KCN-method** [5, 4].

The KCN-method was originally developed in order to target two specific problems: **percolation** [15, 12], and the computation of the **spectra** of sparse symmetric matrices. The method was then extended to inference problems in the context of probabilistic graphical models on networks and statistical mechanics



(a) (b)

Fig. 1: A (connected) base graph (a) and some realized graph associated to it (b). The realized graph consists of two connected components which are colored in red and blue, respectively.

[6], and it has since then found applications in a wide variety of contexts [1, 16, 3, 5]. The interested reader may find the overview in [11] very useful.

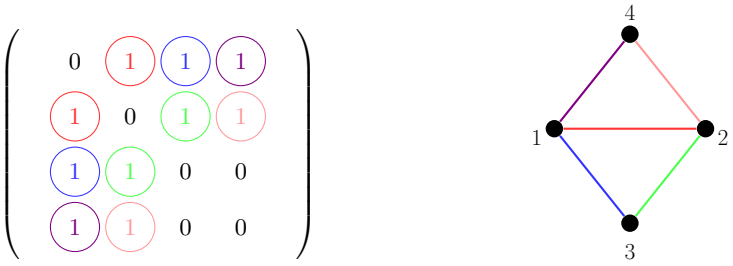
Recently, an improvement on the KCN-method, the so-called **NIB-method** [4], was introduced in the context of network inference. Since no further applications of the NIB-method have been considered, our purpose here is to show that the NIB-method also provides an improvement on the KCN-method in the context of the original target problems for which the KCN-method was developed.

2 Target problem I: Percolation

We assume here we are given some **base graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ which consists of a set of nodes $\mathcal{V}$ and a set of potential connections $\mathcal{E}$ between pairs of nodes, with $(i, j) \in \mathcal{E}$ representing a potential connections between $i \in \mathcal{V}$ and $j \in \mathcal{V}$. While we also assume the base graph to be connected, some of its connections may not be actually available. In fact, we flip a coin with some bias p independently for each edge $(i, j) \in \mathcal{E}$ such that (i, j) becomes **available** or **occupied** with probability p [2]. That is, the **realized** or **occupied graph** $\mathcal{G}'$ is a subgraph of the base graph, $\mathcal{G}' \subseteq \mathcal{G}$. We include a base graph and some realized graph in Figure 1. The process we described is known as **bond** percolation and should not be confused with **site** percolation.

Our aim is to understand the distribution of the sizes of the connected subgraphs or **clusters** and to determine the existence of a **percolating cluster**, that is, a cluster occupying a non-vanishing network fraction in the limit of large size. That is, our aim is to understand the distribution of realized graphs $\mathcal{G}'$ for some fixed base graph $\mathcal{G}$.

The main quantity we intend to compute is the probability that node i belongs to a non-giant cluster of size s, $\pi_i(s)$. Given $\pi_i(s)$, we can compute the probability that node i belongs to a small cluster (of any size), and also the expected fraction of the network that belongs to the percolation cluster. Lastly, we can compute the expected size of the clusters that $i \in \mathcal{V}$ belongs to, $\langle s_i \rangle \equiv \sum_s s\pi_i(s)$. We explain how to deal with percolation through the NIB-method in Section 5.



(a) (b)

Fig. 2: A symmetric matrix (a) and its associated weighted graph (b). We color the entries of the matrix in the same color as the edges in the associated graph. Since the weights are all equal in this example, we do not include them in the figure.

3 Target problem II: Matrix spectra

We assume here we are given some $n \times n$ symmetric matrix $\mathbf{A}$. Our aim is to compute the **spectrum** of $\mathbf{A}$, that is, its set of eigenvalues. In order to do so, we can approximate its **spectral density** $\rho(x) \equiv \frac{1}{n}\sum_{i=1}^{n}\delta(x-\lambda_i)$, where $\{\lambda_i\}_i$ are the eigenvalues of $\mathbf{A}$, and $\delta(\cdot)$ is Dirac's delta.

By Using [10, Eq. 21], and taking and $z \equiv x + i\eta$, one can show [2] that the spectral density is approximately the imaginary part of the **complex spectral density** $\rho(z) \equiv -\frac{1}{n\pi}\sum_{i=1}^{n}\frac{1}{z-\lambda_i} = -\frac{1}{n\pi z}\sum_{s=0}^{\infty}\sum_{i=1}^{n}\frac{X_i^s}{z^s}$, where $X_i^s \equiv [\mathbf{A}^s]_{ii}$ is the ith diagonal element of $\mathbf{A}^s$. In order to accurately approximate the spectral density, one ought to take the limit as the imaginary part $\eta \to 0$ from above.

We can associate to every $n \times n$ symmetric matrix $\mathbf{A}$ a **weighted graph** $\mathcal{G}_{\mathbf{A}} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where we associate one vertex to each index that a column or row of $\mathbf{A}$ may take, $\mathcal{V} \equiv \{1, \ldots, n\}$; given $i, j \in \mathcal{V}$ with $i \leq j$, then $(i, j) \in \mathcal{E}$ if and only if $[\mathbf{A}]_{ij} \neq 0$ [3]; if $(i, j) \in \mathcal{E}$, then $w_{(i,j)} = [\mathbf{A}]_{ij}$. We include an example of the weighted graph associated to some symmetric matrix in Figure 2.

A **closed walk** that starts and ends at $i \in \mathcal{V}$, or i**-walk** to be precise, is a sequence of vertices $(i_0, \ldots, i_m)$ such that $i_0 = i_m = i$ and $(i_t, i_{t+1}) \in \mathcal{E}$ for $0 \leq t \leq m-1$. Such a walk has an associated **weight**, which is the product of the matrix elements on the edges traversed by the walk. Closed walks and their weights are important in order to study matrix spectra since X_i^s equals the sum of the weights of all the length s closed walks on $\mathcal{G}_{\mathbf{A}}$ that start and end at $i \in \mathcal{V}$.

An **excursion**, or an i**-excursion** to be more precise, is a closed walk that starts in $i \in \mathcal{V}$ and only returns once to i (i.e. at the **end** of the walk). This means that any closed walk c returning m times to i can be decomposed as a succession

[3] We could avoid this condition and simply take $\mathcal{G}_A$ to have full connectivity with some weights being null. However, since our message passing methods will be exploiting the sparsity of $\mathbf{A}$, it is more convenient to associate a sparse graph $\mathcal{G}_{\mathbf{A}}$ to a sparse matrix $\mathbf{A}$.

of m excursions $(w_i)_{i=1}^m$, with the length of c being s provided $s = \sum_{u=1}^m s_u$, where s_i is the length of w_i for $i = 1, \ldots, m$.

Excursions are key in our study of matrix spectra. This is the case since, if we denote by Y_i^s the sum of the weights of all excursions of length s that start and end at node i, then, given that

$$X_i^s = \sum_{m=0}^{\infty} \left[\sum_{s_1=1}^{\infty} \cdots \sum_{s_m=1}^{\infty} \delta\big(s, \textstyle\sum_{u=1}^m s_u\big) \displaystyle\prod_{u=1}^m Y_i^{s_u} \right], \tag{1}$$

we get that the complex spectral density becomes $\rho(z) = -\frac{1}{n\pi} \sum_{i=1}^n \frac{1}{z - H_i(z)}$, where $H_i(z) \equiv \sum_{s=1}^{\infty} \frac{Y_i^s}{z^{s-1}}$. Thus, we can compute $\rho(z)$ and determine the spectrum of $\mathbf{A}$ through $H_i(z)$. We explain how to compute $H_i(z)$ through the NIB-method in Section 6.

4 The NIB-method

The NIB-method [4] was introduced in the context of (network) graphical models. One starts with a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that is associated with a probability distribution μ, that is, where the nodes $i \in \mathcal{V}$ represent random variables X_i over which μ is defined and the edges $\mathcal{E}$ represent a factorization of μ in terms of pairs of random variables $\{X_i, X_j\}$, $\mu(x_1, \ldots, x_{|\mathcal{V}|}) \propto \prod_{(i,j)\in\mathcal{G}} f_{i,j}(x_i, x_j)$, where $f_{i,j} : X_i \times X_j \to \mathbb{R}_{\geq 0}$ and we omit a normalization constant.

Formally, the NIB-method uses an underlying integer parameter $r \geq 0$, the **loop bound**, and defines a message-passing scheme for each value of r. In order to do so, it considers two types of neighborhoods: the **primary** neighborhoods $\{N_i^{(r)}\}_{i\in\mathcal{V}}$, where $N_i^{(r)}$ consists of i together with its nearest neighbors $\mathcal{NN}_i \equiv \{j \in \mathcal{V} | (i,j) \in \mathcal{V}\}$ and the edges joining it to them, plus both edges and nodes along paths of length at most r that join two nearest neighbors of i; and the **intersection** neighborhoods $\{N_{i\cap j}^{(r)}\}_{i\in\mathcal{V}, j\in N_i^{(r)}\setminus\{i\}}$, where $N_{i\cap j}^{(r)} \equiv N_i^{(r)} \cap N_j^{(r)}$ and $\cap$ is the usual set intersection.

Another important set of neighborhoods, which are not used in the NIB-method but appear in the KCN-method, are the **difference** neighborhoods $\{N_{i\setminus j}^{(r)}\}_{i\in\mathcal{V}, j\in N_i^{(r)}\setminus\{i\}}$, where $N_{i\setminus j}^{(r)}$ consists of node i together with all the edges in $N_i^{(r)}$ which are not in $N_j^{(r)}$, and the nodes at the endpoints of such edges. In the following, we drop the superscript (r) for commodity.

While they share the same fundamental subgraph partitioning philosophy, the NIB and KCN methods use different subgraphs: the KCN-method uses both primary and difference neighborhoods, while the NIB-method uses only intersection neighborhoods. The difference between the used neighborhoods results in the NIB-method having a smaller time complexity. Moreover, it can be shown that such an advantage does not come with an accuracy loss for those graphs where the KCN-method is known to give good results [4].

Depending on whether the loop bound r is **fulfilled**, i.e. all loops around i are contained within $N_i^{(r)}$ for each $i \in \mathcal{V}$, we distinguish two instances of the

NIB-method: r**-bounded loops** (if it is fulfilled), and r**-unbounded loops** (if it is not). As remarked in [4], both cases can be introduced together, although distinguishing between them is useful from a pedagogical point of view.

4.1 r-bounded loops

If the loop bound is **fulfilled**, then one can associate to $\mathcal{G}$ a **hypernetwork** $\mathcal{G}/\sim$ that is loopless. We can do so because of the **equivalence class condition**, which states that, for $i \in \mathcal{V}$ and $j \in \mathcal{V} \cap (N_i \setminus \{i\})$, we have $N_{i\cap j} = N_{k\cap q}$ for all $k, q \in \mathcal{V} \cap N_{i\cap j}$, $k \neq q$. Taking this into account, we introduce an equivalence relation $\sim$ on the intersection neighborhoods, $N_{i\cap j} \sim N_{k\cap q}$ if and only if $N_{i\cap j} = N_{k\cap q}$, and end up with a hypernetwork $\mathcal{G}/\sim \equiv \left(\cap/\sim, \{e_\alpha\}_{\alpha\in \mathrm{Piv}(\mathcal{G})}\right)$ that consists of the equivalence classes as nodes $\cap/\sim \equiv \left\{\overline{i\cap j}\right\}_{i\in\mathcal{V}, j\in N_i\setminus\{i\}}$, with $\overline{i\cap j}$ being the equivalence class of $N_{i\cap j}$, and one hyperedge $e_\alpha \equiv \{\overline{i\cap j} \in \cap/\sim | \alpha \in \overline{i\cap j}\}$ for each node in the **pivots set** $\alpha \in \mathrm{Piv}(\mathcal{G})$, that is, for each $\alpha \in \mathcal{V}$ that belongs to at least two different equivalence classes in $\cap/\sim$.

Since the loop bound is fulfilled, $\mathcal{G}/\sim$ is loopless and the NIB-method provides exact results by exchanging messages between the equivalence classes through the hyperedges $\{m^{(t)}_{\overline{i\cap j}\to i}\}_{i\in V, j\in(N_i\setminus\{i\})/\sim, t\geq 0}$ where, given $j, k \in N_i \setminus \{i\}$, $j \sim k$ if and only if $\overline{i\cap j} = \overline{i\cap k}$. The exact form of the messages in the context of (network) graphical model inference [4] is not relevant for our purposes here.

4.2 r-unbounded loops

If the loop bound is **not** fulfilled, then one cannot associate to $\mathcal{G}$ a loopless hypernetwork anymore. Moreover, the neighborhood intersections do not constitute equivalence classes and may overlap in non-trivial ways. Thus, if we still want to use the neighborhoods intersections in this context and in order to avoid unnecessary errors, we ought to find ways of coping with overcounting during the message update and inference phases. In order to do so, and taking $2^{\mathcal{E}}$ to be the power set of $\mathcal{E}$, [4] introduces two maps:

- $\overline{\mathcal{P}_{i\cap j}} : \{N_{k\cap q}\}_{k\in N_{i\cap j}, q\in N_k} \to 2^{\mathcal{E}}$. To define $\overline{\mathcal{P}_{i\cap j}}(\cdot)$, we first recursively define the set $\mathcal{P}_{i\cap j}$ as follows: we initialize it by including all the functions within $N_{i\cap j}$; at each following step, we pick some $k \in N_{i\cap j}$ and some $q \in N_k$ and we incorporate the functions within $N_{k\cap q}$ to $\mathcal{P}_{i\cap j}$. Lastly, we take $\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})$ to be $\mathcal{P}_{i\cap j}$ at the step right before the pair $k, q \in \mathcal{V}$ is selected.
- $\overline{\mathcal{Q}_i} : \{N_{i\cap j}\}_{j\in N_i\setminus\{i\}} \to 2^{\mathcal{E}}$. To define $\overline{\mathcal{Q}_i}(\cdot)$, we first recursively define the set $\mathcal{Q}_i$ as follows: we initialize it as the empty set; at each following step, we pick some $j \in N_i \setminus \{i\}$ and we incorporate the functions within $N_{i\cap j}$ to $\mathcal{Q}_i$. Lastly, we take $\overline{\mathcal{Q}_i}(N_{i\cap j})$ to be $\mathcal{Q}_i$ at the step right before $j \in \mathcal{V}$ is selected.

The messages $\{m^{(t)}_{k\cap q\to i\cap j}\}_{i\in V, j\in N_i\setminus\{i\}, k\in N_{i\cap j}, q\in N_k, t\geq 0}$ some intersection $N_{i\cap j}$ receives in this case are sent from all the intersections $N_{k\cap q}$, such that $k \in N_{i\cap j}$ and $q \in N_k$. The exact form of the messages in the context of (network) graphical model inference [4] is again not relevant for our purposes here.

5 Percolation via the NIB-method

5.1 r-bounded loops

Since we are randomly occupying the available edges, we can associate to each node $i \in \mathcal{V}$ a random variable $\Gamma_i \subseteq \mathcal{V} \cap N_i$ which consists of the set of variables in N_i which are reachable from i traversing only occupied edges in some specific configuration. In order to compute $\pi_i(s)$, we first compute $\pi_i(s|\Gamma_i)$, the probability that $i \in \mathcal{V}$ belongs to a cluster of size s given some configuration of occupied edges in N_i, Γ_i.

Given some Γ_i and some $j \in \Gamma_i$, and taking as $s_{\overline{j \cap k}}$ the size of the cluster that node j would belong to provided we remove from $\mathcal{G}$ all the equivalence classes that j belongs to except for $\overline{j \cap k}$, we get that

$$\pi_i(s|\Gamma_i) = \sum_{\{s_{\overline{j\cap k}}: j\in\Gamma_i\setminus\{i\}, k\in(N_j\setminus\{i,j\})/\sim\}} \left[\prod_{j\in\Gamma_i\setminus\{i\}} \prod_{k\in(N_j\setminus\{i,j\})/\sim} \pi_{\overline{j\cap k}\to j}(s_{\overline{j\cap k}}) \right] \times \delta(s-1, \textstyle\sum_{j\in\Gamma_i\setminus\{i\}, k\in(N_j\setminus\{i,j\})/\sim} s_{\overline{j\cap k}}), \tag{2}$$

where $\pi_{\overline{j\cap k}\to j}(s)$ is the probability that node j is in a cluster of size s once the edges in the equivalence classes $\overline{j \cap q} \neq \overline{j \cap k}$ have been removed, and $\delta(\cdot,\cdot)$ is the Kronecker delta.

For our purposes, it is useful [12, 2] to define a generating function for $\pi_i(s|\Gamma_i)$: $H_i(z|\Gamma_i) \equiv \sum_s \pi_i(s|\Gamma_i)\, z^s$. In fact, in our setup, we have that

$$H_i(z|\Gamma_i) = z \prod_{j\in(N_i\setminus\{i\})/\sim} \prod_{k\in\overline{i\cap j}\setminus\{i\}} \left[H_{\neg(\overline{i\cap j})\to k}(z) \right]^{w_{ik}^{\overline{i\cap j}}}, \tag{3}$$

where we have used (2), we have introduced both $\Gamma_{i\cap j} \equiv \Gamma_i \cap (\overline{i \cap j})$ and the random variable $w_{ik}^{\overline{i\cap j}}$, $w_{ik}^{\overline{i\cap j}} \equiv 1$ if there is a path of occupied edges within $\overline{i \cap j}$ from i to k and $w_{ik}^{\overline{i\cap j}} \equiv 0$ otherwise, and, given some $k \in \overline{i \cap j}$, we use the scalar $H_{\neg(\overline{i\cap j})\to k}(z) \equiv \prod_{q\in(N_k\setminus\{i,j\})/\sim} H_{\overline{k\cap q}\to k}(z)$.

In order to compute $\pi_i(s)$, we ought to average $\pi_i(s|\Gamma_i)$ over the possible configurations Γ_i, that is, we have that $\pi_i(s) = \left\langle \pi_i(s|\Gamma_i) \right\rangle_{\Gamma_i}$, where the average is weighted via the probability of each realization Γ_i: $p^k(1-p)^{m-k}$, where $m \equiv |N_i \cap \mathcal{E}|$ is the number of edges in N_i, and k is the number of occupied edges in N_i.

Averaging over Γ_i in (3) and taking $H_i(z) \equiv \sum_s \pi_i(s)\, z^s = \left\langle H_i(z|\Gamma_i) \right\rangle_{\Gamma_i}$ we obtain $H_i(z) = zG_i(\mathbf{H}_{\to i}(z))$, where we denote by $G_{\overline{i\cap j}\to i}(\mathbf{y})$ a generating function for $w_{ik}^{\overline{i\cap j}}$,

$$G_{\overline{i\cap j}\to i}(\mathbf{y}) \equiv \Big\langle \prod_{k\in\overline{i\cap j}\setminus\{i\}} y_k^{w_{ik}^{\overline{i\cap j}}} \Big\rangle_{\Gamma_{i\cap j}} \tag{4}$$

and we define $G_i(\mathbf{y}) \equiv \prod_{j\in(N_i\setminus\{i\})/\sim} G_{\overline{i\cap j}\to i}(y_j)$ and the vector $\mathbf{H}_{\overline{i\cap j}\to i}(z) \equiv \left(H_{\neg(\overline{i\cap j})\to k}(z) \right)_{k\in\overline{i\cap j}\setminus\{i\}}$. Lastly, we take $\mathbf{H}_{\to i}(z) \equiv \left(\mathbf{H}_{\overline{i\cap j}\to i}(z) \right)_{j\in(N_i\setminus\{i\})/\sim}$.

To conclude our calculation, we ought to evaluate the $H_{\overline{k\cap q}\to k}(z)$. This can be done following the idea in the computation of $H_i(z)$, the only difference being that we only consider the product over the elements in the equivalence class $\overline{k\cap q}$. That is, we can derive a generating function

$$H_{\overline{k\cap q}\to k}(z|\Gamma_{k\cap q}) = z\prod_{s\in\overline{k\cap q}\setminus\{k\}}\Big[H_{\neg(\overline{k\cap q})\to s}(z)\Big]^{w_{ks}^{\overline{k\cap q}}} \tag{5}$$

that, once averaged over $\Gamma_{k\cap q}$, yields

$$H_{\overline{k\cap q}\to k}(z) = zG_{\overline{k\cap q}\to k}\big(\mathbf{H}_{\overline{k\cap q}\to k}(z)\big). \tag{6}$$

We can solve (6) iteratively by message passing, starting with some initial random values and iterating the equations to convergence.

From the cluster size generating function we can derive other quantities of interest: The probability that node i belongs to a small cluster of any size is $H_i(1) = \sum_s \pi_i(s)$. The expected fraction S of the network taken up by the percolating cluster is $S = 1 - \frac{1}{n}\sum_i H_i(1)$. This is the case since, if it does not belong to a small cluster, then a node must be in the percolating cluster. The expected size of the clusters that $i\in\mathcal{V}$ belongs to is

$$\begin{aligned}\langle s_i\rangle = \sum_{j\in(N_i\setminus\{i\})/\sim}\sum_{k\in\overline{i\cap j}\setminus\{i\}}\sum_{q\in(N_k\setminus\{i,j\})/\sim}\partial_{\overline{i\cap j}}G_i\big(\mathbf{H}_{\to i}(1)\big)\times\\ \partial_k G_{\overline{i\cap j}\to i}\big(\mathbf{H}_{\overline{i\cap j}\to i}(1)\big)\partial_{\overline{k\cap q}}H_{\neg(\overline{i\cap j})\to k}(1)H'_{\overline{k\cap q}\to k}(1) + H_i(1)\end{aligned}$$

where H' is the derivative of H, $\partial_{\overline{i\cap j}}G_i$ is the partial derivative of G_i with respect to its jth argument, and the same holds for $\partial_k G_{\overline{i\cap j}\to i}$ and $\partial_{\overline{k\cap q}}H_{\neg(\overline{i\cap j})\to k}$.

$H'_{\overline{k\cap q}\to k}(1)$ can be found by differentiating Eq. (6), setting $z=1$, and iterating the self-consistent equations

$$\begin{aligned}H'_{\overline{k\cap q}\to k}(1) = \sum_{s\in\overline{k\cap q}\setminus\{k\}}\sum_{v\in(N_s\setminus\{k,q\})/\sim}\partial_s G_{\overline{k\cap q}\to k}\big(\mathbf{H}_{\overline{k\cap q}\to k}(1)\big)\\ \times\,\partial_{\overline{s\cap v}}H_{\setminus(\overline{k\cap q})\to s}(1)H'_{\overline{s\cap v}\to s}(1) + H_{\overline{k\cap q}\to k}(1)\end{aligned} \tag{7}$$

until convergence.

Since the loop bound is fulfilled, the equations are exact. Moreover, they provide an advantage regarding time complexity compared to the KCN-method [2]: Instead of summing over N_i and $N_{j\setminus i}$, we only sum over $N_{i\cap j}$. In fact, following [4, Claim 4], we can show that the approach to percolation in this section is optimal in terms of time complexity.

5.2 r-unbounded loops

Regarding the message passing equations, we use

$$G^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}_{k\cap q\to i\cap j}(y) \equiv \Big\langle\prod_{s\in N_{k\cap q}} y_s^{w_{ks}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}}\Big\rangle_{\Gamma_{k\cap q}}$$

and $H_{k\cap q\to i\cap j}(z) \equiv zG_{k\cap q\to i\cap j}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}\big(\mathbf{H}_{k\cap q\to i\cap j}(z)\big)$, where we have introduced the random variable $w_{ks}^{\overline{\mathcal{P}_{i\cap j}}}$, which takes the value 1 if there is a path of occupied edges within $\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})$ from k to s, and zero otherwise. Moreover, we use the notation $H_{p\to k\cap q}(z) \equiv \prod_{s\in N_p\setminus\{p\}} H_{p\cap s\to k\cap q}(z)$ and $\mathbf{H}_{k\cap q\to i\cap j}(z) \equiv \big(H_{p\to k\cap q}(z)\big)_{p\in N_{k\cap q}\setminus\{k\}}$.

For the inference stage, we use the equation $H_i(z) = zG_i^{\overline{\mathcal{Q}_i}}(\mathbf{H}_{\to i}(z))$, where

$$G_{i\cap j\to i}^{\overline{\mathcal{Q}_i}(N_{i\cap j})}(\mathbf{y}) \equiv \Big\langle \prod_{k\in N_{i\cap j}\setminus\{i\}} y_k^{w_{ik}^{\overline{\mathcal{Q}_i}(N_{i\cap j})}} \Big\rangle_{\Gamma_{i\cap j}}$$

and $G_i^{\overline{\mathcal{Q}_i}}(\mathbf{y}) \equiv \prod_{j\in N_i\setminus\{i\}} G_{i\cap j\to i}^{\overline{\mathcal{Q}_i}(N_{i\cap j})}\big(y_j\big)$. Moreover, we use the notation $\mathbf{H}_{i\cap j\to i}(z) \equiv \big(H_{k\to i\cap j}(z)\big)_{k\in N_{i\cap j}\setminus\{i\}}$ and $\mathbf{H}_{\to i}(z) \equiv \big(\mathbf{H}_{i\cap j\to i}(z)\big)_{j\in N_i\setminus\{i\}}$.

To conclude this section, we only ought to show how to compute the expected value of s_i. We can do so using the following equation

$$\begin{aligned}\langle s_i\rangle \equiv H_i(1) + \sum_{j\in N_i\setminus\{i\}} \sum_{k\in N_{i\cap j}\setminus\{i\}} \sum_{q\in N_k} \partial_{i\cap j} G_i^{\overline{\mathcal{Q}_i}}\big(\mathbf{H}_{\to i}(1)\big) \\ \times \partial_k G_{i\cap j\to i}^{\overline{\mathcal{Q}_i}(N_{i\cap j})}\big(\mathbf{H}_{i\cap j\to i}(1)\big)\partial_{k\cap q} H_{k\to i\cap j}(1) H'_{k\cap q\to i\cap j}(1),\end{aligned}$$

where $H'_{k\cap q\to i\cap j}(1)$ can be found by iterating the self-consistent equations

$$\begin{aligned}H'_{k\cap q\to i\cap j}(1) \equiv \sum_{s\in k\cap q\setminus\{k\}} \sum_{v\in N_s} \partial_s G_{k\cap q\to i\cap j}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}\big(\mathbf{H}_{k\cap q\to i\cap j}(1)\big) \\ \times \partial_{s\cap v} H_{s\to k\cap q}(1) H'_{s\cap v\to k\cap q}(1) + H_{k\cap q\to i\cap j}(1)\end{aligned}$$

Since the loop bound is not fulfilled, these equations are only approximate. The time complexity advantage compared to the KCN-method remains, and the accuracy does not decrease provided we consider **locally dense and globally sparse** networks [4, Claim 5]. In general, the equations in the unbounded KCN and NIB methods may be different, and part of the extra complexity in the KCN may be used to compute some correlations more precisely.

6 Matrix spectrum via the NIB-method

6.1 r-bounded loops

If the loop bound is fulfilled, then any i-excursion can be decomposed as an i-excursion w_i within some equivalence class $w_i \subseteq \overline{i\cap j}$ together with some number of additional closed walks outside $\overline{i\cap j}$ that each start at some node $k \in (w_i \cap \overline{i\cap j}) \setminus \{i\}$ and return some time later to k. Since the loop bound is fulfilled, the additional walks must return to the same node they started at. We give an example of such an excursion in Figure 3.

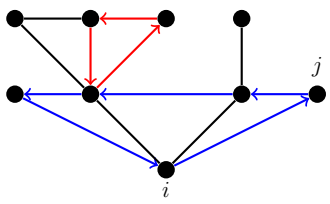


Fig. 3: Decomposition of an i-excursion in an i-excursion within $\overline{i \cap j}$ (in blue) and a closed walks outside $\overline{i \cap j}$ (in red).

To fix some notation, we assume that the length of the i-excursion w_i is $l+1$, that is, w_i visits ℓ (not necessarily distinct) nodes $j_1, \ldots, j_\ell \in \overline{i \cap j} \setminus \{i\}$ within the equivalence class other than the starting node i. Moreover, we take $s_{\overline{j\cap k}\to j}$ to be the length of some closed walk (if it exists) that starts and ends at $j \in w_i \setminus \{i\}$ and does not traverse any edges in some equivalence class containing j that is different from $\overline{j \cap k}$. If no such a walk exists, then we take $s_{\overline{j\cap k}\to j}$ to be zero.

The total length of a **non-trivial** i-excursion w_i[4] will be

$$\ell + 1 + \textstyle\sum_{j\in w_i\setminus\{i\}, k:\overline{j\cap k}\in\cap/\sim\setminus\{\overline{i\cap j}\}} s_{\overline{j\cap k}\to j},$$

and the sum of the weights of all excursions of length s with w_i as their foundation will be

$$\begin{aligned} &|w_i| \sum_{\{s_{\overline{j\cap k}\to j}: j\in w_i\setminus\{i\}, k:\overline{j\cap k}\in\cap/\sim\setminus\{\overline{i\cap j}\}} \prod_{j\in w_i\setminus\{i\}} \prod_{k:\overline{j\cap k}\in\cap/\sim\setminus\{\overline{i\cap j}\}} X^{s_{\overline{j\cap k}\to j}}_{\overline{j\cap k}\to j} \\ &\times \delta\big(s, \ell+1+\textstyle\sum_{j\in w_i\setminus\{i\}, k:\overline{j\cap k}\in\cap/\sim\setminus\{\overline{i\cap j}\}} s_{\overline{j\cap k}\to j}\big), \end{aligned} \tag{8}$$

where $|w_i|$ is the weight of w_i, and $X^s_{\overline{j\cap k}\to j}$ is the sum of weights of length-s j-walks if the equivalence classes different from $\overline{j \cap k}$ that j belongs to are removed from the graph. In fact, following the derivation of Eq. (1), we have

$$X^s_{\overline{j\cap k}\to j} = \sum_{m=0}^{\infty} \left[\sum_{s_1=1}^{\infty} \cdots \sum_{s_m=1}^{\infty} \delta\big(s, \textstyle\sum_{u=1}^m s_u\big) \prod_{u=1}^m Y^{s_u}_{\overline{j\cap k}\to j} \right], \tag{9}$$

where, after the removal of the the equivalence classes different from $\overline{j \cap k}$ that j belongs to, the sum of weights of length-s j-excursions is denoted by $Y^s_{\overline{j\cap k}\to j}$. As a last step, we note that Y^s_i can be decomposed as follows:

$$\begin{aligned} Y^s_i = [\mathbf{A}]_{ii}\delta(s,1) + \sum_{j\in(N_i\setminus\{i\})/\sim} \sum_{\ell_{\overline{i\cap j}}=0}^{\infty} \sum_{w_{\overline{i\cap j}}\in W^{\ell_{\overline{i\cap j}}}_{\overline{i\cap j}}} |w_{\overline{i\cap j}}| \sum_{\{s_{\overline{k\cap q}\to k}: k\in w_{\overline{i\cap j}}\setminus\{i\}, q:\overline{k\cap q}\in\cap/\sim\setminus\{\overline{i\cap j}\}\}} \\ \times \prod_{k\in w_{\overline{i\cap j}}\setminus\{i\}} \prod_{q:\overline{k\cap q}\in\cap/\sim\setminus\{\overline{i\cap j}\}} X^{s_{\overline{k\cap q}\to k}}_{\overline{k\cap q}\to k} \delta\big(s, \ell_{\overline{i\cap j}} + 1 + \textstyle\sum_{k\in w_{\overline{i\cap j}}\setminus\{i\}, q\in(N_k\setminus\{i,j\})/\sim} s_{\overline{k\cap q}\to k}\big), \end{aligned} \tag{10}$$

[4] By non-trivial we mean $w_i \not\subseteq \overline{i \cap i}$, since otherwise the length is one by definition.

where $W_{\overline{i\cap j}}^{\ell_{\overline{i\cap j}}}$ is the set of i-excursions of length $\ell_{\overline{i\cap j}}+1$ when, except for $\overline{i\cap j}$, the edges in all equivalence classes that i belongs to are removed. By putting together Eqs. (9) and (10), we obtain

$$H_i(z) = [\mathbf{A}]_{ii} + \sum_{j\in(N_i\setminus\{i\})/\sim} \sum_{w_{\overline{i\cap j}}\in W_{\overline{i\cap j}}} |w_{\overline{i\cap j}}| \prod_{k\in w_{\overline{i\cap j}}\setminus\{i\}} \prod_{q:\overline{k\cap q}\in\cap/\sim\setminus\{\overline{i\cap j}\}} \frac{1}{z - H_{\overline{k\cap q}\to k}(z)}, \tag{11}$$

where $W_{\overline{i\cap j}} \equiv \bigcup_{\ell_{\overline{i\cap j}}} W_{\overline{i\cap j}}^{\ell_{\overline{i\cap j}}}$, and we take $H_{\overline{k\cap q}\to k}(z) \equiv \sum_{s=1}^{\infty} \frac{Y_{\overline{k\cap q}\to k}^{s}}{z^{s-1}}$. In the same vein, we get that

$$H_{\overline{k\cap q}\to k}(z) = \sum_{w_{\overline{k\cap q}}\in W_{\overline{k\cap q}}} |w_{\overline{k\cap q}}| \prod_{s\in w_{\overline{k\cap q}}\setminus\{k\}} \prod_{v:\overline{s\cap v}\in\cap/\sim\setminus\{\overline{k\cap q}\}} \frac{1}{z - H_{\overline{s\cap v}\to s}(z)} \tag{12}$$

if $k\neq q$ and $H_{\overline{k\cap q}\to k}(z) = [\mathbf{A}]_{kk}$ if $k=q$.

Equation (12) is our message passing scheme for the spectral density: We begin with suitable starting values and then iterate these equations to convergence. Once converged, we infer $H_i(z)$ through Eq. (11) and then use it to compute the spectral density.

In order to make this approach practical, we ought to have some efficient method to evaluate the sum in Eq. (12) [2, Supplementary Material]:

We begin by considering $\mathbf{v}_{\overline{k\cap q}\to k,v}$ the vector of matrix elements associated to edges connected to k in $\overline{k\cap q}$, $\mathbf{v}_{\overline{k\cap q}\to k,v} \equiv [\mathbf{A}]_{kv}$ if $(k,v)\in\overline{k\cap q}$, and $\mathbf{v}_{\overline{k\cap q}\to k,v} \equiv 0$ otherwise, and by defining $\mathbf{A}^{\overline{k\cap q}}$ the adjacency matrix of the neighborhood of $\overline{k\cap q}$, $\left[\mathbf{A}^{\overline{k\cap q}}\right]_{sv} \equiv [\mathbf{A}]_{sv}$ if $s,v\neq k$ and $(s,v)\in\overline{k\cap q}$, and zero otherwise. Lastly, we let $\mathbf{D}^{\overline{k\cap q}\to k}(z)$ be the diagonal matrix with entries

$$\left[D^{\overline{k\cap q}\to k}(z)\right]_{ss} \equiv \prod_{\overline{s\cap v}\neq\overline{k\cap q}} \left(z - H_{\overline{s\cap v}\to s}(z)\right),$$

and we obtain that, for $k\neq q$, Equation (12) can then be written as

$$H_{\overline{k\cap q}\to k}(z) = \mathbf{v}_{\overline{k\cap q}\to k}^{T}\left(\mathbf{D}^{\overline{k\cap q}\to k}(z) - \mathbf{A}^{\overline{k\cap q}}\right)^{-1}\mathbf{v}_{\overline{k\cap q}\to k}. \tag{13}$$

Since the loop bound is fulfilled, the equations in this section provide exact results. Moreover, they provide an advantage regarding time complexity compared to the matrix spectra version of the KCN-method [2]. In fact, following [4, Claim 4], we can again show that the approach to matrix spectra in this section is optimal in terms of time complexity.

6.2 r-unbounded loops

Regarding message passing, and aside from the trivial messages $H_{k\cap k\to i\cap j}(z) = [\mathbf{A}]_{kk}$ for all z, the analogous of (13) is

$$H_{k\cap q\to i\cap j}(z) \equiv (\mathbf{v}_{k\cap q\to i\cap j}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})})^{T}\left(\mathbf{D}^{k\cap q\to i\cap j} - \mathbf{A}_{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}^{k\cap q\to i\cap j}\right)^{-1}\mathbf{v}_{k\cap q\to i\cap j}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})},$$

where $\mathbf{v}^{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}_{k\cap q\to i\cap j,v} \equiv [\mathbf{A}]_{kv}$ if $(k,v) \in N_{k\cap q} \setminus \overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})$, and it equals zero otherwise; and $\left[\mathbf{A}^{k\cap q\to i\cap j}_{\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})}\right]_{sv} \equiv [\mathbf{A}]_{sv}$ if $s, v \neq k$ and$(s,v) \in N_{k\cap q}\setminus\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})$, and it equals zero otherwise. Moreover, the matrix $\mathbf{D}^{k\cap q\to i\cap j}(z)$ is diagonal

$$\left[\mathbf{D}^{k\cap q\to i\cap j}(z)\right]_{ss} \equiv \prod_{v\in N_s} \left(z - H_{s\cap v\to k\cap q}(z)\right).$$

To conclude, the inference formula (11) for H_i becomes

$$H_i(z) = [\mathbf{A}]_{ii} + \sum_{j\in N_i\setminus\{i\}} \sum_{w_{i\cap j}\in W_{i\cap j\setminus\overline{\mathcal{Q}_i}(N_{i\cap j})}} |w_{i\cap j}|\times$$
$$\prod_{k\in w_{i\cap j}\setminus\{i\}} \prod_{q\in N_k: N_{k\cap q}\setminus\overline{\mathcal{P}_{i\cap j}}(N_{k\cap q})\neq\emptyset} \frac{1}{z - H_{k\cap q\to i\cap j}(z)},$$

where $W_{i\cap j\setminus\overline{\mathcal{Q}_i}(N_{i\cap j})}$ is the set of i-excursions that use edges within $N_{i\cap j} \setminus \overline{\mathcal{Q}_i}(N_{i\cap j})$.

Since the loop bound is not fulfilled, the equations only provide approximate results. The time complexity advantage compared to the KCN-method remains provided we consider locally dense and globally sparse networks [4, Claim 5], although the equations may be different, and part of the extra complexity in the KCN may be used to compute some correlations more precisely.

7 Conclusion

We have extended the NIB-method to percolation and the computation of matrix spectra, showing that one can also achieve an improvement on the KCN-method in these applications. If either the loop bound is fulfilled or it is not fulfilled but the graph is locally dense and globally sparse, then the improvement can be shown analytically, as we have argued. If the loop bound is not fulfilled, then it is reasonable to assume that the numerical evidence comparing the performance of the KCN and NIB methods in the context of probabilistic graphical models will extend to the applications discussed here. However, providing such numerical evidence remains a task for the future.

Concerning percolation, it was already argued [2] that the KCN-method and classical direct simulations compute different quantities. That is, while the latter only considers a single realized graph and one would need to perform several runs in order to obtain average values, the former directly provides averaged values. The NIB-method also computes averaged values as well. After the introduction of the KCN-method, a motif-based message passing approach [8] was developed. Although it was conceived for a different purpose, it is important to note that its message passing algorithm is limited to some specific graphs and it requires some graph-dependent analytical derivations. This contrasts with the generality of the KCN and NIB methods. Concerning matrix spectra, the KCN-method can substantially outperform traditional methods [2], thus enabling the computation

of the spectra of some previously inaccessible large systems. The NIB-method can extend the set of accessible systems even further.

In the context of inference, it would be important to extend the NIB-method from networks to general graphical models. It would be interesting to extend the KCN and NIB methods to other applications, like epidemic models or graph coloring. A very interesting use case could be the computation of thresholds in the context of quantum error correction and the erasure channel [14], which is closely related to percolation. The application of these methods to compute the spectra of non-symmetric matrices has also not been developed yet.

References

1. Bianconi, G., Dorogovtsev, S.N.: Theory of percolation on hypergraphs. Physical Review E **109**(1), 014,306 (2024)
2. Cantwell, G.T., Newman, M.E.: Message passing on networks with loops. Proceedings of the National Academy of Sciences **116**(47), 23,398–23,403 (2019)
3. Castro Guzman, G.E., Stadler, P.F., Fujita, A.: A message-passing approach to obtain the trace of matrix functions with applications to network analysis. Numerical Algorithms pp. 1–22 (2025)
4. Hack, P.: Belief propagation for networks with loops: The neighborhoods-intersections-based method. arXiv preprint arXiv:2506.13791 (2025)
5. Hack, P., Mendl, C.B., Paler, A.: Belief propagation for general graphical models with loops. arXiv preprint arXiv:2411.04957 (2024)
6. Kirkley, A., Cantwell, G.T., Newman, M.: Belief propagation for networks with loops. Science Advances **7**(17), eabf1211 (2021)
7. Liu, Y.H., Poulin, D.: Neural belief-propagation decoders for quantum error-correcting codes. Physical review letters **122**(20), 200,501 (2019)
8. Mann, P., Dobson, S.: Belief propagation on networks with cliques and chordless cycles. Physical Review E **107**(5), 054,303 (2023)
9. Mezard, M., Montanari, A.: Information, physics, and computation. Oxford University Press (2009)
10. Nadakuditi, R.R., Newman, M.E.: Spectra of random graphs with arbitrary expected degrees. Physical Review E—Statistical, Nonlinear, and Soft Matter Physics **87**(1), 012,803 (2013)
11. Newman, M.: Message passing methods on complex networks. Proceedings of the Royal Society A **479**(2270), 20220,774 (2023)
12. Newman, M., Ziff, R.M.: Efficient monte carlo algorithm and high-precision results for percolation. Physical Review Letters **85**(19), 4104 (2000)
13. Richardson, T., Urbanke, R.: Modern coding theory. Cambridge university press (2008)
14. Stace, T.M., Barrett, S.D., Doherty, A.C.: Thresholds for topological codes in the presence of loss. Physical review letters **102**(20), 200,501 (2009)
15. Stauffer, D., Aharony, A.: Introduction to percolation theory. Taylor & Francis (2018)
16. Xiong, K., Dong, H., Liu, Y., Zhou, M., Liu, W.: Regulation of thermal transport by cycle structures in complex networks. Chaos, Solitons & Fractals **191**, 115,766 (2025)
17. Yedidia, J.S., Freeman, W., Weiss, Y.: Generalized belief propagation. Advances in neural information processing systems **13** (2000)