Cross-validation estimates of the number of modules in higher-order networks

Tatsuro Kawamoto

Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology, 2-3-26 Aomi, Koto-ku, Tokyo, Japan

E-mail: kawamoto.tatsuro@aist.go.jp

Abstract. A graph clustering algorithm detects macroscopic patterns of connectivity hidden in network data. We regard each vertex in the network as belonging to a module, and the vertices in the same module have a statistically equivalent connection pattern. While a network with binary labels—e.g., similar or unrelated—is often considered as the simplest example, network data with finer information about the relationships—e.g., similar, unrelated, and dissimilar—are common in practice. For this reason, we consider the graph clustering of networks with labeled edges. In graph clustering, selecting the number of modules that are considered to be statistically significant is a crucial step. In other words, it determines the level of coarse graining. Here, we propose efficient cross-validation estimates for the number of modules based on the prediction errors. While the straightforward implementation of cross-validation is typically computationally demanding, we show that leave-one-out cross-validation has a good affinity with the belief propagation algorithm, and thus can be measured efficiently.

1. Introduction

Suppose that you collect relational data among the people in an international conference so that you can build a social network. Each vertex represents a participant of the conference, and each edge represents the relationship between a pair of participants. You ask a participant whether a pair of other participants are friends with each other (and believe his/her information); the corresponding vertices are connected via an undirected edge if they are friends and are not connected otherwise. You conduct this survey to multiple participants until you obtain a sufficient number of edges. From the resulting social network, we expect that a graph clustering algorithm detects social groups among the participants.

However, when the number of participants $N$ is very large, it is difficult to obtain the relational data of $O(N^2)$. Thus, the resulting network will typically be very sparse. When the network is sparse, because we do not have sufficient information on each vertex, it is a difficult task to infer the social groups accurately. A simple solution is to introduce a new edge type, “enemy,” and distinguish the unconnected pairs into “enemy” and “unknown,” so that we can increase the number of edges. We can also ask for more details when we collect the relationships and add more edges of different types.

To detect the module structure from sparse networks, we consider the statistical inference of modules using the so-called stochastic blockmodel (SBM) [1]. The SBM is a random graph model with a module structure. In the SBM, each vertex has a planted module assignment, and the vertices in the same label have a statistically equivalent connection pattern. In particular,
we consider a variant of the SBM that has labeled edges; it is called the labeled SBM. An instance of the labeled SBM is shown in Fig. 1. Note that if we increase the number of modules $q$ to which vertices can belong, the SBM becomes more flexible so that it is easier to fit the real network. However, we obviously do not want to set each vertex to a different module—i.e., $q = N$. Therefore, we wish to find the parsimonious number of modules $q^*$ such that the corresponding SBM efficiently summarizes the network structure.

Here, we extend the cross-validation estimate for the number of modules that we proposed [6] for the networks with binary edges (edges and nonedges) to the ones with labeled edges. Our cross-validation estimate is closely related to the inference algorithm. We consider the expectation-maximization (EM) algorithm with belief propagation (BP) for its E-step. By utilizing the quantities we obtained as a result of the algorithm, we can efficiently perform leave-one-out cross-validation (LOOCV).

The rest of the article is organized as follows. First, we describe the generation of the labeled SBM in detail (Sec. 2) and the specific inference algorithm (Sec. 3). We then explain the method of cross-validation estimate for the networks with labeled edges (Sec. 4), and investigate the performance of the proposed estimates (Sec. 5). We confirm the performance of the cross-validation estimates through numerical experiments (Sec. 6). The last section is devoted to summary and discussion (Sec. 7).

2. Labeled stochastic blockmodel

We consider a network with vertex set $V$ and edge set $E$. The number of vertices is $N$, and the total number of edges is $L$. The edge set comprises $p$ types of edge sets, so $E = \cup_{i=1}^{p} E_{\alpha}$, and we denote the number of $\alpha$-edges as $L_{\alpha}$ (i.e., $\sum_{\alpha} L_{\alpha} = L$). We also refer to the nonedges as the edges with $\alpha = 0$. We denote the adjacency matrix of a network as $A$, i.e., $A_{ij} = \alpha$ if vertex pair $i$ and $j$ is connected via an $\alpha$-edge. We denote the average degree of $\alpha$-edges as $c_{\alpha}$ and $c = \sum_{\alpha} c_{\alpha}$ as the average degree of all edges.

The labeled SBM of $q$ planted modules is generated as follows. We denote the module label as $\sigma \in \{1, \ldots, q\}$. For each vertex in the vertex set $V$, we assign a module assignment $\sigma_i$ ($i \in V$) independently and randomly with probability $p(\sigma_i|\gamma) = \gamma_{\sigma_i}$, where $\sum_{\sigma=1}^{q} \gamma_{\sigma} = 1$. We then generate edges independently based on the matrices called the affinity matrices $\{c_{\alpha}\}$. Each affinity matrix $c_{\alpha}$ represents the connection pattern between/within modules. For example, vertices $i$ and $j$ are connected via an $\alpha$-edge ($\alpha > 0$) with probability $c_{\alpha,\sigma_i,\sigma_j}/N$. Therefore, the
where the vertex \( k \)

\[ p(\mathbf{A}, \mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \}) = \prod_{i=1}^{N} \gamma_{\sigma_i} \prod_{i<j}^{p} \frac{\left( \sum_{\sigma \in \{ \alpha \}} \delta_{A_{ij}, \sigma} \right)}{N}. \quad (1) \]

Note that because each vertex pair is connected via a nonedge or one of the \( \alpha \)-edges, the elements of the affinity matrices are normalized as \( \sum_{\alpha=0}^{N} \phi_{\alpha} / N = 1 \) for any \( \alpha \) and \( \alpha' \). We consider that each element of \( \phi^\alpha \) is of \( O(1) \) except for \( \alpha = 0 \), so the resulting network is sparse.

### 3. EM algorithm with BP

We infer the module assignments using the EM algorithm [3]. For the statistical inference of the labeled SBM, we aim to infer the posterior marginal probability \( p(\sigma_i | \mathbf{A}, \gamma, \{ \mathbf{c}^\alpha \}) \) of the module assignment for each vertex and learn the model parameters so that the marginal log-likelihood \( \log p(\mathbf{A} | \gamma, \{ \mathbf{c}^\alpha \}) \) is maximized (i.e., we perform the point estimate for the model parameters).

Using the variational expression, the marginal log-likelihood can be written as

\[ \log \sum_{\alpha} p(\mathbf{A}, \mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \}) = \mathbb{E}_{\psi} \left[ \log \frac{p(\mathbf{A}, \mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \})}{\psi} \right] + D_{\text{KL}} \left[ \psi \parallel p(\mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \}) \right], \quad (2) \]

where \( \psi \) is the variational distribution of the module assignments \( \sigma_i \), and \( \mathbb{E}_{\psi} \) represents the average with respect to \( \psi \). \( D_{\text{KL}} [P || Q] \) is the Kullback-Leibler divergence between \( P \) and \( Q \). In the EM algorithm, in the E-step, we evaluate the variational distribution \( \psi \) so that \( D_{\text{KL}} \left[ \psi || p(\mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \}) \right] \) vanishes (or is minimized). In the M-step, we update the model parameters so that the first term of Eq. (2) is maximized. After the model-parameter update, the variational distribution \( \psi \) that minimizes \( D_{\text{KL}} \left[ \psi || p(\mathbf{A} \mid \gamma, \{ \mathbf{c}^\alpha \}) \right] \) may be varied. Thus, we iterate the E-step and M-step until convergence. Hereafter, we often omit the hat notation that we use for the estimated quantities.

#### 3.1. E-step

To perform the E-step, we use BP, which is an approximate algorithm that solves for the marginal distribution of a probabilistic model with a treelike structure [4, 5]. We expect that the performance of BP is good because the networks we consider are sparse. Following the standard formulation of BP, the estimate of the marginal probability \( \psi^i_\alpha \) with respect to the likelihood (1) for vertex \( i \) is expressed as

\[ \psi^i_\alpha = \frac{\gamma_{\sigma_i}}{Z^i} \prod_{\ell \in [\partial i]^0}^{\alpha} \left( 1 - \sum_{\sigma \in \{ \sigma \}} \psi^j_{\sigma \ell} c_{\sigma \sigma}^{\alpha} \right) \prod_{\alpha > 0} \prod_{k \in [\partial i]^0} \sum_{\sigma_k} \psi^k_{\sigma_k} c_{\sigma_k \sigma}^{\alpha} \approx \frac{\gamma_{\sigma_i}}{\sum_{\alpha > 0} \psi^i_{\alpha \sigma}}, \quad (3) \]

\[ \varphi^{i-j}_{\alpha, \sigma} = e^{-N^{-1} \sum_{\ell \in [\partial i]^0} \sum_{\sigma} \psi^j_{\sigma \ell} c_{\sigma \sigma}^{\alpha}} \prod_{k \in [\partial j]^0} \sum_{\sigma_k} \psi^j_{\sigma_k} c_{\sigma_k \sigma}^{\alpha}, \quad (4) \]

where the vertex \( k \in [\partial j]^0 \) is a neighbor of \( i \) such that \( (i, k) \in E_{\alpha} \), \( Z^i \) is a normalization factor of \( \psi^i_\alpha \), and

\[ \psi^i_{\alpha} = \frac{\gamma_{\sigma_i}}{Z^{i-j}} \prod_{\ell \in [\partial i \cup j]^0}^{\alpha} \left( 1 - \sum_{\sigma \in \{ \sigma \}} \psi^j_{\sigma \ell} c_{\sigma \sigma}^{\alpha} \right) \prod_{\alpha > 0} \prod_{k \in [\partial i \cup j]^0} \sum_{\sigma_k} \psi^k_{\sigma_k} c_{\sigma_k \sigma}^{\alpha} \approx \frac{\gamma_{\sigma_i}}{\sum_{\alpha > 0} \psi^i_{\alpha \sigma}}, \quad (5) \]

\[ \varphi^{i-j}_{\alpha, \sigma} = e^{-N^{-1} \sum_{\ell \in [\partial i \cup j]^0} \sum_{\sigma} \psi^j_{\sigma \ell} c_{\sigma \sigma}^{\alpha}} \prod_{k \in [\partial j]^0} \sum_{\sigma_k} \psi^j_{\sigma_k} c_{\sigma_k \sigma}^{\alpha}, \quad (6) \]
Analogously, the vertex \( k \in [i] \) is a neighbor of \( i \) such that \((i, k) \in E \) but \( k \neq j \) and \( Z^{i \rightarrow j} \) is a normalization factor of \( \psi_{\sigma}^{i \rightarrow j} \).

The probability \( \psi_{\sigma}^{i \rightarrow j} \) represents the marginal probability of vertex \( i \) with the missing knowledge about the edge \((i, j)\), which is called the cavity bias. Because Eq. (5) constitutes a set of close equations with respect to the cavity biases, we iteratively update them until convergence. We then substitute the result into Eq. (3) to obtain the full marginal.

3.2. M-step

To derive the updated equation of the M-step, as we mentioned above, we take the extremum point of the first term in Eq. (2). As a result, we have the estimate \( \hat{\gamma}_\sigma \) for the fraction of the module size as

\[
\hat{\gamma}_\sigma = \frac{1}{N} \sum_{i=1}^{N} \psi_{\sigma}^{i}. \tag{7}
\]

For the estimate \( \hat{c}_{\sigma \sigma'}^{(\text{new})} \) of an element of the affinity matrix \( c_{\sigma}^{(\text{new})} \), we have

\[
\hat{c}_{\sigma \sigma'}^{(\text{new})} = c_{\alpha} \left( \frac{1}{\hat{\gamma}_\sigma \hat{\gamma}_{\sigma'}} \sum_{(i,j) \in E} \sum_{\sigma \sigma'} \hat{c}_{\sigma \sigma'}^{\alpha} \left( \psi_{\sigma}^{i \rightarrow j} \psi_{\sigma'}^{j \rightarrow i} + \psi_{\sigma}^{i \rightarrow j} \psi_{\sigma'}^{j \rightarrow i} \right) \right), \tag{8}
\]

where \( \hat{c}_{\sigma \sigma'}^{(\text{new})} \) is the updated estimate, which is written in terms of the estimate from the previous update.

We then substitute these updated values of the model parameters into the BP equations (3) and (5) to update the E-step again.

4. Cross-validation estimates of the prediction errors

To perform the EM algorithm above, we must specify the number of modules \( q \) as an input. While evaluating the saturation of the free energy (or the Bethe free energy [3]) is a choice, it is empirically known that it tends to overfit for real-world networks. For this reason, we measure the prediction errors using the leave-one-out cross-validation (LOOCV) [7]. While the straightforward implementation of cross-validation is computationally demanding, LOOCV is an exceptional case in which the prediction errors can be efficiently estimated using the results of BP.

4.1. Bayes prediction error

The prediction error that we consider is the predictability of edge information. To this end, we measure the posterior distribution of \( A_{ij} \) given the adjacency matrix except for \( A_{ij} \) (we denote it as \( A^{\setminus(i,j)} \)). Because the edge is generated based on the module assignments of the vertex pair in the SBM, we have

\[
p(A_{ij} = \alpha|A^{\setminus(i,j)}) = \sum_{\sigma_i, \sigma_j} p(A_{ij} = \alpha|\sigma_i, \sigma_j)p(\sigma_i, \sigma_j|A^{\setminus(i,j)}), \tag{9}
\]

For \( \alpha > 0 \), \( p(A_{ij} = \alpha|\sigma_i, \sigma_j) = c_{\sigma_i \sigma_j}^{\alpha}/N \) by the model definition and \( p(\sigma_i, \sigma_j|A^{\setminus(i,j)}) \) can be estimated using the result of BP as

\[
p(\sigma_i, \sigma_j|A^{\setminus(i,j)}) = p(\sigma_i|A^{\setminus(i,j)})p(\sigma_j|A^{\setminus(i,j)}) = \psi_{\alpha}^{i \rightarrow j} \psi_{\alpha}^{j \rightarrow i}. \tag{10}
\]
We then average over the selection of the module assignments. That is, 

\[
p(A_{ij} = \alpha | A^{\setminus(i,j)}) = \frac{1}{N} \sum_{\sigma_i, \sigma_j} \psi_{\sigma_i}^{i \rightarrow j} \frac{c_{\sigma_i \sigma_j}}{N} \psi_{\sigma_j}^{j \rightarrow i}
\]  

(11)

and we denote it as \( Z_{ij}^{\alpha} \).

We measure the prediction error using the cross-entropy error function \([2]\). We denote the cross-entropy with respect to \( p(A_{ij} = \alpha | A^{\setminus(i,j)}) \) that is normalized by \( L \) as the Bayes prediction error \( E_{\text{Bayes}}(q) \).

\[
E_{\text{Bayes}}(q) = \frac{1}{L} \sum_{i < j} \sum_{\alpha = 0}^{p} \delta_{A_{ij}, \alpha} \left[ -\log p(A_{ij} = \alpha | A^{\setminus(i,j)}) \right]
\]

(12)

Using the fact that the network is sparse, \( E_{\text{Bayes}}(q) \) can be approximated as follows.

\[
E_{\text{Bayes}}(q) \approx -\frac{1}{L} \sum_{\alpha > 0} \sum_{(i,j) \in E_n} \left( \log Z_{ij}^{\alpha} - \log(1 - \sum_{\alpha > 0} Z_{ij}^{\alpha}) \right) - \frac{1}{L} \sum_{i < j} \log(1 - \sum_{\alpha > 0} Z_{ij}^{\alpha}) \]

\[
\approx 1 - \frac{1}{L} \sum_{\alpha > 0} \sum_{(i,j) \in E_n} \log Z_{ij}^{\alpha},
\]

(13)

where we used the fact that \(-\log(1 - \sum_{\alpha > 0} Z_{ij}^{\alpha}) \approx \sum_{\alpha > 0} Z_{ij}^{\alpha}\) is of \(O(1/N)\) and is the probability that an edge exists between \(i\) and \(j\) irrespective of its label.

4.2. Gibbs prediction error

As shown in Eq. (9), an edge is generated via a two-step process in the SBM: i.e., the selection of module assignments and the generation of an edge based on the selected module assignments. When we measure the prediction error, instead of using \( p(A_{ij} = \alpha | A^{\setminus(i,j)}) \), we can measure the cross-entropy with respect to the probability \( p(A_{ij} = \alpha | \sigma_i, \sigma_j) \) with fixed module assignments. We then average over the selection of the module assignments. That is,

\[
E_{\text{Gibbs}}(q) = \frac{1}{L} \sum_{i < j} \sum_{\sigma_i, \sigma_j} p(\sigma_i, \sigma_j | A^{\setminus(i,j)}) \sum_{\alpha = 0}^{p} \delta_{A_{ij}, \alpha} \left[ -\log p(A_{ij} = \alpha | \sigma_i, \sigma_j) \right]
\]

(14)

We denote this estimate of the prediction error \( E_{\text{Gibbs}}(q) \) as the Gibbs prediction error. As with the Bayes prediction error \( E_{\text{Bayes}}(q) \), we rewrite \( E_{\text{Gibbs}}(q) \) using the sparse approximation, as follows.

\[
E_{\text{Gibbs}}(q) = -\frac{1}{L} \sum_{\alpha = 0}^{p} \sum_{(i,j) \in E_n} \sum_{\sigma_i, \sigma_j} \psi_{\sigma_i}^{i \rightarrow j} \psi_{\sigma_j}^{j \rightarrow i} \left( \log \frac{c_{\sigma_i \sigma_j}}{N} \right)
\]

\[
\approx 1 - \frac{1}{L} \sum_{\alpha = 1}^{p} \sum_{(i,j) \in E_n} \sum_{\sigma_i, \sigma_j} \psi_{\sigma_i}^{i \rightarrow j} \psi_{\sigma_j}^{j \rightarrow i} \log \frac{c_{\sigma_i \sigma_j}}{N} + \log N.
\]

(15)

Similarly, we can also define the so-called MAP estimate of the Gibbs prediction error. We denote it as \( E_{\text{MAP}}(q) \). It can be obtained by replacing \( \psi_{\sigma_i}^{i \rightarrow j} \) and \( \psi_{\sigma_j}^{j \rightarrow i} \) with the delta functions that is peaked at \( \arg \max \{ \psi_{\sigma_i}^{i \rightarrow j} \} \) and \( \arg \max \{ \psi_{\sigma_j}^{j \rightarrow i} \} \), respectively.
4.3. Gibbs training error

Finally, we can define the LOOCV estimate of the training error. In the Gibbs prediction error $E_{\text{Gibbs}}(q)$, we used the module-assignment probability $p(\sigma_i, \sigma_j | A^{(i,j)})$ without the knowledge of $A_{ij}$. On the other hand, it will be the training error if we utilize the actual information of $A_{ij}$ as well. We denote it as the Gibbs training error and it is defined as follows.

$$E_{\text{training}}(q) = \frac{1}{L} \sum_{i<j} \sum_{\sigma_i, \sigma_j} p(\sigma_i, \sigma_j | A) \sum_{\alpha=0}^{p} \delta_{A_{ij}, \alpha} \left[ -\log p(A_{ij} = \alpha | \sigma_i, \sigma_j) \right]$$

Using Bayes’ rule, $p(\sigma_i, \sigma_j | A)$ is recast as

$$p(\sigma_i, \sigma_j | A) = \frac{p(A_{ij}, \sigma_i, \sigma_j | A^{(i,j)})}{p(A_{ij} | A^{(i,j)})} = \frac{1}{Z_{\alpha}^{A_{ij}}} \psi_{i,j}^{\alpha} \frac{\psi_{i}^{\alpha} \psi_{j}^{\alpha}}{N}.$$ 

Again, we approximate $E_{\text{Gibbs}}(q)$ using the sparse approximation as follows.

$$E_{\text{training}}(q) \approx 1 - \frac{1}{L} \sum_{\alpha=1}^{p} \sum_{(i,j) \in E_{\alpha}} \frac{1}{Z_{\alpha}^{A_{ij}}} \sum_{\sigma_i, \sigma_j} \psi_{i,j}^{\alpha} \frac{\psi_{i}^{\alpha} \psi_{j}^{\alpha}}{N} \log \frac{\psi_{i}^{\alpha} \psi_{j}^{\alpha}}{N} + \log N.$$ 

Note that all cross-validation estimates that we introduced are written as averages over the set of edges $E$. Therefore, we can also measure their standard errors, and those errors can be useful when we employ a criterion such as the “one-standard-error rule” [8] in model selection. In the numerical experiments, we include the standard errors as the error bars of the cross-validation estimates.

4.4. Prediction error for $q = 1$

To evaluate whether the network has any module structure at all, it is informative to calculate the prediction error for $q = 1$. Note that because there are no model parameters to be learned when $q = 1$, there are no distinctions among the Bayes, Gibbs, and training errors. Thus, we refer to the prediction error as $E(q = 1)$.

Because we have $Z_{\alpha}^{A_{ij}} = p(A_{ij} = \alpha | A^{(i,j)}) \approx \frac{c_\alpha}{N}$, Eq. (13) yields

$$E(q = 1) \approx 1 - \frac{1}{L} \sum_{\alpha > 0} \sum_{(i,j) \in E_{\alpha}} \log \frac{c_\alpha}{N} = 1 - \log \frac{c}{N} + S[P_{\alpha}],$$

where we define $P_{\alpha} \equiv \frac{c_\alpha}{c}$ as the fraction of $\alpha$-edges, and $S[P_{\alpha}] \equiv -\sum_{\alpha > 0} P_{\alpha} \log P_{\alpha}$ is the entropy.

5. Performance of the LOOCV estimates of the prediction errors

In this section, we investigate the performance of each prediction error. In particular, we focus on the performance near the detectability threshold. The detectability threshold is the critical point in the strength of the module structure, below which the algorithm does not perform better than chance.

When we discuss the performance near the detectability threshold, we assume that the sizes of the planted module are equal. In this case, below the detectability threshold, the estimated probability of module label $\sigma$ is $1/q$ for any vertex; i.e., $\psi_{\sigma}^{\alpha} = \psi_{\gamma}^{\alpha} = \gamma_{\sigma} = 1/q$. Here, we do
not argue the values of model parameters at the detectability threshold themselves. Instead, we analyze how the prediction errors behave when $\psi^{i \rightarrow j}_{\sigma}$ is close to $1/q$. In the following, we assume that the planted model parameters are learned accurately, which is correct in the algorithmically detectable region.

To gain deeper insight about the performance of prediction errors, we focus on the case of simple community structure—i.e.,

$$c_{\sigma' \sigma} = \begin{cases} c_{\text{in}}^\alpha & (\sigma = \sigma') \\ c_{\text{out}}^\alpha & (\sigma \neq \sigma') \end{cases}.$$  

(20)

Here, we introduce $\Delta c_{\alpha} = c_{\text{in}}^\alpha - c_{\text{out}}^\alpha$. Under this parametrization, the Bayes prediction error $E_{\text{Bayes}}(q)$ in Eq. (13) is recast as

$$E_{\text{Bayes}}(q) = E(q = 1) - \sum_{\alpha > 0} P_{\alpha} \frac{1}{L_{\alpha}} \sum_{(i,j) \in E_{\alpha}} \log \left[ 1 + \frac{\Delta c_{\alpha}}{c_{\alpha}} \sum_{\sigma} (\psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma} - \gamma_{\sigma}^2) \right].$$  

(21)

The parenthesis in the second term indicates how much the estimate of the algorithm is correlated to the planted structure. In the case of edges that indicate an assortative structure, because the edge is more likely to connect the vertices of the same module assignment, $\psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma}$ is greater than $\gamma_{\sigma}^2$ (this is the case of the random guess). If this tends to be true, the overall sign of the second term in Eq. (21) is negative because $\Delta c_{\alpha} > 0$. Conversely, in the case of edges that indicate a disassortative structure, $\psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma}$ is smaller than $\gamma_{\sigma}^2$. If this tends to be true, the overall sign of the second term is again negative because $\Delta c_{\alpha} < 0$. Therefore, as long as the estimated module assignments are correlated with the planted structure, the Bayes prediction error $E_{\text{Bayes}}(q)$ is smaller than $E(q = 1)$ and supports partition of the network. When the set of model parameters reaches the detectability threshold, the second term in Eq. (21) vanishes, and we have $E_{\text{Bayes}}(q) = E(q = 1)$.

Let us next consider the behavior of the Gibbs prediction error $E_{\text{Gibbs}}(q)$. When the affinity matrix is given by Eq. (20), Eq. (15) reads

$$E_{\text{Gibbs}}(q) = E(q = 1) - \sum_{\alpha > 0} P_{\alpha} G_{\alpha},$$  

(22)

$$G_{\alpha} = \left( 1 - \frac{1}{L_{\alpha}} \sum_{\sigma} \psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma} \right) \log \left( 1 - \frac{\Delta c_{\alpha}}{c_{\alpha}} \gamma^\top \gamma \right)$$

$$+ \left( \frac{1}{L_{\alpha}} \sum_{\sigma} \psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma} \right) \log \left( 1 - \frac{\Delta c_{\alpha}}{c_{\alpha}} \gamma^\top \gamma + \frac{\Delta c_{\alpha}}{c_{\alpha}} \right),$$  

(23)

where we define

$$\left( \sum_{\sigma} \psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma} \right) \equiv \frac{1}{L_{\alpha}} \sum_{(i,j) \in E_{\alpha}} \sum_{\sigma} \psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma}$$  

(24)

and this is equal to $\gamma^\top \gamma$ at the detectability threshold. Note that $0 \leq \left( \sum_{\sigma} \psi^{i \rightarrow j}_{\sigma} \psi^{j \rightarrow i}_{\sigma} \right) \leq 1$ and $0 \leq \gamma^\top \gamma \leq 1$. Note also that the conditions $c_{\text{in}} > 0$ and $c_{\text{out}} > 0$ restrict the value of $\Delta c_{\alpha}/c_{\alpha}$ to

$$\frac{1}{1 - \gamma^\top \gamma} \leq \frac{\Delta c_{\alpha}}{c_{\alpha}} \leq \frac{1}{\gamma^\top \gamma}.$$  

(25)
When the planted structure is strong enough, we can confirm that $E_{\text{Gibbs}}(q)$ is smaller than $E(q = 1)$ and supports partitioning of the network. For the $\alpha$-edges with an assortative structure, $G_{\alpha} > 0$ because the first term in Eq. (23) is dominant and it gives a positive contribution. We also have $G_{\alpha} > 0$ for the $\alpha$-edges with a disassortative structure because the second term in Eq. (23) is dominant and it again gives a positive contribution.

When the planted structure is close to the detectability threshold, however, the Gibbs prediction error $E_{\text{Gibbs}}(q)$ becomes strictly greater than $E(q = 1)$. To see this, we recast $G$ as

$$G = \log \left( 1 - \frac{\Delta c_1}{c_1} \gamma^\top \gamma \right) + \left\langle \sum_\sigma \psi_\sigma^{i\to j} \psi_\sigma^{j\to i} \right\rangle_\alpha \log \left( 1 + \frac{\Delta c_1}{c_1} \gamma^\top \gamma \frac{1}{1 - \Delta c_1/c_1 \gamma^\top \gamma} \right).$$

(26)

Here, we set

$$x = \frac{\Delta c_1}{c_1} \gamma^\top \gamma \frac{1}{1 - \Delta c_1/c_1 \gamma^\top \gamma}, \quad y = \frac{1}{\gamma^\top \gamma},$$

(27)

and use the inequality $\log(1 + xy) < y \log(1 + x)$ ($x \geq 0, y > 1$). We then have

$$G_{\alpha} < \left( 1 - \frac{\left\langle \sum_\sigma \psi_\sigma^{i\to j} \psi_\sigma^{j\to i} \right\rangle_\alpha}{\gamma^\top \gamma} \right) \log \left( 1 - \frac{\Delta c_1}{c_1} \gamma^\top \gamma \right).$$

(28)

At the detectability threshold ($\left\langle \sum_\sigma \psi_\sigma^{i\to j} \psi_\sigma^{j\to i} \right\rangle_\alpha = \gamma^\top \gamma$), $G_{\alpha} < 0$. Thus, $E_{\text{Gibbs}}(q) > E(q = 1)$.

6. Numerical experiments

We numerically confirm the performance of the cross-validation estimates for the prediction errors. Two results of the labeled SBM are shown in Fig. 2. In both cases, the labeled SBM has two types of edges ($p = 2$) whose average degrees are $c_1 = 3$ and $c_2 = 5$. Figure 2a shows the result on the network with five planted modules. The values of model parameters are $\Delta c_1 / c_1 = 4.375$ and $\Delta c_2 / c_2 = -0.625$. The prediction errors, the Gibbs prediction errors in particular, are minimized or saturated at $q = 5$. On the other hand, 2b shows the result on the network with three planted modules. The values of model parameters are $\Delta c_1 / c_1 = 2.1$ and $\Delta c_2 / c_2 = 0.075$. In this case, however, the network is in the region where the planted structure is algorithmically undetectable. According to Eq. (19), the prediction error for $q = 1$ is at $E(q = 1) \approx 6.38$. We can thus confirm that the prediction errors are minimized or already saturated at $q = 1$. Therefore, the cross-validation estimates perform reasonably on the labeled SBMs.

7. Summary and Discussion

We introduced the LOOCV estimates of the prediction errors for the labeled SBM. While the cross-validation is computationally demanding when it is implemented straightforwardly, we showed that it can be performed efficiently using BP. Our analytical expressions of the LOOCV estimates for the labeled SBM are extensions of the results for the standard (binary label) SBM [6].

Compared to the case of the standard SBM, the cross-validation estimate of the labeled SBM is given as the linear combination of the contribution from each edge type. We also analyzed the performance of the Bayes and Gibbs prediction errors near the detectability threshold. As in the case of the standard SBM, we confirmed that the Gibbs prediction error strictly underfits, whereas the Bayes prediction error supports partitioning all the way down to the detectability threshold.
Figure 2. The cross-validation estimates of the prediction errors on the labeled SBM with two types of edges. The average degrees are $c_1 = 3$ and $c_2 = 5$. In both figures, the horizontal axis indicates the input value for the number of modules $q$, and the vertical axis indicates the value of the cross-validation estimates. From top to bottom, the curves represent the Gibbs prediction error $E_{Gibbs}$ (green), MAP estimate of the Gibbs prediction error $E_{MAP}$ (yellow), Bayes prediction error $E_{Bayes}$ (red), and Gibbs training error $E_{training}$ (blue). The shadow on each curve represents the standard error. In (a), the network has 1,000 vertices of five equally sized planted modules, and the model parameters are in the detectable region. On the other hand, in (b), the network consists of 900 vertices with three equally sized planted modules, and the model parameters are in the algorithmically undetectable region. For each network, the learned affinity matrices are visualized at the bottom. The block in darker color represents the densely connected components in the affinity matrix, whereas the block in the lighter color represents the sparsely connected component. The size of each module is reflected in the size of the corresponding element in the matrix.

In Ref. [6], we derived that the inequality $E_{training} \leq E_{Bayes} \leq E_{Gibbs}$ holds in general. Moreover, when the partitions with different $q$ constitute a hierarchical structure, we can also show the inequality $q^*_G \leq q^*_B$, where $q^*_B$ and $q^*_G$ are the estimated number of modules by $E_{Bayes}$ and $E_{Gibbs}$, respectively. Because the derivations of these inequalities are independent of the edge type, those relations among the cross-validation errors also hold in the present case.

Acknowledgments
This work was supported by the New Energy and Industrial Technology Development Organization (NEDO).

References
[1] Holland P W, Laskey K B, and Leinhardt S 1983, Soc.Netw. 5, 109
[2] Bishop C M 2006, Pattern Recognition and Machine Learning, (Singapore: Springer)
[3] Decelle A, Krzakala F, Moore C, and Zdeborová L 2011, Phys. Rev. E 84 066106
[4] Yedidia J S, Freeman W T, and Weiss Y 2001, Advances in neural information processing systems 13
[5] Mézard M and Montanari A 2009, *Information, Physics, and Computation* (Oxford: Oxford University Press)
[6] Kawamoto T and Kabashima Y 2017, *Sci. Rep.* 7, 3327
[7] Arlot S and Celisse A 2010, *Statistics surveys*, **4**, 40-79
[8] Friedman J, Hastie T, and Tishirani R 2001, *The elements of statistical learning* (New York: Springer series in statistics)
[9] Heimlicher S, Lelarge M, and Massoulié L 2012, *arXiv preprint* arXiv:1209.2910
[10] Kawamoto T 2017, *arXiv preprint* arXiv:1710.08816
[11] Kawamoto T 2017, *arXiv preprint* arXiv:1710.08841