Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

On the Scalable Learning of Stochastic Blockmodel

Bo Yang and **Xuehua Zhao**
School of Computer Science and Technology, Jilin University, Changchun, China
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, China
ybo@jlu.edu.cn

Abstract

Stochastic blockmodel (SBM) enables us to decompose and analyze an exploratory network without a priori knowledge about its intrinsic structure. However, the task of effectively and efficiently learning a SBM from a large-scale network is still challenging due to the high computational cost of its model selection and parameter estimation. To address this issue, we present a novel SBM learning algorithm referred to as BLOS (BLOckwise Sbm learning). Distinct from the literature, the model selection and parameter estimation of SBM are concurrently, rather than alternately, executed in BLOS by embedding the minimum message length criterion into a block-wise EM algorithm, which greatly reduces the time complexity of SBM learning without losing learning accuracy and modeling flexibility. Its effectiveness and efficiency have been tested through rigorous comparisons with the state-of-the-art methods on both synthetic and real-world networks.

Introduction

Formally, a standard SBM is defined as a triple (K, Π, Ω). K is the number of blocks. Π is a $K \times K$ matrix, in which π_{ql} denotes the probability that a link from one node in block q connects to another node in block l. Ω is a K-dimension vector, in which ω_k denotes the probability that a randomly chosen node falls in block k.

SBM is often used as a generative model to decompose real-world networks or synthesize artificial networks, which contain either assortative communities, disassortative multipartites, or arbitrary mixtures of them. Moreover, SBM can be used as a prediction model for link prediction. Being a powerful tool of network analysis, SBM has attracted more and more attentions (Newman and Leicht 2007; Airoldi et al. 2009; Latouche et al. 2011; Karrer and Newman 2011; Yang et al. 2011; Yang, Liu, and Liu 2012) since it was originally proposed (Holland and Leinhardt 1981).

Although SBM has superiority in structure analysis, however, SBM learning is computationally intractable, which limits it to a narrow range of applications just involving very small networks. For the current algorithms, given the number of blocks K, i.e. we do not consider model selection, the time of learning is at least $O(K^2n^2)$. Otherwise, it quickly goes up to $O(n^5)$ in that the process of determining "true" block numbers is very time-consuming. In another word, if we use a conventional PC to run current algorithms, the networks we can efficiently handle contain at most hundreds of nodes, far from the scales faced in practice.

The learning of SBM consists of two main sub-tasks: to determine block number K and to estimate parameter Π and Ω, corresponding to model selection and parameter estimation, respectively. Model selection aims at selecting a model having a good tradeoff between data fitting and model complexity, in obtaining a better generalization ability. Since the tradeoff can be measured by the quantity of its parameters, to select a "good" model for SBM means to determine a reasonable value of K in the sense that the parameter number of SBM is actually a function of K. For example, the parameter number of a standard SBM is equal to $K^2 + K + 1$. Formally, for a given network N, the objective of SBM learning can be stated as: $\arg\min_{K,h} C(N, K, h)$, where h denotes model parameters (i.e. Π and Ω), C denotes the cost function evaluating the tradeoff of parameterized model (K, h). A widely used cost function is: $C(N, K, h) = -\log L(N|K, h) + p(K, h)$, where $\log L(N|K, h)$ indicates the data fitting in terms of the maximum log-likelihood of N given a model and its parameters, and $p(K, h)$ is a regularization item that penalizes models with high complexity.

In the literature, MCMC (Snijders and Nowicki 1997; Yang et al. 2011; McDaid et al. 2013), EM (Newman and Leicht 2007), variational EM (Latouche, Birmele, and Ambroise 2012), and variational Bayes EM (Airoldi et al. 2009; Latouche, Birmele, and Ambroise 2012; Gopalan et al. 2012) have been adopted to estimate the parameters of SBM. Currently, the model selection methods used by SBM learning are either cross validation (Airoldi et al. 2009), or MDL (Yang, Liu, and Liu 2012), or different approximations of Bayesian model evidence, mainly including BIC (Airoldi et al. 2009), ICL(Daudin, Picard, and Robin 2008), and Variation based approximate evidence (Hofman and Wiggins 2008; Latouche, Birmele, and Ambroise 2012).

Current SBM learning algorithms adopt a model-wise learning mechanism to integrate the aforementioned methods of parameter estimation and model selection. That is, they parameterize and then evaluate all candidates in a model space one by one. Finally, the parameterized model

with the best evaluation is selected. Let $[K_{min}, K_{max}]$ denote a model space, the pseudo codes of model-wise learning mechanism can be described as follows:

FOR $K = K_{min} : K_{max} : 1$
  estimate h for a given K;
  compute $C(N, K, h)$;
$(K, h)^* = \arg\min_{K,h} C(N, K, h)$.

For an exploratory network we usually have no idea about its true block number, hence a complete model space $[1, n]$ should be exhaustively searched in order to safely find out it. As a result, an extremely expensive computational cost will be resulted by such a model-wise learning. For example, if h is estimated by an EM-like algorithm, such as SILvb (Latouche, Birmele, and Ambroise 2012), the entire time of model-wise learning will be $O(n^5)$.

So far, how to significantly improve the scalability of SBM learning while retaining its learning accuracy and modeling flexibility, in order to properly handle large-scale exploratory networks, is still an open problem. In this work, we will address this problem from two new perspectives, and accordingly our main contributions are twofold.

(1) To reduce the time complexity of parameter estimation by presenting a new SBM model.

Note that, if one adopts EM-like algorithms to estimate parameters, the calculation of Π is the most expensive and dominates the entire time of parameter estimation. In view of this, an indirect rather than direct way is suggested to perform the calculation of Π. In doing so, we first present a new SBM model, referred to as fine-gained SBM (fg-SBM for short), in which Π, a block-to-block connection matrix, is replaced with Θ, a newly introduced block-to-node connection matrix, so that Θ is readily calculated with a much fewer time while ensuring Π can be exactly represented in terms of Θ together with other parameters. In this way, it is expected to reduce the time of parameter estimation while preserving the flexibility of block modeling. It is also important that, the posterior distribution of Z (the laten variable of SBM) can be analytically derived from Θ, and thereby one can directly calculate it by an exact EM instead of estimating an approximate posterior via variational techniques. In what follows, one will see how this feature enables us to derive a much more efficient mechanism to learn SBM.

(2) To reduce the time complexity of model selection by presenting a block-wise learning mechanism.

As mentioned above, current SBM learning algorithms adopt a model-wise learning mechanism to integrate parameter estimation and model selection, in which the processes of parameterizing and evaluating respective candidate models are completely independent of each other. Accordingly, much of the information that could be shared with each other has to be recalculated for each candidate, leading to a very high computational cost. In view of this, we propose a bock-wise learning algorithm named as BLOS (BLock-wise Sbm learning) to efficiently learn the proposed fg-SBM. Instead of the "serial" learning mechanism adopted by current SBMs, the proposed BLOS ingeniously integrates the minimum message length (MML) criterion into a block-wise EM algorithm to achieve a "parallel" learning process, in which the model selection and parameter estimation are executed concurrently in the scale of blocks. In this way, it is expected to greatly reduce the time complexity of SBM learning while preserving its accuracy, which enables BLOS to efficiently and effectively handel much larger networks. To the best of our knowledge, this is the first effort in the literature to propose a block-wise SBM learning algorithm.

Model and Method

The reparameterization of stochastic blockmodel

Let $A_{n\times n}$ be the adjacency matrix of a binary network N containing n nodes. The fine-gained SBM (fg-SBM for short) is defined as a triple X=(K, Θ, Ω). K is the number of blocks. Θ is a $K \times n$ block-node coupling matrix, in which θ_{kj} depicts the probability of a node from block k connecting to node j. Ω is still the prior of block assignment. In addition, from N one can deduce a latent block indicator Z, a $n \times K$ matrix, indicating the relationship between node and block assignment. $z_{ik} = 1$ if node i is assigned to block k, otherwise $z_{ik} = 0$. It is easy to proof, in terms of the reparameterized Θ, the block-block coupling matrix Π in the standard SBM can be represented as $\Pi = \Theta Z D^{-1}$, where D=block-diag$\{n\omega_1^{-1}, \cdots, n\omega_K^{-1}\}$.

According to fg-SBM, one can generate a synthetic net with a block structure by: 1) assigning a node to block k according to ω_k; 2) generating a link from node i to j according to the Bernoulli distribution with a parameter θ_{kj}, where k indicates the block to which node i belongs. Accordingly, the log-likelihood of a network to be generated is:

$$\log p(N|X) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} (\prod_{j=1}^{n} f(\theta_{kj}, a_{ij}))\omega_k \qquad (1)$$

where $f(x, y) = x^y(1-x)^{(1-y)}$ is a Bernoulli distribution.

Considering Z as a latent variable, then the log-likelihood of complete data given a fg-SBM is:

$$\log p(N,Z|X) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}(\sum_{j=1}^{n} \log f(\theta_{kj}, a_{ij}) + \log\omega_k) \qquad (2)$$

A block-wise SBM learning algorithm

In contrast to the model-wise mechanism adopted by current SBM learning, we provide a block-wise learning mechanism to concurrently perform parameter estimation and model selection, described as follows:

Initialize block candidate set: $B = \{b_1, \cdots, b_{K_{max}}\}$;
REPEAT
  FOR $\forall b \in B$ DO
    evaluate block b;
    IF b is good enough
      parameterize b ;
    ELSE
      $B \leftarrow B - \{b\}$;
  compute $C(N, B, h)$, the cost of current model;
UNTIL C is convergent or $\|B\| < K_{min}$;

In the framework, candidates in the scale of blocks, rather than in the scale of full models, are parameterized and evaluated in turn. The processes of handling respective candidate

blocks are dependent. The information obtained from the parameterization and evaluation of one block can be instantly used for handling next blocks, which will avoid a great deal of duplicated calculations in the whole process of learning. Moreover, during the course of block-wise learning, only the blocks that are evaluated as good enough will be further considered to estimate their parameters. Otherwise, they will be removed from candidate set and not considered anymore.

To implement the framework, we integrate MML into a block-wise EM algorithm to evaluate and parameterize each block, respectively. We choose MML as an evaluation criterion mainly because MML sufficiently considers the prior of models, and more importantly as we can see next, such a prior enables MML to be readily integrated into the above block-wise learning framework.

The derivation of cost function Given N, we expect to select an optimal X from its model space to properly fit and to precisely predict the behaviors of the network. According to the MAP principle (maximum a posteriori), the optimal X given network N will be the one with the maximum posterior probability. Moreover, we have: $p(X|N) \propto p(N|X)p(X)$, where $p(X|N)$, $p(N|X)$ and $p(X)$ denote the posteriori of X given N, the likelihood of N given X, and the prior of X, respectively. Next, we will derive the form of $\log p(X|N)$, i.e. the cost function $C(N, X)$, from an integration of MML, standard SBM and fg-SBM.

MML selects models by minimizing the code-length of both data and model. Formally, the cost function of MML is (Lanterman 2001; Figueiredo and Jain 2002):

$$\begin{aligned} C(N,X) = &-\log p(N|X) - \log p(X) \\ &+ \frac{1}{2}\log|\mathbf{I}(X)| + \frac{d}{2}(1+\log\kappa_d) \end{aligned} \tag{3}$$

where d is dimension of X (i.e. the number of parameters of X), $\mathbf{I}(X) \equiv -E[D_X^2 \log p(N|X)]$ is the Fisher information matrix and $|\mathbf{I}(X)|$ denotes its determinant, and κ_d approaches $(2\pi e)^{-1}$ as d grows.

We start our derivation from a standard SBM, denoted as $X_S = (K, \Pi, \Omega)$. Since it is not easy to analytically get $\mathbf{I}(X_S)$, we turn to the Fisher information matrix of complete data likelihood, $\mathbf{I}_c(X_S) \equiv -E[D_{X_S}^2 \log p(N, Z|X_S)]$, which is the upper-bound of $\mathbf{I}(X_S)$ (Titterington et al. 1985). The log-likelihood of complete data given a X_S is:

$$\begin{aligned} \log p(N,Z|X_S) = &\sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\log\omega_k \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{q=1}^{K}\sum_{l=1}^{K} z_{iq}z_{il}\log\pi_{ql}^{a_{ij}}(1-\pi_{ql})^{1-a_{ij}} \end{aligned}$$

From the log-likelihood, $\mathbf{I}_c(X_S)$ is derived as:

$$\begin{aligned} \mathbf{I}_c(X_S) = \text{block-diag}\{&n\omega_1^{-1},\ldots,n\omega_K^{-1}, \\ &\frac{n^2\omega_1\omega_1}{\pi_{11}(1-\pi_{11})},\ldots,\frac{n^2\omega_1\omega_K}{\pi_{1K}(1-\pi_{1K})},\ldots, \\ &\frac{n^2\omega_K\omega_1}{\pi_{K1}(1-\pi_{K1})},\ldots,\frac{n^2\omega_K\omega_K}{\pi_{KK}(1-\pi_{KK})}\} \end{aligned}$$

Accordingly, we have:

$$|\mathbf{I}_c(X_S)| = n^{2K^2+K}\prod_{k=1}^{K}\omega_k^{-1}\prod_{q=1}^{K}\prod_{l=1}^{K}\frac{\omega_q\omega_l}{\pi_{ql}(1-\pi_{ql})} \tag{4}$$

We use a noninformative prior to depict the lack of knowledge about model parameters, in which the prior of Ω and Π are independent and the priori of respective π_{ql} are also independent. Specifically, we have: $p(X_S) = p(\omega_1,...,\omega_k)\prod_{q=1}^{K}\prod_{l=1}^{K}p(\pi_{ql})$, $p(\omega_1,...,\omega_k) \propto \sqrt{|\mathbf{I}(\Omega)|} = (\prod_{k=1}^{K}\omega_k)^{-\frac{1}{2}}$ and $p(\pi_{ql}) \propto \sqrt{|\mathbf{I}(\pi_{ql})|} = (\pi_{ql}(1-\pi_{ql}))^{-\frac{1}{2}}$.

Based on above analysis, overall we have:

$$\begin{aligned} C(N,X_S) = &-\log p(N|X_S) + \frac{1}{2}\sum_{q=1}^{K}\sum_{l=1}^{K}\log\omega_q\omega_l \\ &+ \frac{2K^2+K}{2}\log n + \frac{K^2+K}{2}(1+\log\kappa_d) \end{aligned} \tag{5}$$

Now let us connect two SBMs, i.e., X_S and X. Note that: 1) Π can be represented as $\Theta Z D^{-1}$; and 2) Z is independent on Π and Θ given K and Ω, respectively. So, we have:

$$\begin{aligned} \log p(N|X_S) &= \log\textstyle\sum_Z p(N,Z|K,\Pi,\Omega) \\ &= \log\textstyle\sum_Z p(N|Z,K,\Pi,\Omega)p(Z|K,\Omega) \\ &= \log\textstyle\sum_Z p(N|Z,K,\Theta ZD^{-1},\Omega)p(Z|K,\Omega) \\ &= \log\textstyle\sum_Z p(N,Z|K,\Theta,\Omega) \\ &= \log p(N|K,\Theta,\Omega) = \log p(N|X). \end{aligned}$$

In addition, we have: 1) K and Ω in X_S and X are the same, and 2) the parameters of zero-probability block (i.e. $\omega_k = 0$) will not make any contribution to total code-length. Let $K_g \leq K$ be the number of greater-than-zero probability blocks, then Eq. 5 becomes:

$$\begin{aligned} C(N,X) = &-\log p(N|X) + \frac{1}{2}\sum_{\omega_q>0}\sum_{\omega_l>0}\log\omega_q\omega_l \\ &+ \frac{2K_g^2+K_g}{2}\log n + \frac{K_g^2+K_g}{2}(1+\log\kappa_d) \end{aligned} \tag{6}$$

Optimization method According to information theory, the cost in terms of Eq. 6 is the sum of code-length of data, denoted by the minus likelihood $-\log p(N|X)$, and code-length of model, denoted by the remaining part. While, from Bayesian point, the minus of Eq. 6 can be regarded as the posteriori of X, $\log p(X|N)$, which is the sum of a log-likelihood $\log p(N|X)$ and a priori $-\frac{1}{2}\sum_q\sum_l\log\omega_q\omega_l - \frac{2K_g^2+K_g}{2}\log n - \frac{K_g^2+K_g}{2}(1+\log\kappa_d)$. It means to minimize Eq. 6 is to maximize the posteriori. Next, we use a standard EM to estimate an optimal X by maximizing $\log p(X|N)$. Its E-step and M-step are designed respectively as follows.

E-step: Given N, K, and $h^{(t-1)}$, where h and t respectively denote the parameters (Θ, Ω) and the current iteration, to compute the conditional expectation of complete log-likelihood, i.e., the Q-function.

$$Q(h,h^{(t-1)}) = \sum_{i=1}^{n}\sum_{k=1}^{K}\gamma_{ik}(\sum_{j=1}^{n}\log f(\theta_{kj},a_{ij}) + \log\omega_k) \tag{7}$$

where $\gamma_{ik} = E[z_{ik}; h^{(t-1)}]$ denotes the posteriori probability of node i belonging to block k given $h^{(t-1)}$. We have:

$$\gamma_{ik} = \frac{\omega_k^{(t-1)} \prod_{j=1}^{n} f(\theta_{kj}^{(t-1)}, a_{ij})}{\sum_{k=1}^{K} \omega_k^{(t-1)} \prod_{j=1}^{n} f(\theta_{kj}^{(t-1)}, a_{ij})} \tag{8}$$

M-step: To maximize $Q(h, h^{(t-1)}) + \log p(h)$, where $\log p(h) = -\frac{1}{2} \sum_{\omega_q>0} \sum_{\omega_l>0} \log \omega_q \omega_l - \frac{2K_g^2+K_g}{2} \log n - \frac{K_g^2+K_g}{2}(1+\log \kappa_d)$. By solving this optimization with a constraint $\sum_{k=1}^{K} \omega_k = 1$, we have:

$$\begin{cases} \omega_k^{(t)} = \frac{\max\{0, \sum_{i=1}^{n} \gamma_{ik} - K_g\}}{\sum_{j=1}^{K} \max\{0, \sum_{i=1}^{n} \gamma_{ij} - K_g\}} \\ \theta_{kj}^{(t)} = \frac{\sum_{i=1}^{n} a_{ij} \gamma_{ik}}{\sum_{i=1}^{n} \gamma_{ik}} \end{cases} \tag{9}$$

Note that, the parameter Π of standard SBM can also be iteratively computed in terms of γ, as follows:

$$\pi_{pl} = \frac{\sum_i \sum_j \gamma_{ip} \gamma_{jl} a_{ij}}{\sum_i \sum_j \gamma_{ip} \gamma_{jl}} \tag{10}$$

It is easy to verify, the complexity of calculating Θ of fg-SBM according to Eq.9 is $O(Kn^2)$, yet the time of calculating Π according to Eq.10 is $O(K^2n^2)$.

Since the prior of block assignment Ω characterizes the normalized distribution of block size, the calculation of ω_k in Eq.9 partially reflect the process of block-wise model selection, in which blocks being not sufficiently supported by data will be annihilated timely. More specifically, for each individual block k, ω_k will become and thereafter keep zero if its expectation size at present, i.e. $\sum_{i=1}^{n} \gamma_{ik}$, is less than the number of existing blocks.

If one considers such a model selection as a voting game, Eq.9 actually implies a new mechanism design particularly for SBM learning according to MML, in which candidates will be disqualified and then timely excluded from the current playoff of the game if the votes they have won from all nodes are less than the total number of existing candidate blocks. Note that, the threshold for qualifying individual blocks, i.e. K_g, is not fixed but self-adjusted during whole learning process. That is to say, the regulations of threshold at different stages will be self-adaptive to the block parameterization (in terms of the calculation of Θ and Γ) and block evaluation (in terms of the calculation of Ω) of both previous and current playoffs. The self-adaption of evaluating criterion is one of main features of block-wise SBM learning. In addition, the criterions at different stages will be evolving from strict to loose with the gradual reduction of candidates during playoffs, implying many trivial blocks will be removed as early as possible and thereby considerable computational cost of corresponding parameterization will be saved in this way.

The mechanism of block-wise SBM learning Based on the above analysis, Table 1 summarizes the detailed mechanism of block-wise SBM learning. Corresponding to the aforementioned framework, the evaluation, selection, parameterization and annihilation of blocks are performed in a block-wise mode within a FOR-loop.

Table 1: The implementation of block-wise SBM learning

Algorithm BLOS

Input: N, K_{min}, K_{max}
Output: X and Z

01 Initial: $B = \{b_1, ..., b_{K_{max}}\}$; $t \leftarrow 0$; $K_g \leftarrow K_{max}$; ε; $\Theta^{(0)}$;
02 $\Omega^{(0)}$; $u_{ik}^{(0)} \leftarrow \prod_{j=1}^{n} f(\theta_{kj}^{(0)}, a_{ij})$, for $i = 1, ..., n$ and $\forall b_k \in B$;
03 REPEAT
04 $t \leftarrow t + 1$;
05 FOR $\forall b_k \in B$ DO
06 $\gamma_{ik}^{(t)} \leftarrow \frac{\omega_k^{(t-1)} u_{ik}^{(t-1)}}{\sum_{b_j \in B} \omega_j^{(t-1)} u_{ij}^{(t-1)}}$, for $i = 1, ..., n$;
07 $\omega_k^{(t)} \leftarrow \frac{\max\{0, \sum_{i=1}^{n} \gamma_{ik}^{(t)} - K_g\}}{\sum_{b_j \in B} \max\{0, \sum_{i=1}^{n} \gamma_{ij}^{(t)} - K_g\}}$;
08 $S \leftarrow \sum_{b_j \in B} \omega_j^{(t)}$;
09 $\omega_j^{(t)} \leftarrow \omega_j^{(t)} S^{-1}, \forall b_j \in B$;
10 IF $\omega_k^{(t)} > 0$ THEN
11 $\theta_{ki}^{(t)} \leftarrow \frac{\sum_{i=1}^{n} a_{ij} \gamma_{ik}^{(t)}}{\sum_{i=1}^{n} \gamma_{ik}^{(t)}}$, for $i = 1, ..., n$;
12 $u_{ik}^{(t)} \leftarrow \prod_{j=1}^{n} f(\theta_{kj}^{(t)}, a_{ij})$, for $i = 1, ..., n$;
13 ELSE
14 $K_g \leftarrow K_g - 1$;
15 $B \leftarrow B - \{b_k\}$;
16 ENDIF
17 ENDFOR
18 $X^{(t)} \leftarrow \{K_g, \Theta^{(t)}, \Omega^{(t)}\}$;
19 compute $C(N, X^{(t)})$ by Eq. 6;
20 UNTIL $|C(N, X^{(t-1)}) - C(N, X^{(t)})| < \varepsilon$ or $K_g < K_{min}$;
21 $X \leftarrow X^{(t)}$;

Time complexity analysis The nested REPEAT and FOR loops are the most time-consuming in BLOS, which dominate the whole time of learning. In the body of FOR-loop, it takes $O(nK_g)$ time to calculate $\gamma_{\cdot k}$ in line 06 and ω_k in line 07, respectively, and takes $O(n^2)$ time to calculate $\theta_{k\cdot}$ in line 11 and $u_{\cdot k}$ in line 12, respectively. Accordingly, the FOR-loop takes $O(nK_g^{(t)}K_g^{(t)} + n^2K_g^{(t+1)})$ time, where $K_g^{(t)}$ denotes the size of set B at the t-th iteration of REPEAT-loop. Cost computation in line 19 takes $O(n^2K_g^{(t+1)} + (K_g^{(t+1)})^2)$ time. So, it will take $O(nK_g^{(t)}K_g^{(t)} + n^2K_g^{(t+1)} + (K_g^{(t+1)})^2) < O(n^2K_g^{(t)})$ time to perform the t-th REPEAT-loop. Thus, the complexity of REPEAT-loop is $O(\sum_{t=1}^{T} n^2K_g^{(t)})$, where T is number of total iterations. Note that, the initialization of all $u_{ik}^{(0)}$ takes $O(n^2K_{max})$ time, so the total time complexity of BLOS is $O(\sum_{t=1}^{T} n^2K_g^{(t)} + n^2K_{max})$. Since $K_g^{(t)} \leq K_{max}$, in the worst case, the time of BLOS is bounded by $O(Tn^2K_{max})$. If the real number of blocks (say K) is known, the worst time of BLOS is $O(Tn^2K)$ by initializing $K_{max} = K$. Otherwise, it will be $O(Tn^3)$ by initializing $K_{max} = O(n)$.

Validations

Next, we design experiments oriented toward evaluating the accuracy, the scalability, and the tradeoff between accuracy and scalability of BLOS. In order to sufficiently demonstrate the superiority of BLOS, Four state-of-the-art SBM learning methods, VBMOD (Hofman and Wiggins 2008), GSMDL

(Yang, Liu, and Liu 2012), SICL (Daudin, Picard, and Robin 2008) and SILvb (Latouche, Birmele, and Ambroise 2012), are selected as comparative methods, whose rationale and time complexity are summarized in Table 2. All experiments are performed on a conventional personal computer with a 2GH CPU and a 4GB RAM.

Table 2: Time complexity of SBM learning algorithms

Algorithm	Parameter estimation	Model selection	Learning strategy	K	N/K
BLOS	BEM	MML	B	$O(TKn^2)$	$O(Tn^3)$
VBMOD	VBEM	VAE	M	$O(TKn^2)$	$O(Tn^4)$
GSMDL	EM	MDL	M	$O(TK^2n^2)$	$O(Tn^5)$
SICL	VEM	ICL	M	$O(TK^2n^2)$	$O(Tn^5)$
SILvb	VBEM	VAE	M	$O(TK^2n^2)$	$O(Tn^5)$

In Table 2, K is the real number of blocks and T is the required iterations. "BEM", "VBEM", "VEM" mean "block-wise EM", "variational Bayes EM", "variational EM", respectively. "B" and "M" mean "block-wise" and "model-wise", respectively. "VAE" is the abbreviation of variational based approximate evidence. "K" and "N/K" indicate "K is known " and "K is unknown", respectively. In both cases, we list the worst time complexity of respective algorithms.

Validation on accuracy We first generate three types of synthetic networks according to specific SBMs, which respectively contain a community structure, a hub structure, and a hybrid structure of community and multipartite. Each type of networks is further divided into five groups according to the true block number they contain, i.e., $K_{true} = 3$, 4, 5, 6 or 7. Each group has 100 networks and each network contains 50 nodes. The parameters of three types of SBMs are given as follows:

Type I, containing a community structure: $\pi_{ij} = 0.9 \times I(i = j) + 0.1 \times I(i \neq j)$ and $\forall k : \omega_k = 1/K_{true}$.

Type II, containing a hub structure: $\pi_{ij} = 0.9 \times I(i = j$ or $i = 1$ or $j = 1) + 0.1 \times I(i \neq j$ and $i, j \neq 1)$ and $\forall k : \omega_k = 1/K_{true}$.

Type III, containing a hybrid structure of community and multipartite: $\forall k, \omega_k = 1/K_{true}$ and block matrix Π takes following form:

$$\Pi = \begin{bmatrix} p_1 & & \boldsymbol{p}_2 & \cdot & & & \\ & \ddots & & \cdot & & \boldsymbol{p}_2 & \\ \boldsymbol{p}_2 & & p_1 & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & \cdot & p_2 & & \boldsymbol{p}_1 \\ & \boldsymbol{p}_2 & & \cdot & & \ddots & \\ & & & \cdot & \boldsymbol{p}_1 & & p_2 \end{bmatrix} \begin{matrix} \} k_1 \\ \\ \\ \\ \} k_2 \end{matrix}$$

where k_1 and k_2 denote the number of communities and multipartite components, respectively. We have $k_1 + k_2 = K_{true}$. When K_{true} takes 3, 4, 5, 6 and 7 in turn, k_1 takes 1, 2, 3, 3 and 3, accordingly. In this experiment, we set $p_1 = 0.9$ and $p_2 = 0.1$.

For each type of block structure, we calculate the average NMI(normalized mutual information) over 100 synthetic networks for each algorithm; results are given in Table 3, in which the numbers in brackets on the right hand side show the ranks in terms of decreasing average NMI, indicating the accuracy rank of tested algorithms on average. For networks of Type I, SILvb and VBMOD perform slightly better than other algorithms. For networks of Type II and III, SILvb and BLOS perform better than others. VBMOD is stable for community detection, but it fails to handle networks containing beyond community structures.

Validation on computational scalability Next, we use synthetic networks with various scales to test the computing cost of different algorithms. Here, synthetic networks to be used are also generated according to the SBM of Type III. Specifically, its parameters are: $K_{true} = 8$, $k_1 = 3$, $k_2 = 5$ and $\forall k : \omega_k = 0.125$. n alternatively takes 200, 400, 600, 800, 1000, 2000, 5000, 10000, and 15000. In the case of $n < 5000$, we set $p_1 = 0.5$ and $p_2 = 0.01$. Otherwise, we set $p_1 = 0.1$ and $p_2 = 0.0001$. Correspondingly, we generate nine groups of networks with different scales and each group contains 50 randomly generated networks.

For all five algorithms, we set the same model space to search and the same convergence threshold, i.e., $K_{min} = 1$, $K_{max} = 20$ and $\varepsilon = 10^{-4}$. Fig. 1 shows the actual running time on average of five algorithms. Table 4 shows the NMI of detected block structures for these networks, in which "$-$" denotes "not available due to out of memory".

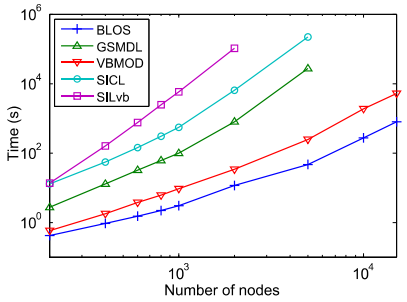

Figure 1: Running time in terms of network scale.

Overall, we have following observations: 1) BLOS runs the most efficient and its actual running time is significantly fewer than its competitors. 2) It is computationally intractable for model-wise methods such as SILvb and SICL to process large networks. Note that, SILvb needs to take 5834 seconds to handle 1000 nodes, and the time will sharply increase to 100,788 seconds (28 hours) when handling 2000 nodes. Comparatively, BLOS runs much faster and is able to handle 2000 nodes within 12 seconds, gaining a 8400-fold speedup of SILvb. SICL spends 216,893 seconds (60 hours) to handle 5000 nodes; BLOS only takes 46 seconds to handle the same network, gaining a 4700-fold speedup. 3) VBMOD adopts a model-wise scheme to learn SBM as well, while it runs much faster than SILvb, SICL and GSMDL. But VBMOD achieves its scalability by greatly simplifying SBM to be learned, i.e. compressing original $K \times K$ matrix Π into two scalar variables, at the price of losing the flexibility of modeling heterogeneous structures. 4) From Fig. 1 and Table 4, one can observe the best tradeoff between accuracy

Table 3: Accuracy of detected block structures in three types of networks

Methods	Networks of Type I						Networks of Type II						Networks of Type III					
	K_{true}						K_{true}						K_{true}					
	3	4	5	6	7	Average	3	4	5	6	7	Average	3	4	5	6	7	Average
BLOS	1	1	1	0.951	0.877	0.966(3)	0.997	1	1	0.950	0.868	0.963(2)	1	1	1	0.978	0.878	0.971(2)
GSMDL	0.998	1	1	0.894	0.783	0.935(5)	0.985	0.994	1	0.889	0.788	0.931(4)	0.989	1	1	0.946	0.851	0.957(4)
VBMOD	1	1	1	1	0.861	0.972(2)	0.592	0.771	0.851	0.850	0.837	0.780(5)	0.764	0.863	0.742	0.811	0.780	0.792(5)
SICL	1	1	1	0.940	0.837	0.955(4)	1	1	1	0.944	0.855	0.960(3)	1	1	1	0.981	0.850	0.966(3)
SILvb	1	1	1	0.999	0.947	0.989(1)	1	1	1	0.999	0.951	0.990(1)	1	1	1	1	0.941	0.988(1)

and scalability demonstrated by BLOS. That is, compared with state-of-the-art algorithms, BLOS is able to effectively and efficiently handle much larger networks while preserving rather good learning precision.

Table 4: NMI of detections by five algorithms

Number of nodes	BLOS	GSMDL	VBMOD	SICL	SILvb
200	1	0.996	0.890	1	1
400	1	0.989	0.890	1	1
600	1	0.937	0.890	1	1
800	1	0.933	0.890	1	1
1000	1	0.924	0.890	1	1
2000	1	0.913	0.890	1	1
5000	1	0.890	0.890	1	–
10000	0.955	–	0.890	–	–
15000	0.940	–	0.890	–	–

Validation on real-world networks Now we test the performance of algorithms with real-world networks. Total 9 real-world networks are selected, which are widely used as benchmarks to validate the performance of block structure detection or scalability. The structural features of these networks are summarized in Table 5. Some have ground truth block structures. "–" means ground truth is not available.

Table 5: Structural features of 12 real-world networks

Network	Type	# of node	# of edge	Clustering coefficient	Average degree	Structure
Karate	Undirected	34	78	0.57	4.59	community
Dolphins	Undirected	62	159	0.26	5.13	community
Foodweb	Undirected	75	113	0.33	3.01	hybrid
Polbooks	Undirected	105	441	0.49	8.40	community
Adjnoun	Undirected	112	425	0.17	7.59	bipartite
Football	Undirected	115	613	0.40	10.7	community
Email	Undirected	1133	5451	0.22	9.62	–
Polblogs	Directed	1222	16714	0.32	27.4	–
Yeast	Undirected	2224	6609	0.13	5.94	–

For each algorithm, we fix $K_{min} = 1$ and set K_{max} according to Table 6. One can see the running time of BLOS is significantly lower than others, particularly for larger networks. For networks having ground truth, the true block numbers are listed below "K_{true}" in Table 7, and the detected numbers by algorithms are listed behind "/". We adopt NMI to measure the distance between ground truth and detections of algorithms. The last line gives the ranks of respective algorithms in terms of average NMI. BLOS performs the best when processing such real-world networks.

Table 6: Actual running time in real-world networks (s)

Networks	K_{max}	BLOS	GSMDL	VBMOD	SICL	SILvb
Karate	$n/2$	**0.13**	0.23	0.19	0.34	0.42
Dolphins	$n/2$	**0.32**	1.55	0.45	4.08	2.03
Foodweb	$n/2$	**0.33**	1.60	0.53	4.92	2.31
Polbooks	$n/2$	**0.96**	8.06	2.02	36.80	12.76
Adjnoun	$n/2$	**1.10**	8.09	2.12	40.08	16.34
Football	$n/2$	**1.20**	8.11	2.20	42.14	18.11
Email	100	**41.09**	18575	389	35288	78597
Polblogs	100	**43.82**	26031	618	44834	106034
Yeast	100	**104**	>48h	1677	>48h	>48h

Table 7: NMI of detections by five algorithms

Networks	K_{true}	BLOS	GSMDL	VBMOD	SICL	SILvb
Karate	2	**0.839**/3	0.754/4	0.837/2	0.792/4	0.770/4
Dolphins	2	**0.660**/3	0.551/4	0.628/4	0.368/3	0.387/3
Foodweb	5	**0.269**/4	0.185/5	0.023/2	0.199/2	0.201/2
Polbooks	3	**0.585**/4	0.469/6	0.512/6	0.458/5	0.455/5
Adjnoun	2	**0.206**/5	0.193/8	0.020/5	0.040/3	0.046/3
Football	12	0.884/10	0.824/10	0.862/9	**0.910**/10	**0.910**/10
avg(rank)		**0.574**(1)	0.496(2)	0.480(3)	0.461(5)	0.461(4)

Conclusion

Current SBMs face two main difficulties, which jointly make their learning processes not scalable. (1) Some parameters like Π cannot be estimated in an efficient way; (2) the posterior of Z cannot be explicitly derived due to the dependency of its components. Therefore, one has to assume an approximate distribution of Z and then turn to variational techniques. While, it is difficult to integrate variational methods with current model evaluation criteria to analytically derive a block-wise learning mechanism, enabling to perform parameter estimation and model selection concurrently. In view of this, we raised a reparameterized SBM and then theoretically derived a bock-wise learning algorithm, in which parameter estimation and model selection are executed concurrently in the scale of blocks. Validations show that BLOS achieves the best tradeoff between effectiveness and efficiency. Particularly, compared to SILvb, a recently proposed method with an excellent learning accuracy, BLOS achieves a n^2-fold speedup, reducing learning time from $O(n^5)$ to $O(n^3)$, while preserving competitive enough learning accuracy.

Acknowledgements

This work was funded by the Program for New Century Excellent Talents in University under Grant NCET-11-0204,

and the National Science Foundation of China under Grants 61133011, 61373053, and 61300146.

References

Airoldi, E. M.; Blei, D. M.; Fienberg, S. E.; and Xing, E. P. 2009. Mixed membership stochastic blockmodels. In *Advances in Neural Information Processing Systems*, 33–40.

Daudin, J.-J.; Picard, F.; and Robin, S. 2008. A mixture model for random graphs. *Statistics and computing* 18(2):173–183.

Figueiredo, M. A., and Jain, A. K. 2002. Unsupervised learning of finite mixture models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24(3):381–396.

Gopalan, P.; Gerrish, S.; Freedman, M.; Blei, D. M.; and Mimno, D. M. 2012. Scalable inference of overlapping communities. In *Advances in Neural Information Processing Systems*, 2249–2257.

Hofman, J. M., and Wiggins, C. H. 2008. Bayesian approach to network modularity. *Physical review letters* 100(25):258701.

Holland, P. W., and Leinhardt, S. 1981. An exponential family of probability distributions for directed graphs. *Journal of the american Statistical association* 76(373):33–50.

Karrer, B., and Newman, M. E. 2011. Stochastic blockmodels and community structure in networks. *Physical Review E* 83(1):016107.

Lanterman, A. D. 2001. Schwarz, wallace, and rissanen: Intertwining themes in theories of model selection. *International Statistical Review* 69(2):185–212.

Latouche, P.; Birmelé, E.; Ambroise, C.; et al. 2011. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics* 5(1):309–336.

Latouche, P.; Birmele, E.; and Ambroise, C. 2012. Variational bayesian inference and complexity control for stochastic block models. *Statistical Modelling* 12(1):93–115.

McDaid, A. F.; Murphy, T. B.; Friel, N.; and Hurley, N. J. 2013. Improved bayesian inference for the stochastic block model with application to large networks. *Computational Statistics & Data Analysis* 60:12–31.

Newman, M. E., and Leicht, E. A. 2007. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences* 104(23):9564–9569.

Snijders, T. A., and Nowicki, K. 1997. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of classification* 14(1):75–100.

Titterington, D. M.; Smith, A. F.; Makov, U. E.; et al. 1985. *Statistical analysis of finite mixture distributions*, volume 7. Wiley New York.

Yang, T.; Chi, Y.; Zhu, S.; Gong, Y.; and Jin, R. 2011. Detecting communities and their evolutions in dynamic social networksa bayesian approach. *Machine learning* 82(2):157–189.

Yang, B.; Liu, J.; and Liu, D. 2012. Characterizing and extracting multiplex patterns in complex networks. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 42(2):469–481.