Mach Learn (2011) 82: 157–189
DOI 10.1007/s10994-010-5214-7

Detecting communities and their evolutions in dynamic social networks—a Bayesian approach

Tianbao Yang · Yun Chi · Shenghuo Zhu · Yihong Gong · Rong Jin

Received: 1 June 2009 / Accepted: 1 May 2010 / Published online: 25 September 2010

Abstract Although a large body of work is devoted to finding communities in static social networks, only a few studies examined the dynamics of communities in evolving social networks. In this paper, we propose a dynamic stochastic block model for finding communities and their evolution in a dynamic social network. The proposed model captures the evolution of communities by explicitly modeling the transition of community memberships for individual nodes in the network. Unlike many existing approaches for modeling social networks that estimate parameters by their most likely values (i.e., point estimation), in this study, we employ a Bayesian treatment for parameter estimation that computes the posterior distributions for all the unknown parameters. This Bayesian treatment allows us to capture the uncertainty in parameter values and therefore is more robust to data noise than point estimation. In addition, an efficient algorithm is developed for Bayesian inference to handle large sparse social networks. Extensive experimental studies based on both synthetic data and real-life data demonstrate that our model achieves higher accuracy and reveals more insights in the data than several state-of-the-art algorithms.

Editors: S.V.N. Vishwanathan, Samuel Kaski, Jennifer Neville, and Stefan Wrobel.

T. Yang (✉) · R. Jin
Department of Computer Science and Engineering Michigan State University, East Lansing, MI 48824, USA
e-mail: yangtia1@msu.edu

R. Jin
e-mail: rongjin@msu.edu

Y. Chi · S. Zhu · Y. Gong
NEC Laboratories America, 10080 N. Wolfe Rd, SW3-350, Cupertino, CA 95014, USA

Y. Chi
e-mail: ychi@sv.nec-labs.com

S. Zhu
e-mail: zsh@sv.nec-labs.com

Y. Gong
e-mail: ygong@sv.nec-labs.com

Keywords Social network · Community · Community evolution · Dynamic stochastic block model · Bayesian inference · Gibbs sampling

1 Introduction

As online social networks such as Facebook[1] and MySpace[2] are gaining popularity rapidly, social networks have become a ubiquitous part of many people's daily lives. Therefore, social network analysis is becoming a more and more important research field. One major topic in social network analysis is the study of communities in social networks. For instance, in Wikipedia,[3] the online social network service is defined as "A social network service focuses on building *online communities* of people who share interests and activities, or who are interested in exploring the interests and activities of others". Analyzing communities in a social network, in addition to serving scientific purposes (e.g., in sociology and social psychology), helps improve user experiences (e.g., through friend recommendation services) and provides business value (e.g., in target advertisement and market segmentation analysis).

Communities have long been studied in various social networks. For example, in social science an important research topic is to identify cohesive subgroups of individuals within a social network where cohesive subgroups are defined as "subsets of actors among whom there are relatively strong, direct, intense, frequent, or positive ties" (Wasserman and Faust 1994). As another example, communities also play an important role in Web analysis, where a Web community is defined as "a set of sites that have more links to members of the community than to non-members" (Flake et al. 2000).

Social networks are usually represented by graphs where nodes represent individuals and edges represent relationships and interactions among individuals. Based on this graph representation, there exists a large body of work on analyzing communities in *static* social networks, ranging from well-established social network analysis (Wasserman and Faust 1994) to recent successful applications such as Web community discovery (Flake et al. 2000). However, these studies overlooked an important feature of communities—communities in real life are usually *dynamic*. On a macroscopic level, community structures evolve over time. For example, a political community whose members' main interest is the presidential election may become less active after the election takes place. On a microscopic level, individuals may change their community memberships, due to the shifts of their interests or due to certain external events. In this respect, the above studies that analyze static communities fail to capture the important dynamics in communities.

Recently, there has been a growing body of work on analyzing *dynamic* communities in social networks. As we will discuss in detail in related work, some of these studies adopt a two-step approach where first static analysis is applied to the snapshots of the social network at different time steps, and then community evolution is introduced afterward to interpret the change of communities over time. Because data in real world are often noisy, such a two-step approach often results in unstable community structures and consequentially, unwarranted community evolution. Some more recent studies attempted to unify the processes of community extraction and evolution extraction by using certain heuristics, such as regularizing *temporal smoothness*. Although some encouraging results are reported, these studies lack

[1] http://www.facebook.com

[2] http://www.myspace.com

[3] http://www.wikipedia.org

rigorous generative models and therefore are usually ad hoc. Furthermore, none of these studies explicitly model the transition or change of community memberships, which is the key to the analysis of dynamic social network. In addition, most existing approaches consider point estimation in their studies, i.e., they only estimate the most likely value for the unknown parameters. Given the large scale of social networks and potential noise in data, it is likely that the network data may not be sufficient to determine the exact value of parameters, and therefore it is important to develop methods beyond point estimation in order to model and capture the uncertainty in parameter estimation.

In this paper, we present a probabilistic framework for analyzing dynamic communities in social networks that explicitly addresses the above two problems. Instead of employing an afterward effect or a regularization term, the proposed approach provides a unified framework for modeling both communities and their evolution simultaneously; the dynamics of communities is modeled explicitly by transition parameters that dictates the changes in community memberships over time; a Bayesian treatment of parameter estimation is employed to avoid the shortcoming of point estimation by using the posterior distributions of parameters for membership prediction. In short, we summarize the contributions of this work as follows.

- We propose a dynamic stochastic block model for modeling communities and their evolution in a unified probabilistic framework. Our framework has two versions, the *online inference* version that progressively updates the probabilistic model over time, and the *offline inference* version that learns the probabilistic model with network data obtained at all time steps in a retrospective way. This is in contrast to most existing studies of social network analysis that only focus on the online inference approaches. We illustrate the advantage of the offline inference approach in our empirical study.
- We present a Bayesian treatment for parameter estimation in the proposed framework. Unlike most existing approaches for social network analysis that only compute the most likely values for the unknown parameters, the Bayesian treatment estimates the posterior distributions for unknown parameters, which is utilized to predict community memberships as well as to derive important characteristics of communities, such as community structures, community evolution, etc.
- We develop a very efficient algorithm for the proposed framework. Our algorithm is executed in an incremental fashion to minimize the computational cost. In addition, our algorithm is designed to fully take advantage of the sparseness of data. We show that for each iteration, our algorithm has a time complexity linear in the size of a social network provided the network is sparse.

We conduct extensive experimental studies on both synthetic data and real data to investigate the performance of our framework. We show that compared to state-of-the-art baseline algorithms, our model is advantageous in (a) achieving better accuracy in community extraction, (b) capturing community evolution more faithfully, and (c) revealing more insights from the network data.

The rest of the paper is organized as follows. In Sect. 2 we discuss related work. In Sect. 3, we present our dynamic stochastic block model for communities and their evolution. In Sect. 4, we provide a point estimation approach to estimate the parameters in our model. In Sect. 5, we propose a Bayesian inference method to learn the parameters in our dynamic stochastic block model. In Sect. 6, we describe some details of our implementation and provide a complexity analysis for our algorithm. In Sect. 7, we discuss several extensions to our basic model. We present experimental studies in Sect. 8 and give conclusion and future directions in Sect. 9.

2 Related work

Finding communities is an important research topic in social network analysis. For the task of community discovery, many approaches such as clique-based, degree-based, and matrix-perturbation-based, have been proposed. Wasserman and Faust (1994) gave a comprehensive survey on these approaches. Community discovery is also related to some important research issues in other fields. For example, in applied physics, communities are important in analyzing modules in a physical system and various algorithms, such as (Newman and Girvan 2004; Newman 2006), have been proposed to discover modular structures in physical systems. As another example, in the machine learning field, finding communities is closely related to graph-based clustering algorithms (Chung 1997), such as the normalized cut algorithm proposed by Shi and Malik (2000), the modularity-based approaches proposed by White and Smyth (2005) and by Chen et al. (2009), and the graph-factorization clustering (GFC) algorithm proposed by Yu et al. (2005). However, all these approaches focus on analyzing *static* networks while our focus in this study is on analyzing *dynamic* social networks.

In the field of statistics, a well-studied probabilistic model is the stochastic block model (SBM). This model was originally proposed by Holland and Leinhardt (1976) and was further extended by others, e.g. (Fienberg et al. 1985; Ho et al. 2002; Shortreed et al. 2006; Snijders 2002; Wasserman and Pattison 1996). The SBM model has been successfully applied in various areas such as bioinformatics and social science (Airoldi et al. 2006; Fienberg et al. 1985; Ho et al. 2002). Researchers have extended the stochastic block model in different directions. For example, Airoldi et al. (2006) proposed a mixed-membership stochastic block model, Kemp et al. (2004) proposed a model that allows an unbounded number of clusters, and Hofman and Wiggins (2008) proposed a Bayesian approach based on the stochastic block model to infer module assignments and to identify the optimal number of modules. Our new model is also an extension of the stochastic block model. However, in comparison to the above approaches which focus on *static* social networks, our approach explicitly models the change of community memberships over time and therefore can discovery communities and their evolution simultaneously in dynamic social networks.

Recently, finding communities and their evolution in dynamic networks has gained more and more attention. Kumar et al. (2003) studied the evolution of the blogosphere as a graph in terms of the change of characteristics, (such as in-degree, out-degree, strongly connected components), the change of communities, as well as the burstiness in blog community. Leskovec et al. (2005) studied the patterns of growth for graphs in various fields and proposed generators that produce graphs exhibiting the discovered patterns. Palla et al. (2007) analyzed a co-authorship network and a mobile phone network, where both networks are dynamic. They use the clique percolation method (CPM) to extract communities at each timestep and then match communities in consecutive timesteps to analyze community evolution. They studied some interesting characteristics, such as community sizes, ages and their correlation, community auto-correlation (relative overlap between the same community at two timesteps t_1 and t_2 as a function of $\tau = t_2 - t_1$), etc. Toyoda and Kitsuregawa (2003) studied the evolution of Web communities from a series of Web archives. They first proposed algorithms for extracting communities in each timestep. And then they proposed different types of community changes, such as emerge, dissolve, grow, and shrink, as well as a set of metrics to quantify such changes for community evolution analysis. Spiliopoulou et al. (2006) proposed a framework, MONIC, to model and monitor cluster transitions over time. They defined a set of *external transitions* such as survive, split, disappear, to model transactions among different clusters and a set of *internal transitions*, such as size and location transitions to model changes within a community. Asur et al. (2007) introduced a family of events on both communities and individuals to characterize evolution of communities.

They also defined a set of metrics to measure the stability, sociability, influence and popularity for communities and individuals. Sun et al. (2007) proposed a parameter-free algorithm, GraphScope, to mine time-evolving graphs where the Minimum Description Length (MDL) principle is employed to extract communities and to detect community changes. Mei and Zhai (2005) extracted latent themes from text and used the evolution graph of themes for temporal text mining. In all these studies, however, community extraction and community evolution are analyzed in two separated stages. That is, when communities are extracted at a given timestep, historic community structure, which contains valuable information related to current community structure, is not taken into account.

There are some recent studies on evolutionary embedding and clustering that are closely related to our work. Sarkar and Moore (2005) proposed a dynamic method that embeds nodes into latent spaces where the locations of the nodes at consecutive timesteps are regularized so that dramatic change is unlikely. Chakrabarti et al. (2006) proposed the first evolutionary clustering methods where the cluster membership at time t is influenced by the clusters at time $t-1$. As a result, the cluster membership for a node at time t depends both on its relationship with other nodes at time t and on its cluster membership at time $t-1$. Tantipathananandh et al. (2007) proposed an optimization-based approach for modeling dynamic community structure. Chi et al. (2007) proposed an evolutionary version of the spectral clustering algorithm. They used graph cut as a metric for measuring community structures and community evolution. Lin et al. (2008, 2009a) extended the graph-factorization clustering (GFC) and proposed the FacetNet algorithm for analyzing dynamic communities. Ahmed and Xing (2008) extended temporal Dirichlet process mixture model for clustering problem for documents. In their model, the probabilities transiting between clusters are considered independent, while we consider the transition follows certain distribution. Lin et al. (2009b) extends (Lin et al. 2008) by modeling of content of documents. Tang et al. (2008) used joint matrix factorization method to discover the community evolution. Kim and Han (2009) proposed the particle-and-density based method to discover the evolution of communities. Its overall quality is measured by the combination of the history quality with the snapshot quality. A preliminary version of our work has been reported in Yang et al. (2009). We will conduct performance studies to compare our algorithm with some of these algorithms. Here we want to point out that compared to our new algorithm, none of these existing approaches has a rigorous probabilistic interpretation and they all are restricted to an online inference framework.

3 The dynamic stochastic block model

3.1 Notations

Before discussing the statistical models, we first introduce the notations that are used throughout this paper. We represent by $W^{(t)} \in \mathbb{R}^{n\times n}$ the *snapshot* of a social network at a given time step t (or snapshot network), where n is the number of nodes in the network. Each element w_{ij} in $W^{(t)}$ is the weight assigned to the link between nodes i and j: it can be the frequency of interactions (i.e., a natural number) or a binary number indicating the presence or absence of interactions between nodes i and j. For the time being, we focus on the binary link, which will be extended to other types of links in Sect. 7. For a dynamic social network, we use $\mathcal{W}_T = \{W^{(1)}, W^{(2)}, \ldots, W^{(T)}\}$ to denote a collection of snapshot graphs for a given social network over T discrete time steps. In our analysis and modeling, we first assume nodes in the social network remain unchanged during all the time steps, followed by

the extension to dynamic social networks where nodes can be removed from and added to networks.

We use $z_i \in \{1, \ldots, K\}$, where K is the total number of communities, to denote the community assignment of node i and we refer to z_i as the *community* of node i. We furthermore introduce $z_{ik} = [z_i = k]$ to indicate if node i is in the kth community where $[x]$ outputs one if x is true and zero otherwise. Community assignments matrix $Z = (z_{ik} : i \in \{1, \ldots, n\}, k \in \{1, \ldots, K\})$ includes the community assignments of all the nodes in a social network at a given time step. Finally, we use $\mathcal{Z}_T = \{Z^{(1)}, \ldots, Z^{(T)}\}$ to denote the collection of community assignments of all nodes over T time steps.

3.2 Stochastic block model (SBM)

We first briefly review the Stochastic Block Model (SBM). SBM is a well studied statistical model that has been successfully used in social network analysis (Hofman and Wiggins 2008; Holland and Leinhardt 1976). In the SBM model, a network is generated in the following way. First, each node is assigned to a community following a probability $\pi = \{\pi_1, \ldots, \pi_K\}$ where π_k is the probability for a node to be assigned to community k. Then, depending on the community assignments of nodes i and j (assuming that $z_{ik} = 1$ and $z_{jl} = 1$), the link between i and j is generated following a Bernoulli distribution with parameter P_{kl}. So the parameters of SBM are $\pi \in \mathbb{R}^K$, the prior distribution of the communities, and $P \in \mathbb{R}^{K \times K}$, the link generation probabilities. The diagonal element P_{kk} of P is called the "within-community" link probability for community k and the off-diagonal element $P_{kl}, k \neq l$ is called "between-community" link probability between communities k and l.

3.3 Dynamic stochastic block model (DSBM)

The traditional stochastic block model can only handle *static* networks. To extend it to handling *dynamic* networks, we propose a Dynamic Stochastic Block Model (DSBM) for modeling communities and their evolution in a unified probabilistic framework. Our DSBM is defined as the following. Assuming the community matrix $Z^{(t-1)}$ for time step $t-1$ is available, we use a transition matrix $A \in \mathbb{R}^{K \times K}$ to model the community matrix $Z^{(t)}$ at time step t. More specifically, for a node i, if node i was assigned to community k at time $t-1$ (i.e., $z_{ik}^{(t-1)} = 1$), then with probability A_{kk} node i will remain in community k at time step t and with probability A_{kl} node i will change to another community l where $k \neq l$. We have each row of A sums to 1, i.e., $\sum_l A_{kl} = 1$. Given the community memberships in $Z^{(t)}$, the link between nodes will be then decided stochastically by probabilities P in the SBM model.

The generative process of the Dynamic Stochastic Block Model and the graphical representation are shown in Table 1 and Fig. 1, respectively. Note that DSBM and SBM differ in how the community assignments are determined. In our DSBM model, instead of following a prior distribution π, the community assignments at any time t ($t > 1$) are determined by those at time $t-1$ through transition matrix A, where A aims to capture the dynamic evolution of communities.

3.4 Likelihood of the complete data

To express the data likelihood for the proposed DSBM model, we make two assumptions about the data generation process. First, link weight w_{ij} is generated independent of the other nodes/links, provided the membership z_i and z_j. Second, the community assignment

Table 1 The generative process of the Dynamic Stochastic Block Model (DSBM)

For time 1:
  generate the Social Network followed by SBM
For each time $t > 1$:
  generate $z_i^{(t)} \sim p(z_i^{(t)}|z_i^{(t-1)}, A)$
For each pair (i, j) at time t:
  generate $w_{ij}^{(t)} \sim Bernoulli(\cdot|P_{z_i^{(t)}, z_j^{(t)}})$

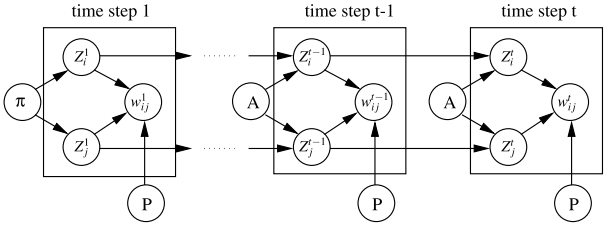


Fig. 1 The graphical representation of the Dynamic Stochastic Block Model (DSBM). For clarity, the figure only shows the representation for a pair of nodes i and j

$z_i^{(t)}$ of node i at time step t is independent of the other nodes/links, provided its community assignment $z_i^{(t-1)}$ at time $t-1$. Using these assumptions, we write the likelihood of the complete data for our DSBM model as follows

$$\begin{aligned}&\Pr(\mathcal{W}_T, \mathcal{Z}_T|\pi, P, A)\\ &= \prod_{t=1}^{T} \Pr(W^{(t)}|Z^{(t)}, P) \prod_{t=2}^{T} \Pr(Z^{(t)}|Z^{(t-1)}, A) \Pr(Z^{(1)}|\pi)\end{aligned} \tag{1}$$

where the emission probability $\Pr(W^{(t)}|Z^{(t)}, P)$ and the transition probability $\Pr(Z^{(t)}|Z^{(t-1)}, A)$ are

$$\begin{aligned}\Pr(W^{(t)}|Z^{(t)}, P) &= \prod_{i\sim j} \Pr(w_{ij}^{(t)}|z_i^{(t)}, z_j^{(t)}, P)\\ &= \prod_{i\sim j} \prod_{k,l} \left(P_{kl}^{w_{ij}^{(t)}} (1-P_{kl})^{1-w_{ij}^{(t)}} \right)^{z_{ik}^{(t)} z_{jl}^{(t)}}\end{aligned}$$

and

$$\begin{aligned}\Pr(Z^{(t)}|Z^{(t-1)}, A) &= \prod_{i=1}^{n} \Pr(z_i^{(t)}|z_i^{(t-1)}, A)\\ &= \prod_{i=1}^{n} \prod_{k,l} A_{kl}^{z_{ik}^{(t-1)} z_{il}^{(t)}},\end{aligned}$$

respectively. Note that in the above equations, $w_{ij}^{(t)} = 1$ if there exists a link between nodes i and j at time t, 0 otherwise. In addition, $z_{ik}^{(t)} z_{jl}^{(t)} = 1$ only if at time t node i belongs to community k and node j belongs to community l. Similarly, $z_{ik}^{(t-1)} z_{il}^{(t)} = 1$ only if node i belongs to community k at time $t-1$, and belongs to community l at time t, where k may

be equal to l. Furthermore, in our model self-loops are not considered and so in the above equations, $i \sim j$ means over all i's and j's such that $i \neq j$. So the above equations are a compact representation of our DSBM model.

Finally, term $\Pr(Z^{(1)}|\pi)$ is the probability of community assignments at the first time step and is expressed as

$$\Pr(Z^{(1)}|\pi) = \prod_{i=1}^{n} \prod_{k} \pi_k^{z_{ik}^{(1)}}.$$

4 Point estimation

For point estimation, we can use EM algorithm to get the maximum values of the parameters π, P, A and the approximate posterior distribution for the community assignments $\mathcal{Z}_T$. We take the variational EM algorithm. To run the variational EM algorithm, we assume the posterior distribution for $\mathcal{Z}_T$ can be factorized as $q(\mathcal{Z}_T) = \prod_{t=1}^{T} \prod_{i=1}^{n} q(z_i^{(t)})$. In the variational E-step, we obtain the posterior distribution $q(z_i^{(t)})$, and in the variational M-step, we obtain the maximum values for the parameters π, A, P. It can be shown that in the variational E-step, the posterior distribution $q(z_i^{(t)})$ is computed as

$t = 1$:

$$\ln q(z_i^{(1)}) = \sum_k z_{ik}^{(1)} \left(\ln \pi_k + \sum_{j \neq i} \sum_l E[z_{jl}^1](w_{ij}^{(1)} \ln P_{kl} + (1 - w_{ij}^{(1)}) \ln(1 - P_{kl})) \right) + \sum_k z_{ik}^{(1)} \sum_l E[z_{il}^{(2)}] \ln A_{kl} + \text{const},$$

$t \in [2, T-1]$:

$$\ln q(z_i^{(t)}) = \sum_k z_{ik}^{(t)} \left(\sum_{j \neq i} \sum_l E[z_{jl}^t](w_{ij}^{(t)} \ln P_{kl} + (1 - w_{ij}^{(t)}) \ln(1 - P_{kl})) \right) + \sum_k z_{ik}^{(t)} \sum_l \left(E[z_{il}^{(t+1)}] \ln A_{kl} + E[z_{il}^{(t-1)}] \ln A_{lk} \right) + \text{const},$$

$t = T$:

$$\ln q(z_i^{(T)}) = \sum_k z_{ik}^{(T)} \left(\sum_{j \neq i} \sum_l E[z_{jl}^T](w_{ij}^{(T)} \ln P_{kl} + (1 - w_{ij}^{(T)}) \ln(1 - P_{kl})) \right) + \sum_k z_{ik}^{(T)} \sum_l E[z_{il}^{(T-1)}] \ln A_{lk} + \text{const}.$$

In can seen that the approximate posterior distribution for $q(z_i^{(t)})$ is multinomial distribution of $q(z_i^{(t)}) = \prod_k \gamma_{ik}^{(t)^{z_{ik}^{(t)}}}$. In the M-step, we maximize the log-likelihood over the parameters π, P, A. The results are

$$\pi_k = \frac{\sum_i E[z_{ik}^{(1)}]}{\sum_k \sum_i E[z_{ik}^{(1)}]} = \frac{\sum_i \gamma_{ik}^{(1)}}{\sum_k \sum_i \gamma_{ik}^{(1)}},$$

$$P_{kl} = \frac{\sum_{t=1}^{T}\sum_{i\sim j}(E[z_{ik}^{(t)}]E[z_{jl}^{(t)}] + E[z_{il}^{(t)}]E[z_{jk}^{(t)}])w_{ij}^{(t)}}{\sum_{t=1}^{T}\sum_{i\sim j}(E[z_{ik}^{(t)}]E[z_{jl}^{(t)}] + E[z_{il}^{(t)}]E[z_{jk}^{(t)}])}$$
$$= \frac{\sum_{t=1}^{T}\sum_{i\sim j}(\gamma_{ik}^{(t)}\gamma_{jl}^{(t)} + \gamma_{il}^{(t)}\gamma_{jk}^{(t)})w_{ij}^{(t)}}{\sum_{t=1}^{T}\sum_{i\sim j}(\gamma_{ik}^{(t)}\gamma_{jl}^{(t)} + \gamma_{il}^{(t)}\gamma_{jk}^{(t)})},$$
$$A_{kl} = \frac{\sum_{t=2}^{T}\sum_{i} E[z_{ik}^{(t-1)}]E[z_{il}^{(t)}]}{\sum_{l}\sum_{t=2}^{T}\sum_{i} E[z_{ik}^{(t-1)}]E[z_{il}^{(t)}]} = \frac{\sum_{t=2}^{T}\sum_{i}\gamma_{ik}^{(t-1)}\gamma_{il}^{(t)}}{\sum_{t=2}^{T}\sum_{i}\gamma_{ik}^{(t-1)}}.$$

After running the variational EM algorithm, the final community assignments are obtained by

$$\mathcal{Z}_T^* = \max_{\mathcal{Z}_T} \ln q(\mathcal{Z}_T) \tag{2}$$

resulting $z_i^{(t)*} = \arg\max_k \gamma_{ik}^{(t)}$.

5 Bayesian inference

In order to predict memberships of nodes in a given dynamic social network, a straightforward approach is to first estimate the most likely values for parameters π, P, and A from the historical data, and then to infer the community memberships in the future using the estimated parameters. This is usually called point estimation in statistics, and is notorious for its instability when data is noisy. We address the limitation of point estimation by Bayesian inference (Bishop 2006). Instead of using the most likely values for the model parameters, we utilize the distribution of model parameters when computing the prediction.

5.1 Conjugate prior for Bayesian inference

We first introduce the prior distributions for model parameters π, P, and A.

The conjugate prior for π is the Dirichlet distribution

$$\Pr(\pi) = \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \prod_k \pi_k^{\gamma_k - 1} \tag{3}$$

where $\Gamma(\cdot)$ is the Gamma function. For the P matrix, we first assume it to be symmetric and therefore reduce the number of parameters to $\frac{n(n+1)}{2}$.

The conjugate prior for each parameter P_{kl} for $l \geq k$ is a Beta distribution, and therefore the prior distribution for P is

$$\Pr(P) = \prod_{k,l\geq k} \frac{\Gamma(\alpha_{kl} + \beta_{kl})}{\Gamma(\alpha_{kl})\Gamma(\beta_{kl})} P_{kl}^{\alpha_{kl}-1}(1 - P_{kl})^{\beta_{kl}-1}. \tag{4}$$

Finally, the conjugate prior for each row A is a Dirichlet distribution and the prior distribution for A is

$$\Pr(A) = \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \prod_l A_{kl}^{\mu_{kl}-1}. \tag{5}$$

5.2 Joint probability of the complete data

To make our presentation concise, we introduce the following notations.

$$n_k^{(t)} = \sum_i z_{ik}^{(t)}, \tag{6}$$

$$n_{k\to l}^{(t_1:t_2)} = \sum_{t=t_1+1}^{t_2} \sum_{i=1}^{n} z_{ik}^{(t-1)} z_{il}^{(t)}, \tag{7}$$

$$n_{k\to\cdot}^{(t_1:t_2)} = \sum_{t=t_1+1}^{t_2} \sum_{i=1}^{n} z_{ik}^{(t-1)}, \tag{8}$$

$$n_{kl}^{(t_1:t_2)} = \sum_{t=t_1}^{t_2} \sum_{i\sim j} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}), \tag{9}$$

$$\hat{n}_{kl}^{(t_1:t_2)} = \sum_{t=t_1}^{t_2} \sum_{i\sim j} w_{ij}^{(t)} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}). \tag{10}$$

Here are some descriptions of the above notations. $n_k^{(t)}$ represents the size, measured by the number of community members, of community k at time t. $n_{k\to l}^{(t_1:t_2)}$ represents the total number of transitions from community k to community l between time t_1 (exclusive) and t_2 (inclusive). $n_{k\to\cdot}^{(t_1:t_2)}$ represents the sum of the sizes of community k over time t_1 (exclusive) and t_2 (inclusive). $n_{kl}^{(t_1:t_2)}$ are the total number of pairs of nodes i and j, over time t_1 (inclusive) and t_2 (inclusive), where $i \neq j$, i belongs to community k and j belongs to community l, or the other way around. $\hat{n}_{kl}^{(t_1:t_2)}$ is defined similarly to $n_{kl}^{(t_1:t_2)}$, except that $\hat{n}_{kl}^{(t_1:t_2)}$ is a weighted sum, weighed by the element $w_{ij}^{(t)}$ in $W^{(t)}$.

Using these notations, and with the prior distributions of the model parameters, Theorem 1 gives the closed form expression for the joint probability of the complete data that is marginalized over the distribution of model parameters.

Theorem 1 *With the priors of parameters $\theta = \{\pi, P, A\}$ defined in (3)–(5) together with the notations given in (6)–(10), the joint probability of observed links and unobserved community assignments is proportional to*

$$\begin{aligned}
\Pr(\mathcal{W}_T, \mathcal{Z}_T) &= \int \Pr(\mathcal{W}_T, \mathcal{Z}_T|\theta) \Pr(\theta) d\theta \\
&\propto \prod_k \Gamma(n_k^{(1)} + \gamma_k) \prod_k \frac{\prod_l \Gamma(n_{k\to l}^{(1:T)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(1:T)} + \sum_l \mu_{kl})} \\
&\quad \times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(1:T)} + \alpha_{kl}, n_{kl}^{(1:T)} - \hat{n}_{kl}^{(1:T)} + \beta_{kl}\right) \\
&\quad \times \prod_k B\left(\frac{\hat{n}_{kk}^{(1:T)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1:T)} - \hat{n}_{kk}^{(1:T)}}{2} + \beta_{kk}\right)
\end{aligned}$$

where $B(\cdot)$ is the Beta function.

The proof of the theorem is provided in the appendix. In this Bayesian inference framework, to obtain the community assignment of each node at each time step, we need to compute the posterior probability $\Pr(\mathcal{Z}_T|\mathcal{W}_T)$.

Next, we introduce two versions of the inference framework—an offline inference approach and an online inference approach.

5.3 Offline inference

In our offline inference, it is assumed that the link data of all time steps are accessible and therefore, the community assignments of all nodes in all time steps can be decided simultaneously by maximizing the posterior probability, i.e.,

$$\mathcal{Z}_T^* = \arg\max_{\mathcal{Z}_T} \Pr(\mathcal{Z}_T|\mathcal{W}_T) = \arg\max_{\mathcal{Z}_T} \Pr(\mathcal{W}_T, \mathcal{Z}_T) \tag{11}$$

where $\Pr(\mathcal{W}_T, \mathcal{Z}_T)$ is given in Theorem 1. Note that in offline inference, the community membership of each node at every time step t is decided by the link data of all time steps in a *retrospective* way, even the link data of time steps later than t. In other words, we try to fit the community membership of each node at time t to the entire available data from time 1 to time T. Given this observation, we expect offline inference to deliver more reliable estimation of community memberships than the online learning that is discussed in the next subsection.

5.4 Online inference

In our online inference, the community memberships are learned incrementally over time. Assume we have decided the community membership $Z^{(t-1)}$ at time step $t-1$, and observed the links $W^{(t)}$ at time t. We decide the community assignments at time t by maximizing the posterior probability of community assignments at time t given $Z^{(t-1)}$ and $W^{(t)}$, i.e.,

$$Z^{*(t)} = \arg\max_{Z^{(t)}} \Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}).$$

Hence, to decide $Z^{(t)}$, the key is to efficiently compute $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$ except for time step 1 in which we need to compute $\Pr(Z^{(1)}|W^{(1)})$. The following theorem, whose proof is given the appendix, provides closed form solutions for the two probabilities. It is important to note that both probabilities are computed by averaging over the distribution of the model parameters.

Theorem 2 *With the priors of parameters $\theta = \{\pi, P, A\}$ given in (3)–(5), the posterior probability of unobserved community assignments given the observed links and the commu-*

nity assignments at previous time step is proportional to

$$\begin{aligned}
\Pr(Z^{(1)}|W^{(1)}) &\propto \prod_k \Gamma(n_k^{(1)} + \gamma_k) \\
&\quad \times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(1)} + \alpha_{kl}, n_{kl}^{(1)} - \hat{n}_{kl}^{(1)} + \beta_{kl}\right) \\
&\quad \times \prod_k B\left(\frac{\hat{n}_{kk}^{(1)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1)} - \hat{n}_{kk}^{(1)}}{2} + \beta_{kk}\right), \\
\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) &\propto \prod_k \left(\prod_l \frac{\Gamma(n_{k\to l}^{(t-1:t)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(t-1:t)} + \sum_l \mu_{kl})}\right) \\
&\quad \times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(t)} + \alpha_{kl}, n_{kl}^{(t)} - \hat{n}_{kl}^{(t)} + \beta_{kl}\right) \\
&\quad \times \prod_k B\left(\frac{\hat{n}_{kk}^{(t)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(t)} - \hat{n}_{kk}^{(t)}}{2} + \beta_{kk}\right).
\end{aligned} \tag{12}$$

In online inference, it is assumed that data arrives sequentially and historic community assignments are not updated upon the arrival of new data. Therefore, the online inference algorithm is done *progressively* and can be implemented more efficiently than the offline inference algorithm.

6 Inference algorithm

In general it is an intractable problem to optimize the posterior probabilities in the offline and online inference algorithms introduced in the previous section. As a consequence, we appeal to the Gibbs sampling method (Geman and Geman 1984; Griffiths and Steyvers 2004) for the solutions. In Gibbs sampling, we need to compute the conditional probability of the community assignment of each node conditioned on the community assignments of other nodes. We will first describe the algorithm and then analyze the time complexity of the proposed algorithm.

6.1 Gibbs sampling algorithm

For offline inference, we need to compute the conditional probability $\Pr(z_i^{(t)}|\mathcal{Z}_{T,\{i,t\}^-}, \mathcal{W}_T)$, via $\Pr(\mathcal{Z}_T|\mathcal{W}_T)$, where $\mathcal{Z}_{T,\{i,t\}^-}$ are the community assignments of all nodes at all time steps except node i at time step t. This can be computed by marginalizing $z_i^{(t)}$ in (11). Similarly, for online learning, we need to compute the conditional probability $\Pr(z_i^{(t)}|Z_{i^-}^{(t)}, W^{(t)}, Z^{(t-1)})$, where $Z_{i^-}^{(t)}$ is the collection of community assignments of all nodes, except node i, at time step t. This can be computed by marginalizing $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$. The following algorithms describe a simulated annealing version of our inference algorithm.

Algorithm 1 *Probabilistic simulated annealing algorithm*

1. *Randomly initialize the community assignment for each node at time step t* (*online inference*) *or at all time steps* (*offline learning*); *select the temperature sequence* $\{T_1, \ldots, T_M\}$ *and the iteration number sequence* $\{N_1, \ldots, N_M\}$.

2. *For each iteration* $m = 1, \ldots, M$, *run* N_m *iterations of Gibbs sampling with target distributions* $\exp\{\log \Pr(\mathcal{Z}_T|\mathcal{W}_T)/T_m\}$ (*offline case*) *or* $\exp\{\log \Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})/T_m\}$ (*online case*).

Algorithm 2 *Gibbs sampling algorithm*

1. *Compute the following statistics with the initial assignments*:

$$n_k^{(1)}$$
$$n_{kl}^{(1:T)}, \hat{n}_{kl}^{(1:T)} \text{ or } n_{kl}^{(t)}, \hat{n}_{kl}^{(t)}$$
$$n_{k\to l}^{(1:T)}, n_{k\to\cdot}^{(1:T)} \text{ or } n_{k\to l}^{(t-1:t)}, n_{k\to\cdot}^{(t-1:t)}.$$

2. *For each iteration* $m_i = 1 : N_m$, *and for each node* $i = 1 : n$ *at each time* t
   - *Compute the objective function in Simulated Annealing*

$$\exp\left\{\log \Pr(z_i^{(t)}|\mathcal{Z}_{T,\{i,t\}^-}, \mathcal{W}_T)/T_m\right\} \quad \text{or}$$
$$\exp\left\{\log \Pr(z_i^{(t)}|Z_{i^-}^{(t)}, W^{(t)}, Z^{(t-1)})/T_m\right\}$$

     up to a constant using the current statistics, *and then obtain the normalized distribution*. (*Note*: *the two objective functions correspond to the offline inference and the online inference*, *respectively*.)
   - *Sample the community assignment for node* i *according to the distribution obtained above*, *update it to the new one*.
   - *Update the statistics*.

6.2 Time complexity

In our implementation, we adopt several techniques to improve the efficiency of the algorithm. First, since in each step of the sampling, only one node i at a given time t changes its community assignment, almost all the statistics can be updated incrementally to avoid recomputing. Second, our algorithm is designed to take advantage of the sparseness of the matrix $W^{(t)}$. For instance, we exploit the sparseness of $W^{(t)}$ to facilitate the computation of $\hat{n}_{kl}^{(t_1:t_2)}$. We give the time complexity as the following.

Theorem 3 *The time complexity of our implementation of the Gibbs sampling algorithm is* $O(nT + eT + K^2T + NT(eC_1 + nC_2))$ *where* e *is the maximal number of edges over all the time steps in the social network*, N *is the number of iterations in Gibbs sampling*, C_1 *and* C_2 *are constants*.

Proof In the initial computation for the statistics in (6)–(10), $O(n)$ is the time for computing $n_k^{(1)}, \forall k$; $O(nT)$ is the time complexity of computing $n_{k\to l}^{(1:T)}$, and $n_{k\to\cdot}^{(1:T)}, \forall k, l$; $O(eT)$ is the time complexity of computing $\hat{n}_{kl}^{(1:T)}, \forall k, l$. In these computations, we make use the sparseness of $z_i^{(t)}$, which has only one non-zero value, and the sparseness of $W^{(t)}$; $O(nT + K^2T)$ is the time for computing $n_{kl}^{(1:T)}, \forall k, l$. In the subsequent updating at each iteration of the Gibbs sampling algorithm, only one $z_i^{(t)}$ is possibly changed, the updating only takes $O(e_i^t C_1 + C_2)$,

where e_i^t is the number of edges associated with node i at time t, summing over all nodes at all time steps, $O(eTC_1 + nTC_2)$ is the time required to update the statistics after updating all community assignments at each iteration. Finally adding the time together, the time complexity of the Gibbs sampling algorithm is $O(n + eT + nT + nT + K^2T + N(eTC_1 + nTC_2))$, which is also $O(nT + eT + K^2T + NT(eC_1 + nC_2))$. □

As can be seen, when the social network is sparse and when the degree of each node is bounded by a constant, the running time of each iteration of our Gibbs sampling algorithm is linear in the size of the social network.

7 Extensions

In this section, we present two extensions to our basic framework, including how to handle different types of links and how to handle insertion and deletion of nodes in the network. In addition, we discuss how to choose the hyperparameters in our model.

7.1 Handling different types of link

So far, we have used binary links in our model, where the binary links (i.e., either $w_{ij} = 1$ or $w_{ij} = 0$) indicate the presence or absence of a relation between a pair of nodes. However, there exist other types of links in social networks as well. Here we show how to extend our model to handle two other cases: when $w_{ij} \in \mathcal{N}$ and when $w_{ij} \in \mathcal{R}^+$.

In some applications, w_{ij} indicates the frequency of certain interactions. For example, w_{ij} may represent the occurrence of interactions between two bloggers during a day, the number of papers that two authors co-authored during a year, etc. In such cases, w_{ij} can be any non-negative integer. Our current model actually can handle this case with little change: the emission probability

$$\Pr(w_{ij}|z_i, z_j) = \prod_{k,l} (P_{kl}^{w_{ij}}(1 - P_{kl}))^{z_{ik}z_{jl}} \tag{13}$$

remains valid for $w_{ij} \in \mathcal{N}$, except that instead of a Bernoulli distribution (i.e., $w_{ij} = 0$ or 1), now w_{ij} follows a geometric distribution. Note that the $(1 - P_{kl})$ term is needed to take into account the case where there is no edge between i and j. With minor modifications, we give the joint probability $\Pr(\mathcal{W}_T, \mathcal{Z}_T)$ in offline inference, and the conditional probability $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$ in online inference as follows:

$$\begin{aligned}
\Pr(\mathcal{W}_T, \mathcal{Z}_T) \propto & \prod_k \Gamma(n_k^{(1)} + \gamma_k) \prod_k \frac{\prod_l \Gamma(n_{k\to l}^{(1:T)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(1:T)} + \sum_l \mu_{kl})} \\
& \times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(1:T)} + \alpha_{kl}, n_{kl}^{(1:T)} + \beta_{kl}\right) \\
& \times \prod_k B\left(\frac{\hat{n}_{kk}^{(1:T)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1:T)}}{2} + \beta_{kk}\right), \\
\Pr(Z^{(1)}|W^{(1)}) \propto & \prod_k \Gamma(n_k^{(1)} + \gamma_k)
\end{aligned}$$

$$\times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(1)} + \alpha_{kl}, n_{kl}^{(1)} + \beta_{kl}\right)$$

$$\times \prod_{k} B\left(\frac{\hat{n}_{kk}^{(1)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1)}}{2} + \beta_{kk}\right),$$

$$\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \propto \prod_{k}\left(\prod_{l} \frac{\Gamma(n_{k\to l}^{(t-1:t)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(t-1:t)} + \sum_l \mu_{kl})}\right)$$

$$\times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(t)} + \alpha_{kl}, n_{kl}^{(t)} + \beta_{kl}\right)$$

$$\times \prod_{k} B\left(\frac{\hat{n}_{kk}^{(t)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(t)}}{2} + \beta_{kk}\right).$$

In other applications, w_{ij} indicates the similarity or distance between nodes. For example, w_{ij} may represent the topic similarity between posts written by two bloggers, the content similarity between a paper and the papers it cites, etc. In such cases, $w_{ij} \in \mathcal{R}^+$ belongs to the set of non-negative real numbers. In such a case, we can first discretize w_{ij} by using finite bins and then introduce the emission probabilities as before. Another way to handle the case when $w_{ij} \in \mathcal{R}^+$ is suggested by Zhu (2005), which is to introduce a k-nearest neighbor graph and therefore reduce the problem to the case when $w_{ij} = 0$ or 1.

7.2 Handling the variability of nodes

In dynamic social networks, at a given time, new individuals may join in the network and old ones may leave. To handle insertion of new nodes and deletion of old ones, existing algorithm such as (Chi et al. 2007) and (Lin et al. 2008) use some heuristics, e.g., by assuming that all the nodes are in the network all the time but in some time steps certain nodes have no incident links. In comparison, in both the online and the offline versions of our algorithm, no such heuristics are necessary. For example, for online inference, let S_t denote the set of nodes at time t, $I_t = S_t \bigcap S_{t-1}$ be set of nodes appearing in both time steps t and $t-1$. $U_t = S_t - S_{t-1}$ be the new nodes at time t. Then we can naturally model the posterior probability of the community assignments at time t as

$$\begin{aligned}\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) &\propto \Pr(Z^{(t)}, W^{(t)}|Z^{(t-1)}) \\ &= \Pr(W^{(t)}|Z^{(t)}) \Pr(Z_{I_t}^{(t)}|Z_{I_t}^{(t-1)}) \Pr(Z_{U_t}^{(t)})\end{aligned} \tag{14}$$

and we can directly write the part corresponding to (12) in Theorem 2 as

$$\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \propto \prod_{k} \Gamma(n_{k,U_t}^{(t)} + \gamma_k) \times \prod_{k}\left(\prod_{l} \frac{\Gamma(n_{k\to l,I_t}^{(t-1:t)} + \mu_{kl})}{\Gamma(n_{k\to\cdot,I_t}^{(t-1:t)} + \sum_l \mu_{kl})}\right)$$

$$\times \prod_{k,l>k} B\left(\hat{n}_{kl,S_t}^{(t)} + \alpha_{kl}, n_{kl,S_t}^{(t)} - \hat{n}_{kl,S_t}^{(t)} + \beta_{kl}\right)$$

$$\times \prod_{k} B\left(\frac{\hat{n}_{kk,S_t}^{(t)}}{2} + \alpha_{kk}, \frac{n_{kk,S_t}^{(t)} - \hat{n}_{kk,S_t}^{(t)}}{2} + \beta_{kk}\right)$$

where $n^*_{*,S}$ is the corresponding statistics evaluated on the nodes set of S. Similar results can be derived for the offline learning algorithm. In brief, our model can handle the insertion and deletion of nodes without using any heuristics.

7.3 Hyperparameters

In this subsection, we discuss the roles of the hyperparameters (γ, α, β, and μ) and give some guidelines on how to choose the values for these hyperparameters. In the experimental studies, we will further investigate the impact of the values of these hyperparameters on the performance of our algorithm.

- γ is the hyperparameter for the prior of π. We can interpret the γ_k as an effective number of observations of $z_{ik} = 1$. Without other prior knowledge we set all γ_k to be the same.
- α, β are the hyperparameters for the prior of P. As stated before, we discriminate two probabilities in P, i.e., P_{kk} the "within-community" link probability, and $P_{kl,l\neq k}$ the "between-community" link probability. For the hyperparameters, we set two groups of values, i.e., (1) $\alpha_{kk}, \beta_{kk}, \forall k$ and (2) $\alpha_{kl,l\neq k}, \beta_{kl,l\neq k}$. Because we have the prior knowledge that nodes in the same community have higher probability to link to each other than nodes in different communities, we set $\alpha_{kk} \geq \alpha_{kl,l\neq k}, \beta_{kk} \leq \beta_{kl,l\neq k}$.
- μ is the hyperparameter for A. $A_{k*} = \{A_{k1}, \ldots, A_{kk}, \ldots, A_{kK}\}$ are the transition probabilities for nodes to switch from the kth community to other (including coming back to the kth) communities in the following time step. $\mu_{k*} = \{\mu_{k1}, \ldots, \mu_{kk}, \ldots, \mu_{kK}\}$ can be interpreted as effective number of nodes in the kth community switching to other (including coming back to the kth) communities in the following time step. With prior belief that most nodes will not change their community memberships over time, we set $\mu_{kk} \geq \mu_{kl,l\neq k}$.

Finally, how to select the exact values for the hyperparameters γ, α, β, and μ is further described in the empirical studies.

8 Experimental studies

In this section, we conduct several experimental studies. First, we show that the performance of our algorithms is not sensitive to most hyperparameters in the Bayesian inference and for the only hyperparameters that impact the performance significantly, we provide a principled method for automatic parameter selection. Second, we show that our Gibbs-sampling-based algorithms have very fast convergence rate, which makes our algorithms very practical for real applications. Third, by using a set of benchmark datasets with a variety of characteristics, we demonstrate that our algorithms clearly outperform several state-of-the-art algorithms in terms of discovering the true community memberships and capturing the true evolution of community memberships. Finally, we use three real datasets of dynamic social networks to illustrate that from these datasets, our algorithms are able to reveal interesting insights that are not directly obtainable from other algorithms. In all the following experiments, the Gibbs-sampling algorithm is run with temperature sequence of 1:-0.1:0, and iteration number sequence of [20, 10, 10, 10, 10, 10, 10, 5, 5, 5, 5], and totally 100 iterations.

8.1 Performance metrics

The experiments we conducted can be categorized into two types, those with ground truth available and those without ground truth. By ground truth we mean the true community

membership of each node at each time step. When the ground truth is available, we measure the performance of an algorithm by the *normalized mutual information* between the true community memberships and those given by the algorithm. More specifically, if the true community memberships are represented by $\mathcal{C} = \{C_1, \ldots, C_K\}$ and those given by the algorithm are represented by $\mathcal{C}' = \{C'_1, \ldots, C'_K\}$, then the *mutual information* between the two is defined as

$$\widehat{MI}(\mathcal{C}, \mathcal{C}') = \sum_{C_i, C'_j} p(C_i, C'_j) \log \frac{p(C_i, C'_j)}{p(C_i)p(C'_j)}$$

and the *normalized mutual information* is defined by

$$MI(\mathcal{C}, \mathcal{C}') = \frac{\widehat{MI}(\mathcal{C}, \mathcal{C}')}{\max(H(\mathcal{C}), H(\mathcal{C}'))}$$

where $H(\mathcal{C})$ and $H(\mathcal{C}')$ are the entropies of the partitions $\mathcal{C}$ and $\mathcal{C}'$. The value of MI is between 0 and 1 and a higher MI value indicates that the result given by the algorithm $\mathcal{C}'$ is closer to the ground truth $\mathcal{C}$. This metric MI has been commonly used in the information retrieval field (Gong and Xu 2007; Xu and Gong 2004).

Where there is no ground truth available in the dataset, we measure the performance by using the metric of *modularity* which is proposed by Newman and Girvan (2004) for measuring community partitions. For a given community partition $\mathcal{C} = \{C_1, \ldots, C_K\}$, the modularity is defined as

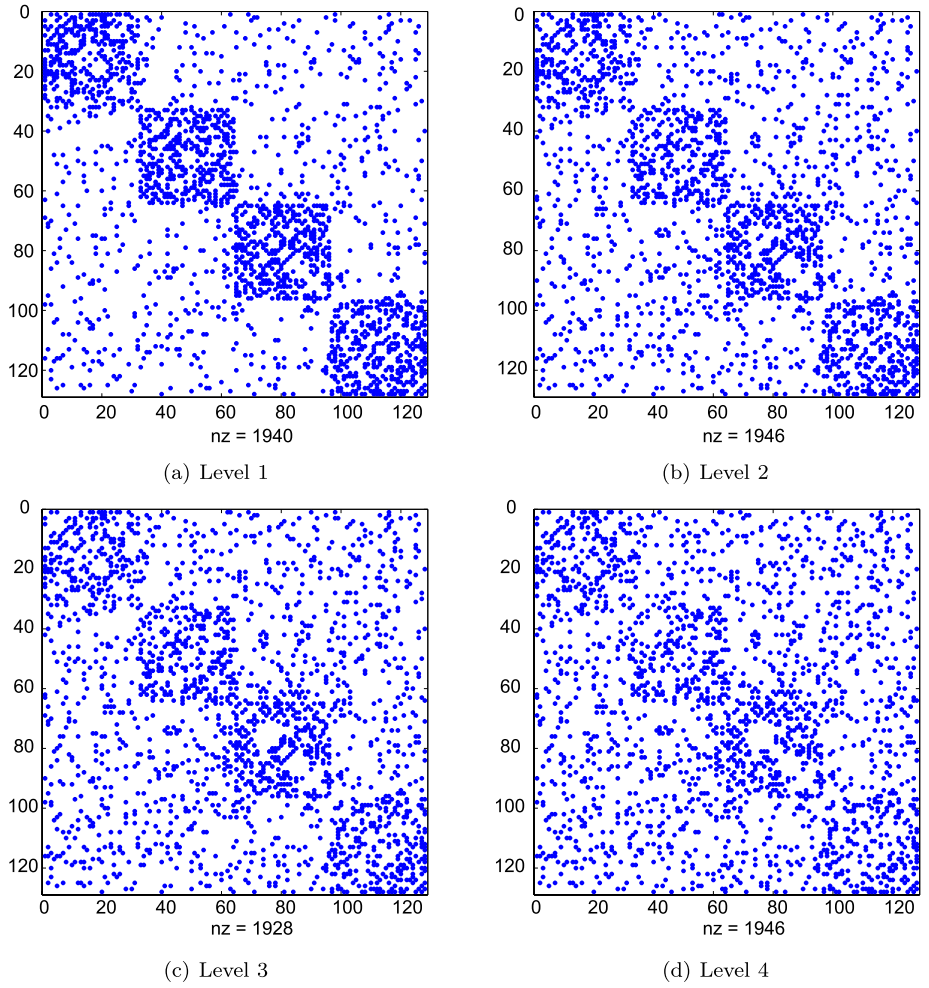
$$Modu(\mathcal{C}) = \sum_k \left[\frac{Cut(V_k, V_k)}{Cut(V, V)} - \left(\frac{Cut(V_k, V)}{Cut(V, V)} \right)^2 \right]$$

where V represents all the nodes in the social network and V_k indicates the set of nodes in the kth community C_k. $Cut(V_i, V_j) = \sum_{p \in V_i, q \in V_j} w_{pq}$. As state in (Newman and Girvan 2004), modularity measures how likely a network is generated due to the proposed community structure versus generated by a random process. Therefore, a higher modularity value indicates a community structure that better explains the observed social network. Many existing studies, such as (Brandes et al. 2008; Chen et al. 2009; Lin et al. 2008; White and Smyth 2005), have used this metric for performance analysis.

8.2 Experiments on synthetic datasets

8.2.1 Data generator

We generate synthetic data by following a procedure suggested by Newman and Girvan (2004). The data consists of 128 nodes that belong to 4 communities with 32 nodes in each community. Links are generated in the following way. For each pair of nodes that belong to the same community, the probability that a link exists between them is p_{in}; the probability that a link exists between a pair of nodes belonging to different communities is p_{out}. However, by fixing the average degree of nodes in the network, which we set to be 16 in our datasets, only one of p_{in} and p_{out} can change freely. In other words, a single parameter z, which represents the mean number of edges from a node to nodes in other communities, is enough to describe the data. By increasing z (and therefore p_{out}), the network becomes more noisy in the sense that the community structure becomes less obvious and hard to detect. In this study, we generate datasets under four different noise levels



(a) Level 1 (b) Level 2

(c) Level 3 (d) Level 4

Fig. 2 The adjacency matrices for the datasets with different noise levels, where the x and y axes represent the nodes, and each dot represents a link between the corresponding pair of nodes

by setting $z = 2, 3, 4, 5$, which correspond to $p_{in} = 0.1935$ ($p_{out} = 0.0208$), $p_{in} = 0.1613$ ($p_{out} = 0.0312$), $p_{in} = 0.1290$ ($p_{out} = 0.0417$), and $p_{in} = 0.0968$ ($p_{out} = 0.0521$), respectively. The adjacency matrices for the datasets at the four noise levels are shown in Fig. 2.

The above network generator described by Newman et al. can only generate static networks. To study dynamic evolution, we let the community structure of the network evolve in the following way. We start with introducing evolution to the community memberships: at each time step after time step 1, we randomly choose 10% of the nodes to leave their original community and join the other three communities at random. After the community memberships are decided, links are generated by following the probabilities p_{in} and p_{out} as before. We generate the network with community evolution in this way for 10 time steps.

8.2.2 *Hyperparameters*

In the first experiment, we study the impact of the hyperparameters on the performance of our algorithm. Figure 3 shows the performance of our algorithm, in terms of the average normalized mutual information and the average modularity over all time steps, under a large range of values for the hyperparameters γ (for the initial probability π) and μ (for the transition matrix A), respectively. As can be seen, the performance varies little under different values for γ and μ, which verifies that our algorithm is robust to the setting of these hyperparameters. As a result, in the following experiments, unless stated otherwise, we simply set $\gamma = 1$ and $\mu_{kk} = 10$. Note that we only show the results of our online inference algorithm for the dataset with noise level 2. The results for the dataset with other noise levels and for the offline inference algorithm are similar and therefore are not shown here.

However, the performance of our algorithm is somewhat sensitive to the hyperparameters α and β for P, which is the stochastic matrix representing the community structure at each time step. In Fig. 4(a) we show the performance of our algorithm under a large range of α and β values, which demonstrates that the performance varies under different α and β values. This result makes sense because α and β are crucial for the stochastic model to correctly capture the community structure of the network. For example, the best performance is achieved when α is in the same range as the total number of links in the network. In addition, we see a clear correlation between the accuracy with respect to the ground truth, which is not seen by our algorithm, and the modularity, which is available to our algorithm. This correlation is clearly demonstrated in Fig. 4(b), where we scaled the modularity so that it has the same mean as the average mutual information. As a result, we can use the modularity value as a validation metric to automatically choose good values for α and β. All the experimental results reported in the following are obtained from this automatic validation procedure.

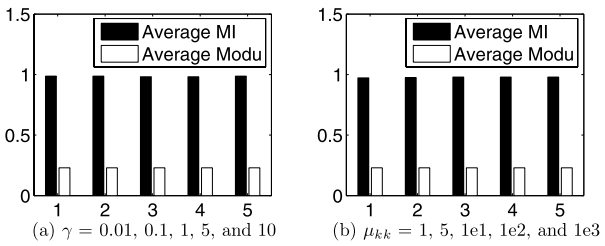


Fig. 3 The performance, in terms of (**a**) the average normalized mutual information and (**b**) the average modularity over all time steps, under different values for γ and μ_{kk} (with $\mu_{kl} = 1, \forall k \neq l$), which shows that the performance is not sensitive to γ and μ

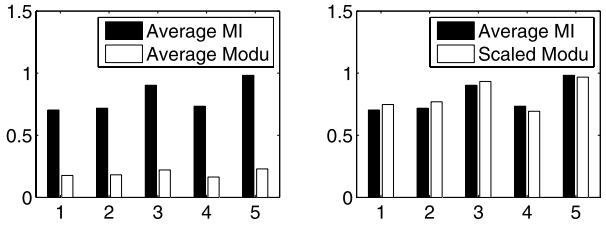


Fig. 4 The performance, in terms of (**a**) the average normalized mutual information vs. the average modularity and (**b**) the average normalized mutual information vs. the scaled modularity, over all time steps, under the cases with α and β valued at α, β are ($\alpha_{kk} = 1, \beta_{kl} = 1$), ($\alpha_{kk} = 5, \beta_{kl} = 1$), ($\alpha_{kk} = 10, \beta_{kl} = 1$), ($\alpha_{kk} = 1e2, \beta_{kl} = 10$), ($\alpha_{kk} = 1e4, \beta_{kl} = 10$), and in all the cases $\alpha_{kl,l\neq k} = 1$

8.2.3 *Comparison with the baseline algorithms*

In this experiment, we compare the performance of the online and offline versions of our DSBM algorithm with those of two recently proposed algorithms for analyzing dynamic communities—the dynamic graph-factorization clustering algorithm (FacetNet) by Lin et al. (2008) and the evolutionary spectral clustering algorithm (EvolSpect) by Chi et al. (2007). In addition, we also provide the performance of the static versions for all the algorithms—static stochastic block models (SSBM, Holland and Leinhardt 1976) for DSBM, static graph-factorization clustering (SGFC, Yu et al. 2005) for FacetNet, and static spectral clustering (SSpect, Shi and Malik 2000) for EvolSpect.

Figure 5 presents the performance, in terms of the normalized mutual information with respect to the ground truth over the 10 time steps, of all the algorithms for the four datasets with different noise levels. We can obtain the following observations from the results. First, our DSBM algorithms have the best accuracy and outperform all other baseline algorithms at every time step for all the four datasets. Second, the offline version of our algorithm, which takes into consideration all the available data simultaneously, has better performance than that of the online version. Third, the evolutionary versions of all the algorithms outperform their static counterparts in most cases, which demonstrates the advantages of the dynamic models in capturing community evolution in dynamic social networks. We obtain similar results for the modularity metric and due to the limit of space, we do not include them here.

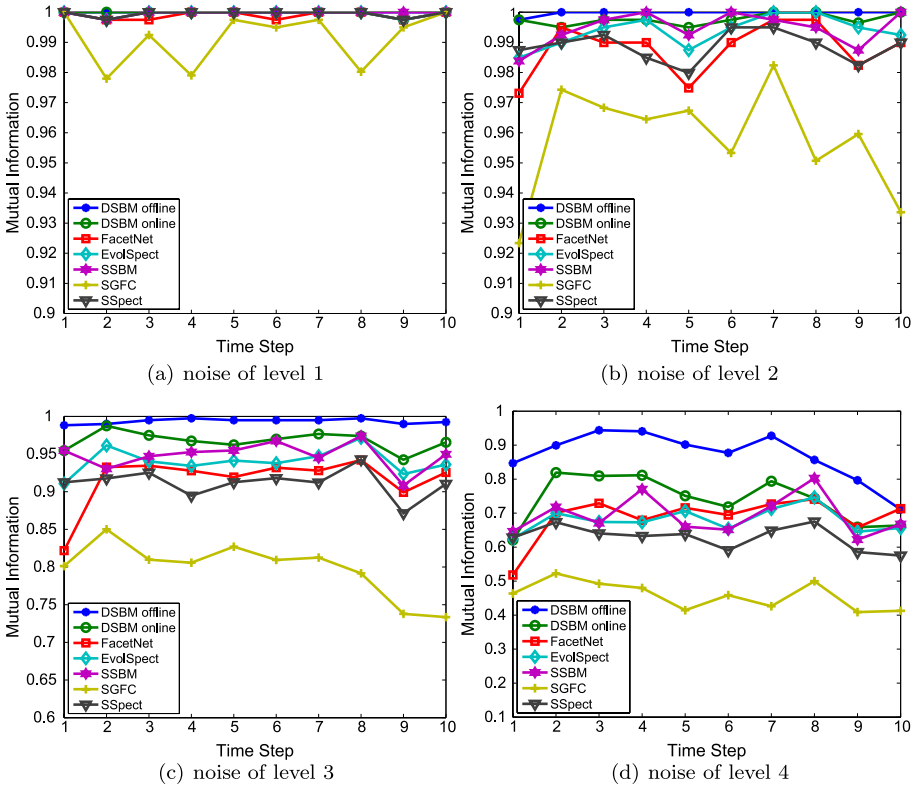


Fig. 5 The normalized mutual information with respect to the ground truth over the 10 time steps, of all the algorithms on the four datasets with different noise levels

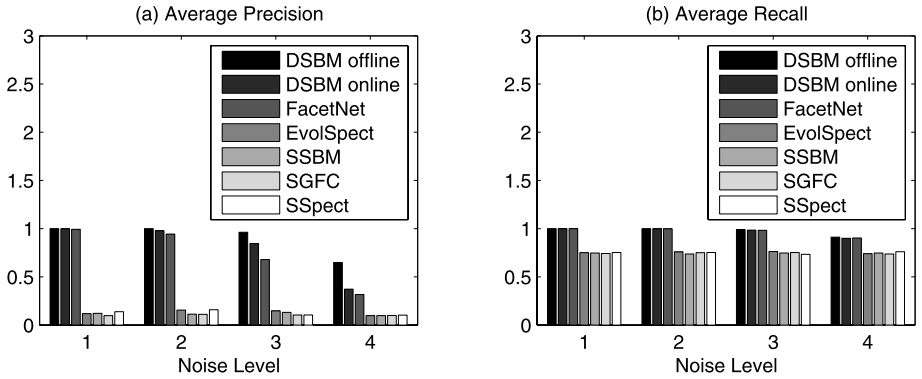
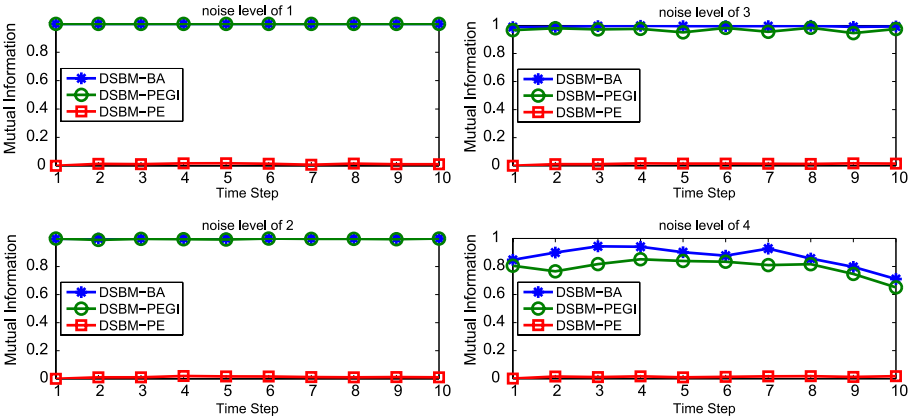


Fig. 6 (**a**) The average precision and (**b**) the average recall, for detecting the nodes that switch their community memberships, over all the time steps for the four datasets with different noise levels

Next, we investigate which algorithms can capture the community evolution more faithfully. For this purpose, since we have the ground truth on which nodes actually changed their communities at each time step, we compute the precision (the fraction of nodes, among those an algorithm found switched communities, that really changed their communities) and the recall (the fraction of nodes, among those really changed their communities, that an algorithm found switched communities) for all the algorithms. Figure 6 shows the average precision and recall for all the algorithms over all the time steps for the four datasets with different noise levels. As can be seen, our DSBM algorithms have the best precision and the best recall values for all the four datasets, which illustrates that our algorithms can capture the true community evolution more faithfully than the baseline algorithms.

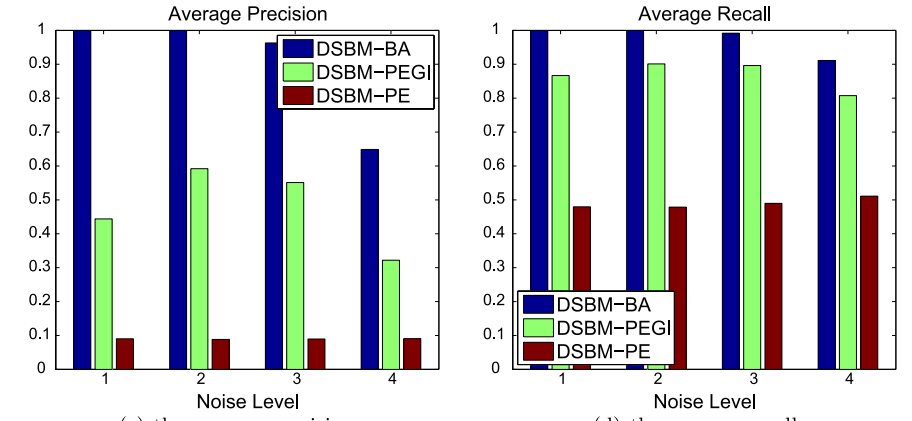
8.2.4 Comparison with point estimation

In this experiment, we compare the performance of the proposed model by using Bayesian inference with that by using point estimation. With this comparison, we can validate the benefit of Bayesian inference. From Sect. 4, we can see that the point estimation approach is best fit into the offline setup. Thus we compare the dynamic stochastic block model by using Bayesian inference with that by using point estimation in the offline setting. We use the same synthetic data sets as in last experiment as shown in Fig. 2. The results are shown in Fig. 7. In the figure, DSBM-BA refers to the Bayesian inference for dynamic stochastic block model, and DSBM-PE refers to the point estimation for the dynamic stochastic block model with random initializations for the parameters γ. It is clear from the figure that using Bayesian Inference is much better than using point estimation. The reason for the bad performance of DSBM-PE is that the EM algorithm converges to a local optimum that there are only two communities found at each time step. Since the point estimation is heavily sensitive to initializations, we also include the result of using point estimation with good initializations as refereed to DSBM-PEGI. To obtain good initializations, we run Bayesian inference with 10 iterations of Gibbs sampling algorithm and initialize γ by the resulting community assignments. From the performance we can see that on the one hand, by comparing DSBM-PEGI with DSBM-PE, we see the benefit of using Bayesian Inference to get good initializations for point estimation; on the other hand, from Fig. 7 we can see even with good initializations, the point estimation approach still has poorer performance than the Bayesian inference approach, especially on capturing the community changes for individuals as measured by precision and recall and also when the data is more noisy.



(a) Mutual information at noise of level 1 and level 2

(b) Mutual Information at noise of level 3 and level 4



(c) the average precision

(d) the average recall

Fig. 7 The normalized mutual information with respect to the ground truth over the 10 time steps, of all the algorithms on the four datasets with different noise levels

8.2.5 Convergence rate and running time

In our algorithms, we adopt the Gibbs sampling for Bayesian inference. In this experiment, we show that this Gibbs sampling procedure converges very quickly. In Fig. 8, we show how the value of the objective function changes over the number of iterations at each time step. As can be seen, the first time step requires more iterations but even for the first time step, fewer than 20 iterations are enough for the algorithm to converge. For the time steps 2 to 10, by using the results at the previous time step as the initial values, the algorithm converges in just a couple of iterations. The total running time of the proposed algorithm on the synthetic data set with 128 nodes, 1024 edges, and 10 times is about 1 minutes on a 2.0 GHz CPU 2.0 GB memory Linux machine. This result, together with the time complexity analysis in Sect. 6.2, demonstrates that our algorithm is practical and is scalable to large social networks in real applications.

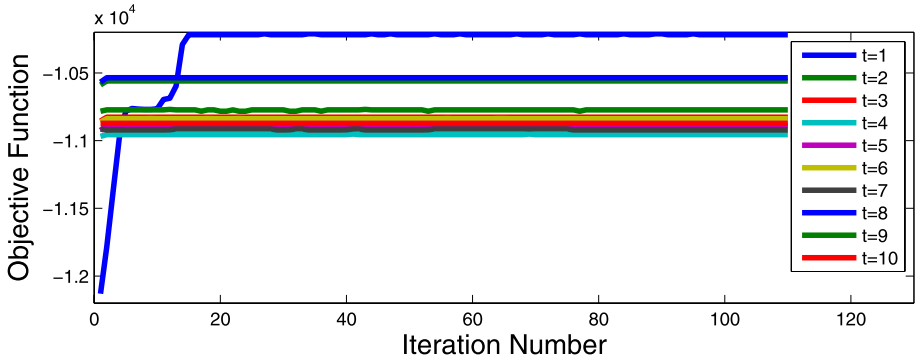


Fig. 8 Convergence rate of Gibbs sampling procedure in the online learning, where the x axis is the iteration number and the y axis represents the value of the objective function

8.3 Experiments on real datasets

In this section we present experimental studies on three real datasets: a traditional social network dataset, a blog dataset, and a paper co-authorship dataset.

8.3.1 The southern women data

The southern women data is a standard benchmark data in social science. It was collected in 1930's in Natchez, Mississippi. The data records the attendance of 18 women in 14 social events during a period of one year. The detailed data is presented in Table 2. We obtain the social network by assigning w_{ij} for women i and j the number of times that they co-participated in the same events. We first apply the static stochastic model (SBM) to the aggregated data and we set the number of communities to be 2, the number used in most previous studies. Not surprisingly, we obtain the same result as most social science methods reported in (Freeman 2003), that is, women 1–9 belong to one community and women 10–18 belong to the other community.

Next, based on the number of events that occurred, we partition the time period into 3 time steps: (1) February–March, when women 1–7,9,10, and 13–18, participated social events 2,5,7, and 11; (2) April–May, when women 8,11,12, and 16 joined in and together they participated in events 3,6,9, and 12; (3) June–November, when events 1,4,8,10, and 13 happened for which women 17 and 18 did not show up. We apply both the offline and the online versions of our algorithm on this dataset with 3 time steps.

It is worth noting that in this dataset, not all individuals showed up in all the three time steps. As a result, when applying our algorithms, we adopted the technique introduced in Sect. 7.2 to handle the insertion of new nodes and deletion of old ones.

The community membership detected by our offline and online algorithms are shown in Table 3. It turns out that the offline algorithm reports no community change for any woman. This result suggests that if we take the overall data into consideration simultaneously, the evidence is not strong enough to justify any change of community membership. However, the online algorithm detects that woman 8 changed her community at timestep 3, as shown in Table 3. By investigating the social events in Table 2 we see that this change is due to the social event 8, which is the only event that woman 8 participated at time step 3. Because woman 8 does not have a long history of community membership (notice that woman 8 has only shown up since timestep 2), we believe this result by our online algorithm makes sense and it actually provides interesting insights into this real-world dataset.

Table 2 The southern women data, from (Freeman 2003)

Names of Participants of Group I	Code Numbers and Dates of Social Events Reported in Old City Herald													
	(1) 6/27	(2) 3/2	(3) 4/12	(4) 9/26	(5) 2/25	(6) 5/19	(7) 3/15	(8) 9/16	(9) 4/8	(10) 6/10	(11) 2/23	(12) 4/7	(13) 11/21	(14) 8/3
1. Mrs. Evelyn Jefferson	×	×	×	×	×	×	...	×	×	...	...	...	...	...
2. Miss Laura Mandeville	×	×	×	...	×	×	×	×	...	...	...	...	...	...
3. Miss Theresa Anderson	...	×	×	×	×	×	×	×	×	...	...	...	...	...
4. Miss Brenda Rogers	×	...	×	×	×	×	×	×	...	...	...	...	...	...
5. Miss Charlotte McDowd	...	...	×	×	×	...	×	...	...	...	...	...	...	...
6. Miss Frances Anderson	...	...	×	...	×	×	...	×	...	...	...	...	...	...
7. Miss Eleanor Nye	...	...	...	...	×	×	×	×	...	...	...	...	...	...
8. Miss Pearl Oglethorpe	...	...	...	...	...	×	...	×	×	...	...	...	...	...
9. Miss Ruth DeSand	...	...	...	...	×	...	×	×	×	...	...	...	...	...
10. Miss Verne Sanderson	...	...	...	...	...	...	×	×	×	...	...	×	...	...
11. Miss Myra Liddell	...	...	...	...	...	...	...	×	×	×	...	×	...	...
12. Miss Katherine Rogers	...	...	...	...	...	...	...	×	×	×	...	×	×	×
13. Mrs. Sylvia Avondale	...	...	...	...	...	...	×	×	×	×	...	×	×	×
14. Mrs. Nora Fayette	...	...	...	...	...	×	×	...	×	×	×	×	×	×
15. Mrs. Helen Lloyd	...	...	...	...	...	...	×	×	...	×	×	×	...	...
16. Mrs. Dorothy Murchison	...	...	...	...	...	...	...	×	×	...	...	...	...	...
17. Mrs. Olivia Carleton	...	...	...	...	...	...	...	...	×	...	×	...	...	...
18. Mrs. Flora Price	...	...	...	...	...	...	...	...	×	...	×	...	...	...

8.3.2 The blog dataset

This blog dataset was collected by the NEC Labs and have been used in several previous studies on dynamic social networks (Chi et al. 2007; Lin et al. 2008). It contains 148,681 entry-to-entry links among 407 blogs during 15 months. In this study, we first partition the data in the following way. The first 7 months are used for the first 7 time steps; data in months 8 and 9 are aggregated into the 8th time step; data in months 10–15 are aggregated into the 9th time step. The reason for this partition is that in the original dataset, the number of links dropped dramatically toward the end of the time and the partition above makes the number of links at each time step to be evenly around 200. Note that in this dataset, instead of binary links, we give weights to the links as the number of entry-to-entry links occurred during a give period. As a result, another purpose of studying this dataset is to demonstrate that our algorithm can handle different types of links.

Clustering performance We run our algorithms on this dataset and compare the performance, in terms of modularity, with that of the two baselines, the dynamic graph-factorization clustering (FacetNet, Lin et al. 2008) and the evolutionary spectral clustering (EvolSpect, Chi et al. 2007). Following (Chi et al. 2007), we set the number of communities to be 2 (which roughly correspond to a technology community and a political community). In terms of hyperparameters for our algorithm, for γ and μ, we simply chose some default values (i.e., $\gamma_k = 1$, $\mu_{kl} = 1$, and $\mu_{kk} = 10$), and for α and β, we chose the ones that result in the best modularity. For the two baseline algorithms, their parameters are chosen to obtain the best modularity. Figure 9 shows the performance of the algorithms. As can be seen, for

Table 3 Communities membership detected by the offline algorithm and the online algorithm, where + and − indicate the two communities

women	offline inference time steps			online inference time steps		
	1	2	3	1	2	3
1	−	−	−	−	−	−
2	−	−	−	−	−	−
3	−	−	−	−	−	−
4	−	−	−	−	−	−
5	−	−	−	−	−	−
6	−	−	−	−	−	−
7	−	−	−	−	−	−
8		−	−		−	+
9	−	−	−	−	−	−
10	+	+	+	+	+	+
11		+	+		+	+
12		+	+		+	+
13	+	+	+	+	+	+
14	+	+	+	+	+	+
15	+	+	+	+	+	+
16		+	+		+	+
17	+	+		+	+	
18	+	+		+	+	

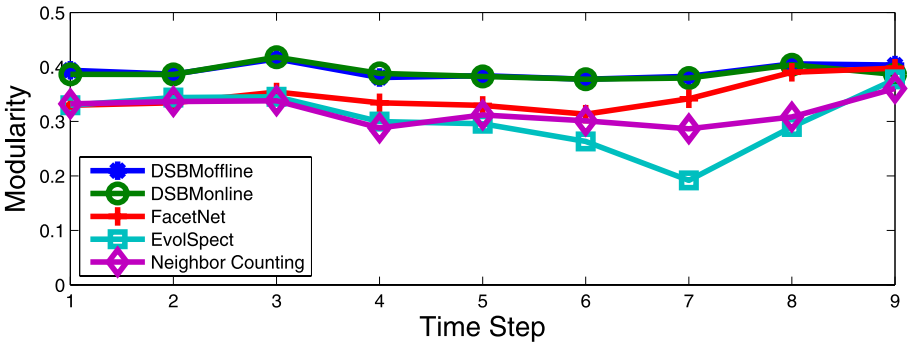


Fig. 9 The performance, in terms of the modularity, of different algorithms (including the naive method using neighbor counting) on the NEC blog datasets, with the community number set to 2

this dataset, the offline and online versions of our algorithm give similar results and they both outperform the baseline algorithms.

Some meaningful community changes Actually, we found that most blogs are stable in terms of their communities. However, there are still some blogs changing their communities detected by our algorithms based on the links information. Here, we present the community memberships of four representative blogs. Three of them (blogs 94, 192, and 357) have the most number of links across the whole time and one of them (blog 230) has the least number of links, only at two time steps. To help the visualization, we assign one of the two labels to

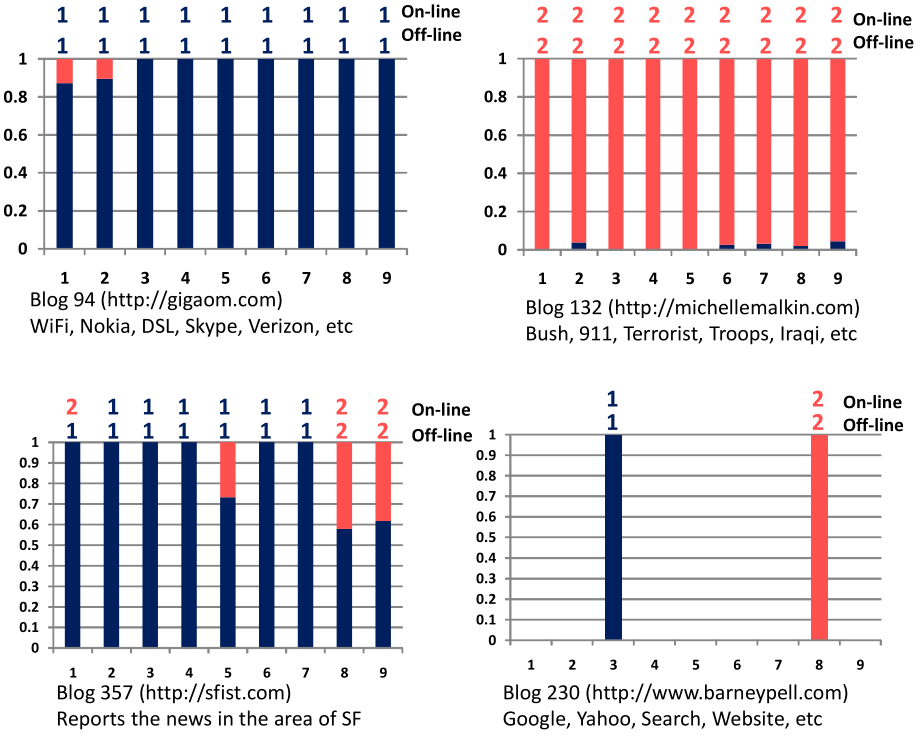


Fig. 10 Neighbor distributions of the four representative blogs (*the color bars in each sub figure*), the community results of the offline and online versions of DSBM (*on the top of each sub figure*), and some top keywords occurred in the blogs (*at the bottom of each sub figure*)

each blog where the labels are obtained by applying the normalized cut algorithm (Shi and Malik 2000) on the aggregated blog graph. Therefore, these labels give us the community membership of each blog if we use *static* analysis on the *aggregated data*. Then to visualize the *dynamic* community memberships, for a blog at a given time step, we show the fractions of the blog's neighbors (through links) that have each of the two possible labels at the given time step.

Figure 10 illustrates how these fractions change over time for these 4 representative blogs. On the top of each subfigure, we show the community memberships computed by our algorithms and at the bottom of each subfigure, we show some top keywords that occurred most frequently in the corresponding blog site. From the figure we can see that blogs 94 and 132, for which our algorithms detect no changes in community memberships, have very homogeneous neighbor labels and very focused top keywords. In comparison, blog 357 has relatively inhomogeneous neighbors during different time steps and has more changes of community memberships detected by our algorithm. It turns out that blog 357 is a blog that reports news events in the area of San Francisco, including both technology events and political events. Finally, we look at blog 230. This blog is more technology focused. However, during the whole time period, it only generated 2 entry-to-entry links in time steps 3 and 8 respectively. The link generated at time 3 pointed to a political blog (blog 60, http://catallarchy.net) and that at time 8 pointed to a technology blog (blog 362, http://www.siliconbeat.com). Although the results obtained by our algorithms are correct

based on these evidence, we can see that the link-based analysis can be unreliable when the links are very sparse.

Now seeing that the simple neighbor counting approach gives results consistent with our algorithms, one may wonder if such a naive approach is good enough for analyzing dynamic social networks. To answer this question, in Fig. 9 we also plot the performance of this neighbor counting approach (as diamond-shaped points on the magenta-colored line) and we can see that such a naive approach by itself does not give performances comparable to our algorithms.

8.3.3 The paper co-authorship (DBLP) data

This data was extracted from DBLP and has been studied in (Asur et al. 2007; Lin et al. 2008). It contains the co-authorship information among the papers in 28 conferences over 10 years (1997–2006). The 28 conferences span three main areas—data mining (DM), database (DB), and artificial intelligence (AI). The nodes in the networks are the authors and the link weights are the number of papers co-authored by a pair of authors. Since the links in each time step are sparse, we aggregate the 10 years into 5 time steps with 2 years each. We apply our algorithm to this dataset with the known community number 3.

We first delineate the community evolution during the 5 time steps shown in Fig. 11. To avid clutter, we only show the most significant evolutions (in terms of the number of authors that made the transitions), which turn out to be mostly from DB to DM communities and from DB to AI communities. From the figure, we can observe the trend that the community of DB gets smaller and the community of DM gets larger along the time, while the size of community AI remains relatively stable. This trend actually matches our knowledge about the period (1997–2006) of the data. Besides the community evolution, we also find some meaningful community evolution for individual researchers. Here we only list four authors who have large number of publications and on whom our algorithms detect a lot of changes in community memberships. These authors include Soumen Chakrabarti (from DB to AI), Laks V. S. Lakshmanan (from DM to DB), Christos Faloutsos (from DB to DM), and Byron Dom (from DB to AI). By checking the conference venues of the papers published by these authors each year and by checking the biographies of these authors, we verified that the above changes all correspond to switches of research focus that really happened.

It is worth mentioning that the conference venues (and therefore the class labels for all the conferences and all the papers) are not used in our algorithms. This implies that by only studying the interactions among individuals (the co-authorship), our algorithms can discover meaningful changes of community memberships that are related to real-world events.

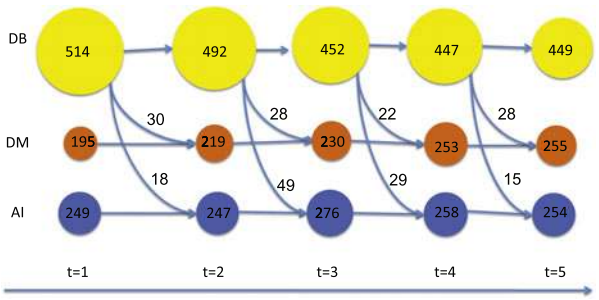


Fig. 11 Community evolutions for DBLP data set

9 Conclusion

In this paper, we proposed a framework based on Bayesian inference to find communities and to capture community evolution in dynamic social networks. The framework is a probabilistic generative model that unifies the communities and their evolution in an intuitive and rigorous way; the Bayesian treatment gives robust prediction of community memberships; the proposed algorithms are implemented efficiently to make them practical in real applications. Extensive experimental studies showed that our algorithms outperform several state-of-the-art baseline algorithms in different measures and reveal very useful insights in several real social networks.

The current Bayesian framework relies solely on the links to infer the community memberships of nodes in social networks. This may be insufficient when the number of links is sparse. In the future, we plan to extend our framework to incorporate information other than links such as the contents of blogs.

Acknowledgements We thank Kai Yu for providing us with the SGFC source code and for the helpful discussion; thank Yu-Ru Lin for providing us with the FacetNet source code; thank Sitaram Asur and Srinivasan Parthasarathy for providing us with the DBLP dataset; and thank Junichi Tatemura and Koji Hino for helping prepare the blog dataset.

Appendix

In this appendix, we provide detailed proof for Theorem 1, 2.

Proof of Theorem 1 The joint probability $\Pr(\mathcal{W}_T, \mathcal{Z}_T)$ is computed by integrating over the parameters $\theta = \{\pi, P, A\}$ under the conjugate prior of $\theta = \{\pi, P, A\}$, i.e.,

$$\Pr(\mathcal{W}_T, \mathcal{Z}_T) = \int_{\pi,P,A} \Pr(\mathcal{W}_T, \mathcal{Z}_T|\pi, P, A) \Pr(\pi) \Pr(P) \Pr(A) d\pi\, dP\, dA.$$

The key to this computation is that we can factorize the probability $\Pr(\mathcal{W}_T, \mathcal{Z}_T|\pi, P, A)$ into three parts that depend on π, P, A, respectively as in (1), i.e.,

$$\Pr(\mathcal{W}_T, \mathcal{Z}_T|\pi, P, A) = \prod_{t=1}^{T} \Pr(W^{(t)}|Z^{(t)}, P) \prod_{t=2}^{T} \Pr(Z^{(t)}|Z^{(t-1)}, A) \Pr(Z^{(1)}|\pi).$$

Then the integral is taken independently,

$$\Pr(\mathcal{W}_T, \mathcal{Z}_T) = \int_{\pi} \Pr(Z^{(1)}|\pi) \Pr(\pi) d\pi \int_{P} \prod_{t=1}^{T} \Pr(W^{(t)}|Z^{(t)}, P) \Pr(P) dP$$
$$\times \int_{A} \prod_{t=2}^{T} \Pr(Z^{(t)}|Z^{(t-1)}, A) \Pr(A) dA.$$

As the prior for π, P, A is assumed to be conjugate to the corresponding likelihood term, the integral can be easily computed.

For π, the integral is

$$\begin{aligned}\int_{\pi} \Pr(Z^{(1)}|\pi)\Pr(\pi)d\pi &= \int_{\pi} \prod_{i,k} \pi_k^{z_{ik}^{(1)}} \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \prod_k \pi_k^{\gamma_k - 1} d\pi \\ &= \int_{\pi} \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \prod_k \pi_k^{\sum_i z_{ik}^{(1)} + \gamma_k - 1} \\ &= \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \times \frac{\prod_k \Gamma(\sum_i z_{ik}^{(1)} + \gamma_k)}{\Gamma(\sum_k \sum_i z_{ik}^{(1)} + \gamma_k)} \\ &= \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \times \frac{\prod_k \Gamma(n_k^{(1)} + \gamma_k)}{\Gamma(n + \sum_k \gamma_k)}.\end{aligned}$$

In the above computation, we use the property of Dirichlet distribution and the definition introduced in (6). For P, the integral is

$$\begin{aligned}&\int_P \prod_{t=1}^{T} \Pr(W^{(t)}|Z^{(t)}, P)\Pr(P)dP \\ &= \int_P \prod_{i=1}^{T} \prod_{i\sim j} \prod_{k,l} \left(P_{kl}^{w_{ij}^{(t)}} (1 - P_{ij})^{1 - w_{ij}^{(t)}} \right)^{z_{ik}^{(t)} z_{jl}^{(t)}} \\ &\quad \times \prod_{k,l \geq l} \frac{1}{B(\alpha_{kl}, \beta_{kl})} P_{kl}^{\alpha_{kl}-1} (1 - P_{kl})^{\beta_{kl}-1} dP \\ &= \int_P \prod_{k,l>k} P_{kl}^{\sum_{t=1}^{T} \sum_{i\sim j} w_{ij}^{(t)} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)})} (1 - P_{kl})^{\sum_{t=1}^{T} \sum_{i\sim j} (1 - w_{ij}^{(t)})(z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)})} \\ &\quad \times \prod_{k,l>k} \frac{1}{B(\alpha_{kl}, \beta_{kl})} P_{kl}^{\alpha_{kl}-1} (1 - P_{kl})^{\beta_{kl}-1} \\ &\quad \times \prod_{k,l=k} P_{kl}^{\frac{1}{2} \sum_{t=1}^{T} \sum_{i\sim j} w_{ij}^{(t)} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)})} (1 - P_{kl})^{\frac{1}{2} \sum_{t=1}^{T} \sum_{i\sim j} (1 - w_{ij}^{(t)})(z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)})} \\ &\quad \times \prod_{k,l=k} \frac{1}{B(\alpha_{kl}, \beta_{kl})} P_{kl}^{\alpha_{kl}-1} (1 - P_{kl})^{\beta_{kl}-1} dP \\ &= \prod_{k,l>k} \frac{1}{B(\alpha_{kl}, \beta_{kl})} B\left(\sum_{t=1}^{T} \sum_{i\sim j} w_{ij}^{(t)} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}), \right. \\ &\qquad \left. \sum_{t=1}^{T} \sum_{i\sim j} (1 - w_{ij}^{(t)})(z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}) \right)\end{aligned}$$

$$\times \prod_{k,l=k} \frac{1}{B(\alpha_{kl},\beta_{kl})} B\left(\frac{1}{2}\sum_{t=1}^{T}\sum_{i\sim j} w_{ij}^{(t)}(z_{ik}^{(t)}z_{jl}^{(t)}+z_{il}^{(t)}z_{jk}^{(t)}),\right.$$

$$\left.\frac{1}{2}\sum_{t=1}^{T}\sum_{i\sim j}(1-w_{ij}^{(t)})(z_{ik}^{(t)}z_{jl}^{(t)}+z_{il}^{(t)}z_{jk}^{(t)})\right)$$

$$= \prod_{k,l>k} \frac{1}{B(\alpha_{kl},\beta_{kl})} B(\hat{n}_{kl}^{(1:T)}+\alpha_{kl}, n_{kl}^{(1:T)}-\hat{n}_{kl}^{(1:T)}+\beta_{kl})$$

$$\times \prod_{k} \frac{1}{B(\alpha_{kk},\beta_{kk})} B\left(\frac{\hat{n}_{kl}^{(1:T)}}{2}+\alpha_{kl}, \frac{n_{kl}^{(1:T)}-\hat{n}_{kl}^{(1:T)}}{2}+\beta_{kl}\right).$$

In the above computation, we use the property of beta distribution, the symmetry of P_{kl}, and the definitions introduced in (9)–(10).

For A, the integral is

$$\int_A \prod_{t=2}^{T} \Pr(Z^{(t)}|Z^{(t-1)},A)\Pr(A)dA$$

$$= \int_A \prod_{t=2}^{T}\prod_{i=1}^{n}\prod_{k,l} A_{kl}^{z_{ik}^{(t-1)}z_{il}^{(t)}} \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \prod_l A_{kl}^{\mu_{kl}-1}$$

$$= \int_A \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \prod_l A_{kl}^{\sum_{t=2}^{T}\sum_{i=1}^{n} z_{ik}^{(t-1)}z_{il}^{(t)}+\mu_{kl}-1}$$

$$= \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \frac{\prod_l \Gamma(\sum_{t=2}^{T}\sum_{i=1}^{n} z_{ik}^{(t-1)}z_{il}^{(t)}+\mu_{kl})}{\Gamma(\sum_{t=2}^{T}\sum_{i=1}^{n}\sum_l z_{ik}^{(t-1)}z_{il}^{(t)}+\sum_l \mu_{kl})}$$

$$= \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \frac{\prod_l \Gamma(n_{k\to l}^{(1:T)}+\mu_{kl})}{\Gamma(n_{k\to\cdot}^{(1:T)}+\sum_l \mu_{kl})}.$$

In the above computation, we use the property of Dirichlet distribution and the definitions introduced in (7), (8).

Finally, coming the three integrals and ignore the terms that are independent of $\mathcal{W}_T$, $\mathcal{Z}_T$, we have

$$\Pr(\mathcal{W}_T,\mathcal{Z}_T) \propto \prod_k \Gamma(n_k^{(1)}+\gamma_k) \prod_{k,l>k} B(\hat{n}_{kl}^{(1:T)}+\alpha_{kl}, n_{kl}^{(1:T)}-\hat{n}_{kl}^{(1:T)}+\beta_{kl})$$

$$\times \prod_k B\left(\frac{\hat{n}_{kl}^{(1:T)}}{2}+\alpha_{kl}, \frac{n_{kl}^{(1:T)}-\hat{n}_{kl}^{(1:T)}}{2}+\beta_{kl}\right) \prod_k \frac{\prod_l \Gamma(n_{k\to l}^{(1:T)}+\mu_{kl})}{\Gamma(n_{k\to\cdot}^{(1:T)}+\sum_l \mu_{kl})}.$$

This completes the proof. □

Proof of Theorem 2 To compute $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$, we note that

$$\begin{aligned}
&\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \\
&\quad \propto \Pr(W^{(t)}, Z^{(t)}, Z^{(t-1)}) \\
&\quad = \Pr(W^{(t)}|Z^{(t)}) \Pr(Z^{(t)}|Z^{(t-1)}) \\
&\quad = \int_P \Pr(W^{(t)}|Z^{(t)}, P) \Pr(P) dP \int_A \Pr(Z^{(t)}|Z^{(t-1)}, A) \Pr(A) dA.
\end{aligned}$$

With similar computations as in proof of Theorem 1, the two integrals results in

$$\begin{aligned}
\Pr(W^{(t)}|Z^{(t)}) &\propto \prod_{k,l>k} B(\hat{n}_{kl}^{(t)} + \alpha_{kl}, n_{kl}^{(t)} - \hat{n}_{kl}^{(t)} + \beta_{kl}) \\
&\quad \times \prod_k B\left(\frac{\hat{n}_{kl}^{(t)}}{2} + \alpha_{kl}, \frac{n_{kl}^{(t)} - \hat{n}_{kl}^{(t)}}{2} + \beta_{kl}\right), \\
\Pr(Z^{(t)}|Z^{(t-1)}) &\propto \prod_k \frac{\prod_l \Gamma(n_{k\to l}^{(t-1:t)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(t-1:t)} + \sum_l \mu_{kl})}.
\end{aligned}$$

Multiplying these two terms, we get the one for $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$ as given in Theorem 2.

To compute $\Pr(Z^{(1)}|W^{(1)})$, we note

$$\Pr(Z^{(1)}|W^{(1)}) \propto \Pr(W^{(1)}|Z^{(1)}) \Pr(Z^{(1)}).$$

The first term $\Pr(W^{(1)}|Z^{(1)})$ is computed as above with t replaced with 1. For $\Pr(Z^1)$, we have already computed in proof of Theorem 1, i.e.,

$$\Pr(Z^{(1)}) = \int_\pi \Pr(Z^{(1)}|\pi) \Pr(\pi) d\pi \propto \prod_k \Gamma(n_k^{(1)} + \gamma_k).$$

Multiplying these two terms, we get the one for $\Pr(Z^{(1)}|W^{(1)})$ as given in Theorem 2. □

References

Ahmed, A., & Xing, E. P. (2008). Dynamic non-parametric mixture models and the recurrent Chinese restaurant process: with applications to evolutionary clustering. In *SDM* (pp. 219–230). Philadelphia: SIAM.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., & Xing, E. P. (2006). Mixed membership stochastic block models for relational data with application to protein-protein interactions. In *Proceedings of the international biometrics society annual meeting*.

Asur, S., Parthasarathy, S., & Ucar, D. (2007). An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *Proceedings of the 13th ACM SIGKDD conference*.

Bishop, C. M. (2006). *Pattern recognition and machine learning (information science and statistics)*. New York: Springer.

Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z., & Wagner, D. (2008). On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, *20*(2).

Chakrabarti, D., Kumar, R., & Tomkins, A. (2006). Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD conference*.

Chen, J., Zaiane, O. R., & Goebel, R. (2009). Detecting communities in social networks using max-min modularity. In *SDM'09: proceedings of the 9th SIAM international conference on data mining*.

Chi, Y., Song, X., Zhou, D., Hino, K., & Tseng, B. L. (2007). Evolutionary spectral clustering by incorporating temporal smoothness. In *Proceedings of the 13th ACM SIGKDD conference*.

Chung, F. R. K. (1997). *Spectral graph theory*. Providence: American Mathematical Society.
Fienberg, S. E., Meyer, M. M., & Wasserman, S. S. (1985). Statistical analysis of multiple sociometric relations. *Journal of the American Statistical Association*, *80*(389).
Flake, G., Lawrence, S., & Giles, C. (2000). Efficient identification of web communities. In *Proceedings of the 6th ACM SIGKDD conference*.
Freeman, L. C. (2003). *Finding social groups: a meta-analysis of the southern women data*. New York: National Academies Press.
Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*.
Gong, Y., & Xu, W. (2007). *Machine learning for multimedia content analysis*. Berlin: Springer.
Griffiths, T. L., & Steyvers, M. (2004). Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, *101*(Suppl. 1), 5228–5235.
Ho, P. D., Raftery, A. E., & H. M. S. (2002). Statistical analysis of multiple sociometric relations. *Latent Space Approaches to Social Network Analysis*, *97*.
Hofman, J. M., & Wiggins, C. H. (2008). A Bayesian approach to network modularity. *Physical Review Letters*, *100*.
Holland, P., & Leinhardt, S. (1976). Local structure in social networks. *Sociological Methodology*.
Kemp, C., Griffiths, T. L., & Tenenbaum, J. B. (2004). In *Discovering latent classes in relational data* (Tech. Rep. AI Memo 2004-019). MIT, Computer Science and Artificial Intelligence Laboratory.
Kim, M. S., & Han, J. (2009). A particle-and-density based evolutionary clustering method for dynamic networks. *Proceedings VLDB Endowment*, *2*(1), 622–633.
Kumar, R., Novak, J., Raghavan, P., & Tomkins, A. (2003). On the bursty evolution of blogspace. In *Proceedings of the 12th WWW conference*.
Leskovec, J., Kleinberg, J., & Faloutsos, C. (2005). Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the 11th ACM SIGKDD conference*.
Lin, Y. R., Chi, Y., Zhu, S., Sundaram, H., & Tseng, B. L. (2008). FacetNet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proceedings of the 17th WWW conference*.
Lin, Y. R., Chi, Y., Zhu, S., Sundaram, H., & Tseng, B. L. (2009a). Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data*, *3*(2).
Lin, Y. R., Sun, J., Castro, P., Konuru, R., Sundaram, H., & Kelliher, A. (2009b). Metafac: community discovery via relational hypergraph factorization. In *KDD '09: proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 527–536). New York: ACM. doi:10.1145/1557019.1557080.
Mei, Q., & Zhai, C. (2005). Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proceedings of the 11th ACM SIGKDD conference*.
Newman, M. E. J. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*
Newman, M. E. J., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*.
Palla, G., Barabasi, A. L., & Vicsek, T. (2007). Quantifying social group evolution. *Nature*, *446*.
Sarkar, P., & Moore, A. W. (2005). Dynamic social network analysis using latent space models. *SIGKDD Exploration Newsletter*, *7*(2).
Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8).
Shortreed, S., Handcock, M. S., & Hoff, P. (2006). A particle-and-density based evolutionary clustering method for dynamic networks. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, *2*(1), 24–33.
Snijders, T. A. B. (2002). Markov chain Monte Carlo estimation of exponential random graph models. *Journal of Social Structure*, *3*.
Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y., & Schult, R. (2006). Monic: modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD conference*.
Sun, J., Faloutsos, C., Papadimitriou, S., & Yu, P. S. (2007). GraphScope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th SIGKDD conference*.
Tang, L., Liu, H., Zhang, J., & Nazeri, Z. (2008). Community evolution in dynamic multi-mode networks. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 677–685). New York: ACM. doi:10.1145/1401890.1401972.
Tantipathananandh, C., Berger-Wolf, T., & Kempe, D. (2007). A framework for community identification in dynamic social networks. In *Proceedings of the 13th ACM SIGKDD conference*.
Toyoda, M., & Kitsuregawa, M. (2003). Extracting evolution of web communities from a series of web archives. In *HYPERTEXT '03: proceedings of the 14th ACM conference on hypertext and hypermedia*.

Wasserman, S., & Faust, K. (1994). *Social network analysis: methods and applications*. Cambridge: Cambridge University Press.

Wasserman, S., & Pattison, P. (1996). Logit models and logistic regressions for social networks, I: an introduction to Markov graphs and p*. *Psychometrika*, *60*.

White, S., & Smyth, P. (2005). A spectral clustering approach to finding communities in graph. In *SDM*.

Xu, W., & Gong, Y. (2004). Document clustering by concept factorization. In *SIGIR* (pp. 202–209).

Yang, T., Chi, Y., Zhu, S., Gong, Y., & Jin, R. (2009). A Bayesian approach toward finding communities and their evolutions in dynamic social networks. In *SDM'09: proceedings of the 2009 SIAM international conference on data mining* (pp. 990–1001).

Yu, K., Yu, S., & Tresp, V. (2005). Soft clustering on graphs. In *NIPS*.

Zhu, X. (2005). *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University.