Soc. Netw. Anal. Min. (2016) 6:55
DOI 10.1007/s13278-016-0368-3

ORIGINAL ARTICLE

Block modelling in dynamic networks with non-homogeneous Poisson processes and exact ICL

Marco Corneli[1] · Pierre Latouche[1] · Fabrice Rossi[1]

Received: 18 December 2015 / Revised: 22 July 2016 / Accepted: 24 July 2016 / Published online: 2 August 2016

Abstract We develop a model in which interactions between nodes of a dynamic network are counted by non-homogeneous Poisson processes. In a block modelling perspective, nodes belong to hidden clusters (whose number is unknown) and the intensity functions of the counting processes only depend on the clusters of nodes. In order to make inference tractable, we move to discrete time by partitioning the entire time horizon in which interactions are observed in fixed-length time sub-intervals. First, we derive an exact integrated classification likelihood criterion and maximize it relying on a greedy search approach. This allows to estimate the memberships to clusters and the number of clusters simultaneously. Then, a maximum likelihood estimator is developed to estimate nonparametrically the integrated intensities. We discuss the over-fitting problems of the model and propose a regularized version solving these issues. Experiments on real and simulated data are carried out in order to assess the proposed methodology.

Keywords Dynamic network · Stochastic block model · Exact ICL · Non-homogeneous Poisson process

1 Introduction

Graph clustering (Schaeffer 2007) is probably one of the main exploratory tools used in network analysis as it provides data analysts with a high level summarized view of complex networks. One of the main paradigms for graph clustering is community search (Fortunato 2010): a *community* is a subset of nodes in a graph that are densely connected and have relatively few connections to nodes outside of the community. While this paradigm is very successful in many applications, it suffers from a main limitation: it cannot be used to detect other important structures that arise in graphs, such as bipartite structures, hubs, authorities and other patterns.

The alternative solution favoured in this paper is provided by block models (Lorrain and White 1971; White et al. 1976): in such a model, a cluster consists of nodes that share the same connectivity patterns to other clusters, regardless of the pattern itself (community, hub, bipartite, etc.). A popular probabilistic view on block models is provided by the stochastic block model (SBM, Holland et al. 1983; Wang and Wong 1987). The main idea is to assume that a hidden random variable is attached to each node. This variable contains the cluster membership information while connection probabilities between clusters are handled by the parameters of the model. The reader is send to Goldenberg et al. (2009) for a survey of probabilistic models for graphs and to Wasserman and Faust (1994), Ch.16, for an overview of the stochastic block models.

This paper focuses on dynamic graphs in the following sense: we assume that nodes of the graph are fixed and that interactions between them are directed and take place at a specific instant. In other words, we consider a directed multigraph (two nodes can be connected by more than one edge) in which each directed edge is labelled with an occurrence time. We are interested in extending the SBM to this type of graphs. More precisely, the proposed model is based on a counting process point of view of the interactions between nodes: we assume that the number of

✉ Marco Corneli
marcogenni@gmail.com

[1] Laboratoire SAMM, Université Paris 1 Panthéon-Sorbonne, 90 rue de Tolbiac, 75634 Paris Cedex 13, France

interactions between two nodes follows a non-homogeneous Poisson counting process (NHPP). As in a standard SBM, nodes are assumed to belong to clusters that do not change over time; thus, the temporal aspect is handled only via the non-homogeneity of the counting processes. Then, the block model hypothesis takes the following form: the intensity of the NHPP that counts interactions between two nodes depends only on the clusters of the nodes. In order to obtain a tractable inference, a segmentation of the time interval under study is introduced and the interactions are aggregated over the sub-intervals of the partition. Following Côme and Latouche (2015), the model is adjusted to the data via the maximization of the integrated classification likelihood (ICL Biernacki et al. 2000) in an exact form. As in Côme and Latouche (2015) (and Wyse et al. (2014) for latent block models), the maximization is done via a greedy search. This allows us to choose automatically the number of clusters in the block model.

When the number of sub-intervals is large, the model can suffer from a form of over-fitting as the ICL penalizes only a large number of clusters. Therefore, we introduce a variant, based on the model developed in Corneli et al. (2015), in which sub-intervals are clustered into classes of homogeneous intensities. Those clusters are accounted for in a new version of the ICL which prevents over-fitting.

This paper is structured as follows: in Sect. 2, we mention works related to the approach we propose, Sect. 3 presents the proposed temporal extension of the SBM and Sect. 4 derives the exact ICL for this model and presents the greedy search algorithm used to maximize the ICL. Section 5 gathers experimental results on simulated data and on real-world data. Section 6 concludes the paper.

2 Related works

Numerous extensions of the original SBM have already been proposed to deal with dynamic graphs. In this context, both nodes memberships to a cluster and interactions between nodes can be seen as stochastic processes. In Yang et al. (2011), for instance, authors introduce a Markov Chain to obtain the cluster of node in t given its cluster at time $t-1$. Xu and Hero (2013) as well as Xing et al. (2010) used a state space model to describe temporal changes at the level of the connectivity pattern. In the latter, the authors developed a method to retrieve overlapping clusters through time. In general, the proposed temporal variations of the SBM share a similar approach: the data set consists in a sequence of graphs rather than the more general structure we assume. Some papers remove those assumptions by considering continuous time models in which edges occur at specific instants (for instance, when someone sends an email). This is the case of e.g. Dubois et al. (2013) and of Guigourès et al. (2012, 2015). A temporal stochastic block model, related to the one presented in this paper, is independently developed by Matias et al. (2015). They assume that nodes in a network belong to clusters whose composition do not change over time and interactions are counted by a non-homogeneous Poisson process whose intensity only depends on the nodes clusters. In order to estimate (nonparametrically) the instantaneous intensity functions of the Poisson processes, they develop a variational EM algorithm to maximize an approximation of the likelihood.

3 The model

We consider a fixed set of N nodes, $\{1,\ldots,N\}$, that can interact as frequently as wanted during the time interval $[0, T]$. Interactions are directed from one node to another and are assumed to be instantaneous.[1] A natural mathematical model for this type of interactions is provided by counting processes on $[0, T]$. Indeed a counting process is a stochastic process with values that are non-negative integers increasing through time: the value at time t can be seen as the number of interactions that took place from 0 to t. Then, the classical adjacency matrix $(X_{ij})_{1\leq i,j\leq N}$ of static graphs is replaced by a $N\times N$ collection of counting processes, $(X_{ij}(t))_{1\leq i,j\leq N}$, where $X_{ij}(t)$ is the counting process that gives the number of interactions from node i to node j. We still call $\mathbf{X}=(X_{ij}(t))_{1\leq i,j\leq N}$ the adjacency matrix of this dynamical graph.

We introduce in this section a generative model for adjacency matrices of dynamical graphs that is inspired by the classical stochastic block model (SBM).

3.1 Non-homogeneous Poisson counting process

We first chose a simple form for $X_{ij}(t)$: we assume that this process is a non-homogeneous Poisson counting process (NHPP) with instantaneous intensity given by the function from $[0, T]$ to $\mathbb{R}$, λ_{ij}. For $s\leq t\leq T$, it then holds

$$p(X_{ij}(t)-X_{ij}(s)|\lambda_{ij})=\frac{\left(\int_s^t\lambda_{ij}(u)\mathrm{d}u\right)^{X_{ij}(t)-X_{ij}(s)}}{(X_{ij}(t)-X_{ij}(s))!}\exp\left(-\int_s^t\lambda_{ij}(u)\mathrm{d}u\right),\tag{1}$$

where $X_{ij}(t)-X_{ij}(s)$ is the (non-negative) number of interactions from i to j that took place during $[s, t]$. (We assume that $X_{ij}(0)=0$.)

[1] In practice, the starting time of an interaction with a duration will be considered.

3.2 Block modelling

The main idea of the SBM (Holland et al. 1983; Wang and Wong 1987) is to assume that nodes have some (hidden) characteristics that solely explain their interactions, in a stochastic sense. In our context, this means that rather than having pairwise intensity functions λ_{ij}, those functions are shared by nodes that have the same characteristics.

In more technical terms, we assume the nodes are grouped in K clusters $(\mathcal{A}_1, \ldots, \mathcal{A}_K)$ and introduce a hidden cluster membership random vector $\mathbf{z} \in \{1, \ldots K\}^N$ such that

$$z_i = k \quad \text{iff} \quad i \in \mathcal{A}_k, \quad k \leq K.$$

The random component z_i is assumed to follow a multinomial distribution with parameter vector ω such that

$$\mathbb{P}\{z_i = k\} = \omega_k \quad \text{with} \quad \sum_{k \leq K} \omega_k = 1.$$

In addition, the $(z_i)_{1 \leq i \leq N}$ are assumed to be independent (knowing ω) and thus

$$p(\mathbf{z}|\boldsymbol{\omega}, K) = \prod_{k \leq K} \omega_k^{|\mathcal{A}_k|}, \tag{2}$$

where $|\mathcal{A}_k|$ denotes the cardinal of $\mathcal{A}_k$. Notice that this part of the model is exactly identical to what is done in the classical SBM.

In a second step, we assume that given $\mathbf{z}$, the counting processes $X_{ij}(t)$ are independent and in addition that the intensity function λ_{ij} depends only on z_i and z_j. In order to keep notations tight, we denote $\lambda_{z_i z_j}$ the common intensity function and we will not use directly the pairwise intensity functions λ_{ij}. We denote $\boldsymbol{\lambda}$ the matrix valued intensity function $\boldsymbol{\lambda} = (\lambda_{kg}(t))_{1 \leq k,g \leq K}$.

Combining all the assumptions, we have for $s \leq t \leq T$

$$p(\mathbf{X}(t) - \mathbf{X}(s)|\mathbf{z}, \boldsymbol{\lambda}) = \prod_{i \neq j} \frac{\left(\int_s^t \lambda_{z_i z_j}(u)\mathrm{d}u\right)^{X_{ij}(t) - X_{ij}(s)}}{(X_{ij}(t) - X_{ij}(s))!} \times \exp\left(-\int_s^t \lambda_{z_i z_j}(u)\mathrm{d}u\right). \tag{3}$$

3.3 Discrete time version

In order to make inference tractable, we move from the continuous time model to a discrete time one. This is done via a partition of the interval [0, T] based on a set of $U + 1$ instants

$$0 = t_0 \leq t_1 \leq \cdots \leq t_{U-1} \leq t_U = T,$$

that defines U intervals $I_u := [t_{u-1}, t_u[$ (with arbitrary length Δ_u). The purpose of the partition is to aggregate the interactions. Let us denote

$$Y_{ij}^{I_u} := X_{ij}(t_u) - X_{ij}(t_{u-1}), \quad u \in \{1, \ldots, U\}. \tag{4}$$

In words, $Y_{ij}^{I_u}$ measures the increment, over the time interval I_u, of the Poisson process counting interactions from i to j. We denote by Y_{ij} the random vector

$$Y_{ij} := \left(Y_{ij}^{I_1}, \ldots, Y_{ij}^{I_U}\right)^T.$$

Thanks to the independence of the increments of a Poisson process, we get the following joint density:

$$p(Y_{ij}|\lambda_{ij}) = \prod_{u=1}^{U} \left(\frac{\left(\int_{I_u} \lambda_{ij}(s)\mathrm{d}s\right)^{Y_{ij}^{I_u}}}{Y_{ij}^{I_u}!} \exp\left(-\int_{I_u} \lambda_{ij}(s)\mathrm{d}s\right)\right). \tag{5}$$

The variations of λ_{ij} inside an interval I_u have no effect on the distribution of Y_{ij}. This allows us to use the integrated intensity function Λ defined on [0, T] by

$$\Lambda_{ij}(t) := \int_0^t \lambda_{ij}(s)ds.$$

In addition, we denote by $\pi_{ij}^{I_u}$ the increment of the integrated intensity function over I_u

$$\pi_{ij}^{I_u} := \Lambda_{ij}(t_u) - \Lambda_{ij}(t_{u-1}), \qquad \forall u \in \{1, \ldots, U\}.$$

Then, Eq. (5) becomes

$$p(Y_{ij}|\pi_{ij}) = \prod_{u=1}^{U} \left(\frac{\left(\pi_{ij}^{I_u}\right)^{Y_{ij}^{I_u}}}{Y_{ij}^{I_u}!} \exp\left(-\pi_{ij}^{I_u}\right)\right), \tag{6}$$

with $\pi_{ij} := (\pi_{ij}^{I_1}, \ldots, \pi_{ij}^{I_U})^T$.

Using the block model assumptions, we have in addition

$$p(Y_{ij}|\pi_{z_i z_j}, z_i, z_j) = \prod_{u=1}^{U} \left(\frac{\left(\pi_{z_i z_j}^{I_u}\right)^{Y_{ij}^{I_u}}}{Y_{ij}^{I_u}!} \exp\left(-\pi_{z_i z_j}^{I_u}\right)\right), \tag{7}$$

where we have used the fact that $\lambda_{ij} = \lambda_{z_i z_j}$ (which leads to $\Lambda_{ij} = \Lambda_{z_i z_j}$, etc.).

Considering the network as a whole, we can introduce two tensors of order 3. Y is a $N \times N \times U$ random tensor whose element (i, j, u) is the random variable $Y_{ij}^{I_u}$ and π is the $K \times K \times U$ tensor whose element (k, g, u) is $\pi_{kg}^{I_u}$. Y can be seen as an aggregated (or discrete time version) of the adjacency process $\mathbf{X}$ while π can be seen as summary of the matrix valued intensity function $\boldsymbol{\lambda}$.

The conditional independence assumption of the block model leads to

$$p(Y|\pi, \mathbf{z}) = \prod_{i,j}^{N} p(Y_{ij}|\pi_{z_i z_j}, z_i, z_j). \tag{8}$$

To simplify the rest of the paper, we will use the following notations

$$\prod_{i,j}\prod_{k,g}\prod_{u} := \prod_{i=1}^{N}\prod_{j=1}^{N}\prod_{k=1}^{K}\prod_{g=1}^{K}\prod_{u=1}^{U}$$

$$\prod_{z_i=k}\left(\prod_{z_j=g}\right) := \prod_{\substack{i:\\ z_i=k}}\left(\prod_{\substack{j:\\ z_j=g}}\right).$$

The joint distribution of Y, given $\mathbf{z}$ and π, is

$$\begin{aligned} p(Y|\mathbf{z},\pi) &= \prod_{i,j}\prod_{u}\left(\frac{\left(\pi^{I_u}_{z_iz_j}\right)^{Y^{I_u}_{ij}}}{Y^{I_u}_{ij}!}\exp\left(-\pi^{I_u}_{z_iz_j}\right)\right) \\ &= \prod_{k,g}\prod_{u}\left(\frac{\left(\pi^{I_u}_{k,g}\right)^{S_{kgu}}}{P_{kgu}}\exp\left(-|A_k||A_g|\pi^{I_u}_{kg}\right)\right), \end{aligned} \tag{9}$$

where

$$S_{kgu} = \sum_{z_i=k}\sum_{z_j=g} Y^{I_u}_{ij},$$

is the total number of interactions from cluster k to cluster g (possibly equal to k) and with

$$P_{kgu} = \prod_{z_i=k}\prod_{z_j=g} Y^{I_u}_{ij}!.$$

3.4 A constrained version

As will be shown in Sect. 4.4, the model presented thus far is prone to over-fitting when the number of sub-intervals U is large compared to N. Additional constraints on the intensity functions $\{\Lambda_{kg}(t)\}_{k,g\leq K}$ are needed in this situation.

Let us consider a fixed pair of clusters (k, g). So far, the increments $\{\pi^{I_u}_{kg}\}_{u\leq U}$ are allowed to differ on each I_u over the considered partition. A constraint can be introduced by assigning the time intervals $(I_1, \ldots I_U)$ to different time clusters and assuming that increments are identical for all the intervals belonging to the same time cluster. Formally, we introduce D clusters $(\mathcal{C}_1, \ldots, \mathcal{C}_D)$ and a hidden random vector $\mathbf{y} \in \{0,1\}^U$, labelling memberships

$$y_u = d \quad \text{iff} \quad I_u \in \mathcal{C}_d.$$

Each y_u is assume to follow a multinomial distribution depending on parameter $\boldsymbol{\rho}$

$$\mathbb{P}\{y_u = d\} = \rho_d \quad \text{with} \quad \sum_{d\leq D}\rho_d = 1,$$

and in addition, the y_u are assumed to be independent, leading to

$$p(\mathbf{y}|\boldsymbol{\rho}, D) = \prod_{d\leq D}\rho_d^{|\mathcal{C}_d|}. \tag{10}$$

The random variable $Y^{I_u}_{ij}$ is now assumed to follow the conditional distribution

$$p\left(Y^{I_u}_{ij}|\mathbf{z},\mathbf{y}\right) = \frac{\left(\pi^{y_u}_{z_iz_j}\right)^{Y^{I_u}_{ij}}}{Y^{I_u}_{ij}!}\exp\left(-\pi^{y_u}_{z_iz_j}\right). \tag{11}$$

Notice that the new Poisson parameter $\pi^{y_u}_{z_iz_j}$ replaces $\pi^{I_u}_{z_iz_j}$ in the unconstrained version. The joint distribution of Y, given $\mathbf{z}$ and $\mathbf{y}$, can easily be obtained

$$p(Y|\mathbf{z},\mathbf{y},\pi) = \prod_{k,g}\prod_{d}\left(\frac{\left(\pi^{d}_{kg}\right)^{S_{kgd}}}{P_{kgd}}\exp\left(-|\mathcal{A}_k||\mathcal{A}_g||\mathcal{C}_d|\pi^{d}_{kg}\right)\right), \tag{12}$$

where

$$S_{kgd} = \sum_{z_i=k}\sum_{z_j=g}\sum_{y_u=d} Y^{I_u}_{ij}, \quad P_{kgd} = \prod_{z_i=k}\prod_{z_j=g}\prod_{y_u=d} Y^{I_u}_{ij}!.$$

Remark 1 The introduction of this hidden vector $\mathbf{y}$ is not the only way to impose regularity constraints to the integrated function $\Lambda_{kg}(t)$. For example, a segmentation constraint could be imposed by forcing each temporal cluster to contain only adjacent time intervals.

3.4.1 Summary

We have defined two generative models:

Model A the model has two meta-parameters, K the number of clusters and $\boldsymbol{\omega}$ the parameters of a multinomial distribution on $\{1, \ldots, K\}$. The hidden variable $\mathbf{z}$ is generated by the multivariate multinomial distribution of Eq. (2). Then, the model has a $K \times K \times U$ tensor of parameters π. Given $\mathbf{z}$ and π, the model generates a tensor of interaction counts Y using Eq. (9).

Model B is a constrained version of model **A**. In addition to the meta-parameters K and $\boldsymbol{\omega}$ of model **A**, it has two meta-parameters, D the number of clusters of time sub-intervals and $\boldsymbol{\rho}$ the parameters of a multinomial distribution on $\{1, \ldots, D\}$. The hidden variable $\mathbf{y}$ is generated by the multivariate multinomial distribution of Eq. (10). Model **B** has a $K \times K \times D$ tensor of parameters π. Given $\mathbf{z}, \mathbf{y}$ and π, the model generates a tensor of interaction counts Y using Eq. (12).

Unless specified otherwise "the model" is used for model **A**.

4 Estimation

4.1 Nonparametric estimation of integrated intensities

In this section, we assume that $\mathbf{z}$ is known. No hypothesis has been formulated about the shape of the functions $\{\Lambda_{kg}(t)\}_{\{k,g \le K, t \le T\}}$ and the increments of these functions over the partition introduced can be estimated by maximum likelihood (ML), thanks to Eq. (9)

$$\log \mathcal{L}(\pi | Y, \mathbf{z}) = \sum_{k,g} \sum_{u} \left[S_{kgu} \log\left(\pi_{kg}^{I_u}\right) - |\mathcal{A}_k||\mathcal{A}_g| \pi_{kg}^{I_u} + c \right],$$

where c denotes those terms not depending on π. It immediately follows

$$\hat{\pi}_{kg}^{I_u} = \frac{S_{kgu}}{|\mathcal{A}_k||\mathcal{A}_g|}, \qquad \forall (k,g), \tag{13}$$

where $\hat{\pi}_{kg}^{I_u}$ denotes the ML estimator of $\pi_{kg}^{I_u}$. In words, $\Lambda_{kg}(t_u) - \Lambda_{kg}(t_{u-1})$ can be estimated by ML as the total number of interactions on the sub-graph corresponding to the connections from cluster A_k to cluster A_g, over the time interval I_u, divided by the number of nodes on this sub-graph. Once the tensor π has been estimated, we have a pointwise, nonparametric estimator of $\Lambda_{kg}(t_u)$, for every $u \le U$, defined by

$$\hat{\Lambda}_{kg}(t_u) = \sum_{l=1}^{u} \hat{\pi}_{kg}^{I_l}, \quad \forall (k,g). \tag{14}$$

Thanks to the properties of the ML estimator, together with the linearity of (14), we know that $\hat{\Lambda}_{kg}(t_u)$ is an unbiased and convergent estimator of $\Lambda_{kg}(t_u)$.

Remark 2 Estimator (14) at times $\{t_u\}_{u \le U}$, can be viewed as an extension to random graphs and mixture models of the nonparametric estimator proposed in Leemis (1991). In that article, N-trajectories of independent NHPPs, sharing the same intensity function, are observed and the proposed estimator is basically obtained via method of moments.

In all the experiments, we consider the following step-wise linear estimator of $\Lambda_{kg}(t)$

$$\hat{\Lambda}_{kg}(t) = \sum_{u=1}^{U} \left[\hat{\Lambda}_{kg}(t_{u-1}) + \frac{\hat{\Lambda}_{kg}(t_u) - \hat{\Lambda}_{kg}(t_{u-1})}{t_u - t_{u-1}} (t - t_{u-1}) \right] \mathbf{1}_{[t_{u-1}, t_u[}(t), \tag{15}$$

which is a linear combination of estimators in Eq. (14) on the interval [0, T]. This is a consistent and unbiased estimator of $\Lambda_{kg}(t)$ at times $\{t_u\}_{u \le U}$ only.

When considering model **B**, Eqs. (13) and (14) are replaced by

$$\hat{\pi}_{kg}^{d} = \frac{S_{kgd}}{|\mathcal{A}_k||\mathcal{A}_g||\mathcal{C}_d|} \tag{16}$$

$$\hat{\Lambda}_{kg}(t_u) = \sum_{l=1}^{u} \hat{\pi}_{kg}^{y_l}. \tag{17}$$

Equation (15) remains unchanged, but an important difference between the constrained model and the unconstrained one should be understood: in the former, each interval I_u corresponds to a different slope for the function $\hat{\Lambda}_{kg}(t)$, whereas in the latter we only have D different slopes, one for each time cluster.

4.2 ICL

Since the vector $\mathbf{z}$, as well as the number of clusters K are unknown, estimator (13) cannot be used directly. Hence, we propose a two step procedure consisting in

1. providing estimates of $\mathbf{z}$ and K,
2. using these estimates to implement (13) and (14).

To accomplish the first task, the same approach followed in Côme and Latouche (2015) is adopted: we directly maximize the joint integrated log likelihood of complete data (ICL), relying on a greedy search over the labels and number of clusters. To perform such a maximization, we need the ICL to have an explicit form. This can be achieved by introducing conjugated prior distributions on the model parameters. The ICL can be written as

$$\mathcal{ICL}(\mathbf{z}, K) := \log(p(Y, \mathbf{z}|K)) = \log(p(Y|\mathbf{z}, K)) + \log(p(\mathbf{z}|K)). \tag{18}$$

This *exact* quantity is approximated by the well-known ICL *criterion* (Biernacki et al. 2000). This criterion, obtained through Laplace and Stirling approximations of the joint density on the left-hand side of Eq. (18), is used as a model selection tool, since it penalizes models with a high number of parameters. In the following, we refer to the joint log density in Eq. (18) as to the *exact ICL* to differentiate it from the ICL criterion.

We are now going to study in detail the two quantities on the r.h.s. of the above equation. The first probability density is obtained by integrating out the parameter π

$$p(Y|\mathbf{z},K) = \int p(Y,\pi|\mathbf{z},K)\mathrm{d}\pi.$$

In order to have an explicit formula for this term, we impose the following Gamma prior conjugated density over the tensor π:

$$p(\pi|a,b) = \prod_{k,g,u} \frac{b^a}{\Gamma(a)} \pi_{kgu}^{a-1} e^{-b\pi_{kgu}},$$

where the hyper-parameters of the Gamma prior distribution have been set constant to a and b for simplicity.[2] By using the Bayes rule

$$p(Y,\pi|\mathbf{z}) = p(Y|\pi,\mathbf{z})p(\pi|a,b),$$

we get:

$$p(Y,\pi|\mathbf{z}) = \prod_{k,g,u} \frac{b^a}{\Gamma(a)P_{kgu}} \pi_{kgu}^{S_{kgu}+a-1} \times \exp\left(-\pi_{kgu}\left[|\mathcal{A}_k||\mathcal{A}_g|+b\right]\right),$$

which can be integrated with respect to π to obtain

$$p(Y|\mathbf{z},K) = \prod_{k,g,u}\left[\frac{b^a}{\Gamma(a)P_{kgu}} \frac{\Gamma[S_{kgu}+a]}{\left[|\mathcal{A}_k||\mathcal{A}_g|+b\right]^{(S_{kgu}+a)}}\right]. \quad (19)$$

We now focus on the second density on the right-hand side

$$p(\mathbf{z}|K) = \int p(\mathbf{z},\boldsymbol{\omega}|K)\mathrm{d}\boldsymbol{\omega}.$$

A Dirichlet *prior* distribution can be attached to $\boldsymbol{w}$ in order to get an explicit formula, in a similar fashion of what we did with π:

$$\nu(\omega|K) = \mathrm{Dir}_K(\boldsymbol{\omega};\alpha,\ldots,\alpha).$$

The integrated density $p(\mathbf{z}|K)$ can be proven to reduce to

$$p(\mathbf{z}|K) = \frac{\Gamma(\alpha K)}{\Gamma(\alpha)^K} \frac{\prod_{k\leq K}\Gamma(|\mathcal{A}_k|+\alpha)}{\Gamma(N+\alpha K)} \quad (20)$$

4.3 Model B

When considering the constrained framework described at the end of the previous section, the ICL is defined

$$\mathcal{ICL}(\mathbf{z},\mathbf{y},K,D) := \log(p(Y,\mathbf{z},\mathbf{y}|K,D)) = \log(p(Y|\mathbf{z},\mathbf{y})) + \log(p(\mathbf{z}|K)) + \log(p(\mathbf{y}|D))$$

and it is maximized to provide estimates of $\mathbf{z}$, $\mathbf{y}$, K and D. The first density on the right-hand side is obtained by integrating out the hyper-parameter π. This integration can be done explicitly by attaching to π the following prior density function

$$\nu(\pi|a,b) = \prod_{k,g}\prod_{d} \frac{b^a}{\Gamma(a)} \pi_{kgd}^{a-1} e^{-b\pi_{kgd}}.$$

The second integrated density on the right-hand side can be read in (20) and the third is obtained by integrating out the parameter $\boldsymbol{\rho}$, whose prior density function is assumed to be

$$\nu(\boldsymbol{\rho}|D) = \mathrm{Dir}_D(\boldsymbol{\rho};\beta,\ldots,\beta).$$

The exact ICL is finally obtained by taking the logarithm of

$$p(Y,\mathbf{z},\mathbf{y}|K,D) = \prod_{k,g,d} \frac{b^a}{\Gamma(a)P_{kgd}} \frac{\Gamma[S_{kgd}+a]}{\left[|\mathcal{A}_k||\mathcal{A}_g||\mathcal{C}_d|+b\right]^{(S_{kgd}+a)}} \times \frac{\Gamma(\alpha K)}{\Gamma(\alpha)^K}\frac{\prod_{k\leq K}\Gamma(|\mathcal{A}_k|+\alpha)}{\Gamma(N+\alpha K)} \times \frac{\Gamma(\beta D)}{\Gamma(\beta)^D}\frac{\prod_{d\leq D}\Gamma(|\mathcal{C}_d|+\beta)}{\Gamma(U+\beta D)}. \quad (21)$$

4.4 Greedy search

By setting conjugated prior distributions over the model parameters, we obtained an ICL (Eq. (18)) in an explicit form. Nonetheless, explicit formulas to maximize it, with respect to $\mathbf{z}$ and K, do not exist. We then rely on a greedy search algorithm, which has been used to maximize the exact ICL, in the context of a standard SBM, by Côme and Latouche (2015). This algorithm basically works as follows:

1. An initial configuration for both $\mathbf{z}$ and K is set (standard clustering algorithms like *k-means* or hierarchical clustering can be used).
2. Labels switches leading to the highest increase in the exact ICL are repeatedly made. A label switch consists in a merge of two clusters or in a node switch from one cluster to another.

Remark 3 The greedy algorithm described in this section, makes the best choice *locally*. A convergence towards the global optimum in not guaranteed and often this optimum can only be approximated by a local optimum reached by the algorithm.

Remark 4 The *exact ICL* (as well as the *ICL criterion*) penalizes the number of parameters. Since the tensor π has dimension $K \times K \times U$, when U, which is fixed, is very

[2] The model can easily be extended to the more general framework: $p(\pi_{kgu}|a_{kgu},b_{kgu}) = \mathrm{Gamma}(\pi_{kgu}|a_{kgu},b_{kgu})$.

hight, the ICL will take its maximum for $K = 1$. In other words, the only way the ICL has to make the model more parsimonious is to reduce K up to one. By doing so, any community (or other) structure will not be detected. This over-fitting problem has nothing to see with the possible limitations of the greedy search algorithm and it can be solved by switching to model **B**.

Once $K_{\max}$ has been fixed, together with an initial value of $\mathbf{z}$, a shuffled sequence of all the nodes in the graph is created. Each node in the sequence is moved to the cluster, leading to the highest increase in the ICL, if any. This procedure is repeated until no further increase in the ICL is still possible. Henceforth, we refer to this step as to *Greedy Exchange* (**G E**). When maximizing the modularity score to detect communities, the **G E** usually is a final refinement step to be adopted after repeatedly merging clusters of nodes. In that context, moreover, the number of clusters is initialized to U and each node is alone in its own cluster. See, for example Noack and Rotta (2008). Here, we follow a different approach, proposed by Côme and Latouche (2015) and Blondel et al. (2008): after running the GE , we try to *merge* the remaining clusters of nodes in the attempt to increase the ICL. In this final step (henceforth **G M**), all the possible merges are tested and the best one is retained.

The ICL does not have to be computed before and after each swap/merge: possible increases can be assessed directly. When switching one node (say i) from cluster $\mathcal{A}_{k'}$ to $\mathcal{A}_l$, with $k' \neq l$, the change in the ICL is given by[3]

$$\Delta_{k'\to l} = \mathrm{ICL}(\mathbf{z}^*, K) - \mathrm{ICL}(\mathbf{z}, K).$$

The only statistics not simplifying are those involving k' and l; hence, the equation above can be read as follows

$$\begin{aligned}\Delta_{k'\to l} := \log&\left(\frac{\Gamma(|\mathcal{A}_{k'}| - 1 + \alpha)\Gamma(|\mathcal{A}_l| + 1 + \alpha)}{\Gamma(|\mathcal{A}_{k'}| + \alpha)\Gamma(|\mathcal{A}_l| + \alpha)}\right)\\
&+ \sum_{g\le K}\sum_{u\le U}\log\left(L^*_{k'gu}\right) + \sum_{g\le K}\sum_{u\le U}\log\left(L^*_{lgu}\right)\\
&+ \sum_{k\le K}\sum_{u\le U}\log\left(L^*_{kk'u}\right) + \sum_{k\le K}\sum_{u\le U}\log\left(L^*_{klu}\right)\\
&- \sum_{u}\left(\log\left(L^*_{k'k'u}\right) + \log\left(L^*_{k'lu}\right) + \log\left(L^*_{lk'u}\right) + \log\left(L^*_{llu}\right)\right)\\
&- \sum_{g\le K}\sum_{u\le U}\log\left(L_{k'gu}\right) - \sum_{g\le K}\sum_{u\le U}\log\left(L_{lgu}\right)\\
&- \sum_{k\le K}\sum_{u\le U}\log\left(L_{kk'u}\right) - \sum_{k\le K}\sum_{u\le U}\log\left(L_{klu}\right)\\
&+ \sum_{u}\left(\log(L_{k'k'u}) + \log(L_{k'lu}) + \log(L_{lk'u}) + \log(L_{llu})\right),\end{aligned} \tag{22}$$

where L_{kgu} is the term inside the product on the right-hand side of Eq. (19) and $\mathbf{z}^*$, and L^*_{kdu} refer to new configuration where the node i in in $\mathcal{A}_l$.

When merging clusters $\mathcal{A}_{k'}$ and $\mathcal{A}_l$ into the cluster $\mathcal{A}_l$, the change in the ICL can be expressed as follows:

$$\begin{aligned}\Delta_{k'\to l} &:= ICL(\mathbf{z}^*, K-1) - ICL(\mathbf{z}, K)\\
&= \log\left(\frac{p(\mathbf{z}^*|K-1)}{p(\mathbf{z}|K)}\right)\\
&\quad + \sum_{g\le K}\sum_{u\le U}\left(\log\left(L^*_{lgu}\right) + \log\left(L^*_{klu}\right)\right) - \sum_{u}\log\left(L^*_{llu}\right)\\
&\quad - \sum_{g\le K}\sum_{u\le U}\log\left(L_{k'gu}\right) - \sum_{g\le K}\sum_{u\le U}\log\left(L_{lgu}\right)\\
&\quad - \sum_{k\le K}\sum_{u\le U}\log\left(L_{kk'u}\right) - \sum_{k\le K}\sum_{u\le U}\log\left(L_{klu}\right)\\
&\quad + \sum_{u}\left(\log(L_{k'k'u}) + \log(L_{k'lu}) + \log(L_{lk'u}) + \log(L_{llu})\right).\end{aligned} \tag{23}$$

When working with model **B**, we need to initialize $D_{\max}$ and $\mathbf{y}$. Then, a shuffled sequence of time intervals $I_1, \ldots, I_U$ is considered and each interval is swapped to the time cluster, leading to the highest increase in the ICL (**G E** for time intervals). When no further increase in the ICL is possible, we look for possible merges between time clusters in the attempt to increase the ICL (**G M** for time intervals). Formulas to directly assess the increase in the ICL can be obtained, similar to those for nodes swaps and merges. In case of model **B**, different strategies are possible to optimize the ICL:

1. **G E** + **G M** for nodes at first and then for times (we will call this strategy **T N**, henceforth).
2. **G E** + **G M** for time intervals at first and then for nodes (**N T** strategy).
3. An hybrid strategy, involving alternate switching of nodes and time intervals (**M** strategy).

We will provide details about the chosen strategy case by case in the following.

5 Experiments

In this section, experiments on both synthetic and real data are provided. All running times are measured on a twelve cores Intel Xeon server with 92 GB of main memory running a GNU Linux operating system, the greedy algorithm described in Sect. 4.4 being implemented in C++. A Euclidean hierarchical clustering algorithm was used to initialize the labels, and $K_{\max}$ was set to $N/2$.

[3] Hereafter, the "*" notation refers to the statistics *after* switching/merging.

In the following, we call TSBM the temporal SBM we propose and we refer to the optimization algorithm described in the previous section as greedy ICL.

5.1 Simulated data

5.1.1 *First scenario*

We start by investigating how the proposed approach can be used to efficiently estimate the vector $\mathbf{z}$ of labels in situations where the standard SBM fails. Thus, we simulate interactions between 50 (N) nodes, grouped in two hidden clusters $\mathcal{A}_1$ and $\mathcal{A}_2$, over 100 (U) time intervals of unitary length. The generative model considered for the simulations depends on two time clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ containing a certain number of time intervals $I_1, \ldots I_U$. If I_u is in $\mathcal{C}_1$, then $Y_{ij}^{I_u}$ is drawn from a Poisson distribution $\mathcal{P}(P_{z_i z_j})$. Otherwise, $Y_{ij}^{I_u}$ is drawn from a Poisson distribution $\mathcal{P}(Q_{z_i z_j})$. The matrices P and Q are given by

$$P = \begin{pmatrix} \psi & 1 \\ 1 & \psi \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} 1 & \psi \\ \psi & 1 \end{pmatrix},$$

where ψ is a free parameter in $[1, \infty)$. When this parameter is equal to 1, we are in a degenerate case and there is not any structure to detect: all the nodes are placed in the same, unique cluster. The higher ψ, the stronger the *contrast* between the interactions pattern inside and outside the cluster. In this paragraph, ψ is set equal to 2 and the proportions of the clusters are set equal ($\boldsymbol{\omega} = (1/2, 1/2)$). The number of time intervals assigned to each time cluster is assumed to be equal to $U/2$. In the following, we consider

$$\mathcal{C}_1 := \{I_1, \ldots, I_{25}\} \cup \{I_{51}, \ldots, I_{75}\},$$
$$\mathcal{C}_2 := \{I_{26}, \ldots, I_{50}\} \cup \{I_{76}, \ldots, I_{100}\}.$$

This generative model defines two integrated intensity functions (IIFs), say $\Lambda_1(t)$ and $\Lambda_2(t)$. The former is the IIF of the Poisson processes counting interactions between nodes sharing the same cluster, the latter is the IIF of the Poisson processes counting interactions between vertices in different clusters. These IIFs are shown in Fig. 1a.

A tensor Y, with dimensions $N \times N \times U$, is drawn. Its (i, j, u) component is the sampled number of interactions from node i to node j over the time interval I_u. Moreover, sampled interactions are aggregated over the whole time horizon to obtain an adjacency matrix. In other words, each tensor is integrated over its third dimension. We compared the greedy ICL algorithm with the Gibbs sampling approach introduced by Nouedoui and Latouche (2013). The former was run on the tensor Y (providing estimates in 11.86 s on average) the latter on the corresponding adjacency matrix. This experiment was repeated 50 times, and estimates of random vector $\mathbf{z}$ were provided at each iteration. Each estimate $\hat{\mathbf{z}}$ is compared with the true $\mathbf{z}$ and an adjusted rand index (ARI Rand 1971) is computed. This index takes values between zero and one, where one corresponds to the perfect clustering (up to label switching).

Remark 5 The true structure is always recovered by the TSBM: 50 unitary values of the ARI are obtained. Conversely, the standard SBM never succeeds in recovering any hidden structures present in the data (50 null ARIs are obtained). This can easily be explained since the time clusters have opposite interaction patterns, making them hard to uncover when aggregating over time.

Relying on an efficient estimate of $\mathbf{z}$, the two integrated intensity functions can be estimated through the estimator in Eq. (15). Results are shown in Fig. 1b, where the estimated functions (coloured dots) overlap the real functions 1a.

Over-fitting We now illustrate how the model discussed so far fails in recovering the true vector $\mathbf{z}$ when the number of time intervals (and hence of free parameters) grows. We consider the same generative model of the previous paragraph, with a lower ψ:

$$P = \begin{pmatrix} 1.4 & 1 \\ 1 & 1.4 \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} 1 & 1.4 \\ 1.4 & 1 \end{pmatrix}.$$

Despite the lower contrast (from 2 to 1.4 in P and Q), with $U = 100$ and time sub-intervals of unitary length, the TSBM model still always recovers the true vector $\mathbf{z}$. Now, we consider a finer partition of [0, 100] by setting $U = 1000$ and $\Delta_u = 0.1$ as well as scaling the intensity matrices as follows

$$\tilde{P} := \begin{pmatrix} 0.14 & 0.1 \\ 0.1 & 0.14 \end{pmatrix} \quad \text{and} \quad \tilde{Q} = \begin{pmatrix} 0.1 & 0.14 \\ 0.14 & 0.1 \end{pmatrix}.$$

Moreover, we set

$$\mathcal{C}_1 := \{I_1, \ldots, I_{250}\} \cup \{I_{501}, \ldots, I_{750}\}$$

and $\mathcal{C}_2$ is the complement of $\mathcal{C}_1$, as previously. Finally, we sampled 50 dynamic graphs over the interval [0, 100] from the corresponding generative model. Thus, each graph is characterized by a sampled tensor Y.

Unfortunately, the model is not robust to such changes. Indeed, when running the greedy ICL algorithm on each sampled tensor Y, the algorithm does not see any community structure and all nodes are placed in the same cluster. This leads to a null ARI, for each estimation. As mentioned in Sect. 4.4, the ICL penalizes the number of parameters and since the tensor π has dimension $K \times K \times U$, for a fixed K, when moving from the larger decomposition ($U = 100$) to the finer one ($U = 1000$), the number of free parameters in the model is

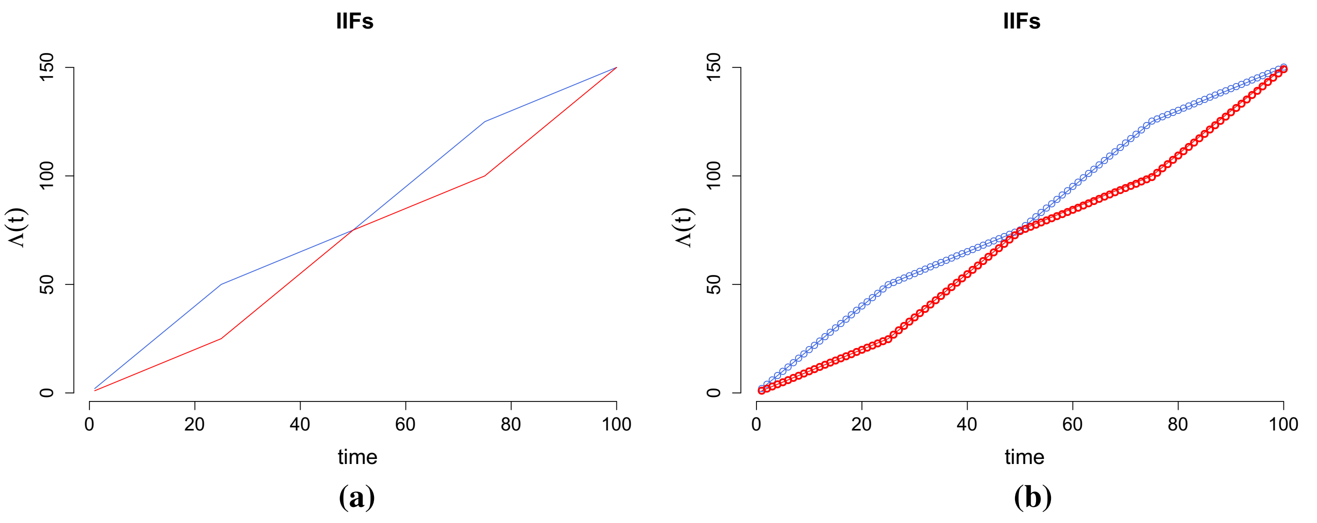


Fig. 1 Real (**a**) and estimated (**b**) integrated intensity functions (IIFs) according to the considered generative model ($\psi = 2$). In *blue* we have $\Lambda_1(t)$, for $\psi = 4$, in *red* $\Lambda_2(t)$ (colour figure online)

approximatively[4] multiplied by 10. The increase we observe in the likelihood, when increasing the number of clusters of nodes from $K = 1$ to $K = 2$, is not sufficient to compensate the penalty due to the high number of parameters, and hence, the ICL decreases. Therefore, the maximum is taken for $K = 1$ and a single cluster is detected.

Model **B** allows to tackle this issue. When allowing the integrated intensity functions $\Lambda_1(t)$ and $\Lambda_2(t)$ to grow at the same rate on each interval I_u belonging to the same time cluster $\mathcal{C}_d$, we basically reduce the third dimension of the tensor π from U to D.

The greedy ICL algorithm for Model **B** was run on each sampled tensor Y, providing estimates of **z** and **y** in 2.38 min, on average. A hierarchical clustering algorithm was used to initialize the time labels **y**, and the initial number of time clusters was set to $D_{\max} = \sqrt{U}$. In an attempt to avoid convergence to local maxima, ten estimates are built for each tensor, and the estimate leading to the best ICL is finally retained. The adjusted rand index is used to evaluate the clustering, as previously, and the results are presented as box plots in Fig. 2. Note that the results were obtained through the optimization strategy **T N**. The other two strategies described in Sect. 4.4, namely the **N T** strategy and the **M** strategy, led to similar results in terms of final ICL and ARIs.

5.1.2 Second scenario

Since the node clusters are fixed over time, the TSBM model can be seen as an alternative to a standard SBM to estimate the label vector **z**. The previous scenario shows that the TSBM can recover the true vector **z** in situations where the SBM fails. In this paragraph, we show how the TSBM and the SBM can sometimes have similar performances.

We considered dynamic graphs with 50 (N) nodes and 50 (U) time intervals

$$I_1, \ldots, I_{50}.$$

These time intervals are grouped in two time clusters $\mathcal{C}_1$ and $\mathcal{C}_2$, the former containing the first 25 time intervals, the latter the last 25 time intervals. If I_u is in $\mathcal{C}_1$, then $Y_{ij}^{I_u}$ is drawn from a Poisson distribution $\mathcal{P}(P_{z_i z_j})$. Otherwise, $Y_{ij}^{I_u}$ is drawn from a Poisson distribution $\mathcal{P}(2P_{z_i z_j})$. The P matrix is given by

$$P = \begin{pmatrix} \psi & 2 \\ 2 & \psi \end{pmatrix}$$

and ψ is a free parameter in $[2, +\infty)$. Hence, we have two different integrated intensity functions, say $\Lambda_1(t)$ and $\Lambda_2(t)$ with the same roles as in the previous section. These two functions are plotted in Fig. 3a, for a value of $\psi = 4$.

We investigated six values for the parameter ψ

$$\{2.1, 2.2, 2.3, 2.4, 2.5, 2.6\}.$$

For each value of ψ, we sampled 50 tensors Y, of dimension $(50 \times 50 \times 50)$, according to the generative model considered. Interactions are aggregated over the time interval [0, 50] to obtain adjacency matrices. We ran the greedy ICL algorithm on each tensor and the Gibbs sampling (SBM) algorithm on each adjacency matrix. For the greedy ICL algorithm, estimates of vector **z** were obtained in a mean running time of 5.52 s. As previously, to avoid

[4] The dimension of the vector $\boldsymbol{\omega}$ does not change.

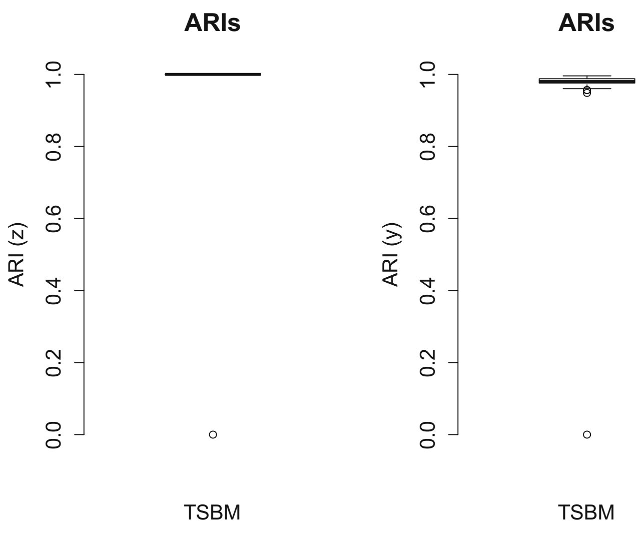


Fig. 2 *Box plots* for both clusterings of nodes and time intervals: 50 dynamic graphs are sampled according to the considered generative model, estimates of **z** and **y** are provided by the greedy ICL (model B)

convergence to local maxima, ten different estimates are built for each tensor, the one leading to the highest ICL being retained. The results are presented as box plots in Fig. 4. Although the SBM leads to slightly better clustering results for small values of ψ (2.2, 2.3) and the TSBM for higher values of ψ (2.5, 2.6), we observe that the two models have quite similar performances (in terms of accuracy) in this scenario.

To provide some intuitions about the scalability (see next paragraph) of the proposed approach we repeated the previous experiment by setting $K = 3$ clusters, corresponding to the following connectivity matrix:

$$P = \begin{pmatrix} \psi & 2 & 2 \\ 2 & \psi & 2 \\ 2 & 2 & \psi \end{pmatrix}.$$

The assignment of the time intervals to the time clusters is unchanged as well as the connectivity pattern on each time cluster are unchanged. The contrast parameter ψ takes values in the set $\{2, 2.5, 2.10, \ldots, 2.8\}$ and 50 dynamic graphs were sampled, according to the described settings, for each value of ψ. We ran the TSBM on each dynamic graph obtaining 50 estimates of the labels vector **z** (one for each ψ) and box and whiskers plots for each group of ARIs are shown in Fig. 5. By comparing this figure with Fig. 4a, we can see that the model needs a slight higher contrast to fully recover the true structure. Actually, when increasing the number of clusters without increasing the number of nodes, the size of each cluster decreases (on average) and since the estimator of **z** we are using is related to the ML estimator, we can imagine a slower convergence to the true value of **z**.

5.1.3 *Scalability*

A full scalability analysis of the proposed algorithm as well as the convergence properties of the proposed estimators are outside the scope of this paper. Nonetheless, in "Appendix" we provide details about the computational complexity of the greedy ICL algorithm. Future works could certainly be devoted to improve both the algorithm efficiency and scalability through the use of more sophisticated data structures.

5.2 Real data

The data set used in this section was collected during the ACM Hypertext conference held in Turin, from 29 June 2009 to 1 July 2009. We focus on the first conference day (24 h) and consider a dynamic network with 113 (N) nodes (conference attendees) and 96 (U) time intervals (the consecutive quarter-hours in the period: 8 a.m. of June 29th–7.59 a.m. of June 30th). The network edges are the proximity face to face interactions between the conference attendees. An interaction is monitored when two attendees are face to face, nearer than 1.5 m for a time period of at least 20 s.[5] The data set we considered consists of several lines similar to the following one

ID1	ID2	Time interval (15 min)	Number of interactions
52	26	5	16

It means that conference attendees 52 and 26, between 9 a.m. and 9.15 a.m., have spoken for $16 \times 20\,\mathrm{s} \approx 5\,\mathrm{min}\,30\,\mathrm{s}$.

We set $K_{\max} = 20$ and the vector **z** was initialized randomly: each node was assigned to a cluster following a multinomial distribution. The greedy algorithm was run ten times on the considered dataset, each time with a different initialization and estimates of **z** and K were provided in 13.81 s, on average. The final values of the ICL can be observed as box plots in Fig. 6.

The estimates associated with the highest ICL correspond to 5 node clusters. In Fig. 7, we focus on the cluster $\mathcal{A}_4$, containing 48 nodes. In Fig. 7a, we plotted the time cumulated interactions inside the cluster. As it can be seen that the connectivity pattern for this cluster is very representative of the entire graph: between 13 p.m. and 14 p.m. and 18 p.m. and 19.30 p.m. there are significant increases

[5] More informations about the way the data were collected can be found in Isella et al. (2011) or visiting the website http://www.sociopatterns.org/datasets/hypertext-2009-dynamic-contact-network/.

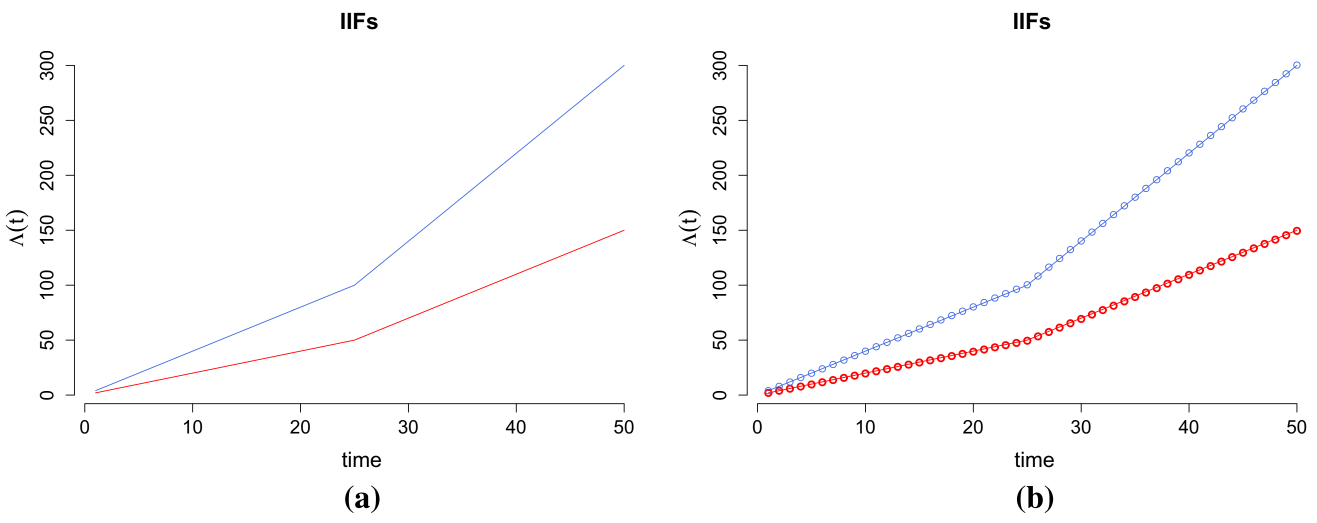


Fig. 3 Real (**a**) and estimated (**b**) integrated intensity functions (IIFs) according to the considered generative model. In *blue* we have $\Lambda_1(t)$, for $\psi = 4$, in *red* $\Lambda_2(t)$ (colour figure online)

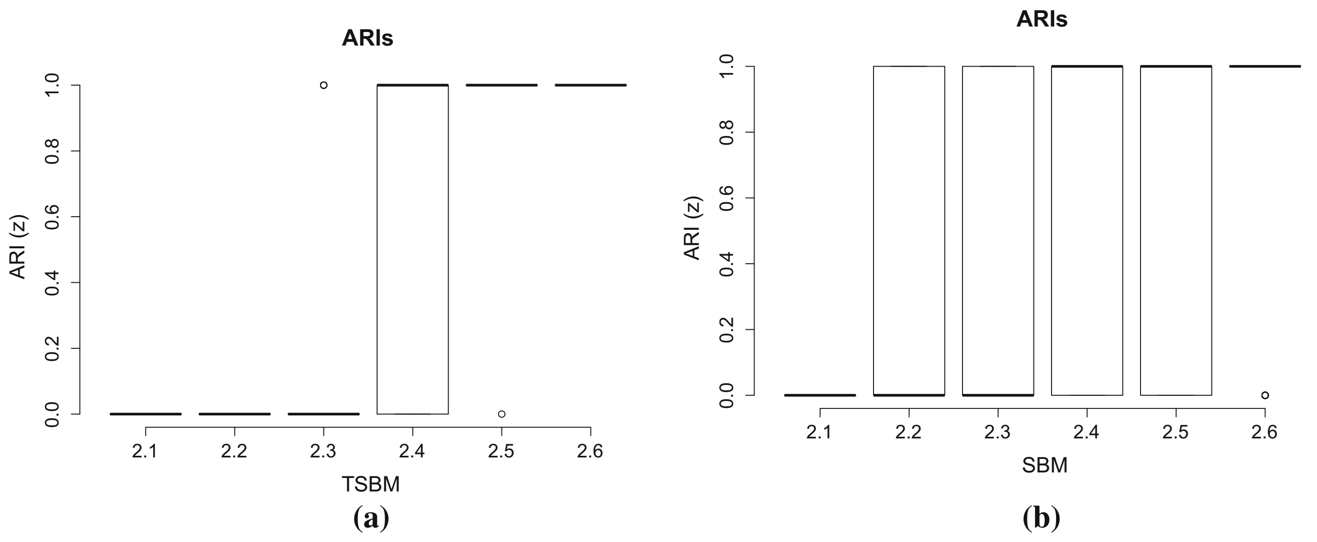


Fig. 4 *Box plots* of ARIs for different levels of contrast (ψ). We compare the proposed model with a standard SBM. **a** ARIs obtained by greedy ICL and **b** ARIs obtained with the Gibbs sampling procedure for SBM

in the interactions intensity. The estimated integrated intensity function (IIF) for interactions inside this cluster is shown in Fig. 7b. The function has a higher slope on those time intervals where attendees in the cluster are more likely to have interactions. The vertical red lines delimit two important times of social gathering:[6]

- 13.00–15.00—lunch break.
- 18.00–19.00—wine and cheese reception.

We conclude this section by illustrating how Model **B** can be used to assign time intervals on which interactions have similar intensity to the same time cluster. We run the greedy ICL algorithm for Model **B** on the data set by using the optimization strategy **M** described at the end of Sect. 4.4 (other strategies lead in this case to similar results) and $D_{\max}$ was set equal to 20. The time clustering provided by the greedy ICL algorithm is shown in Fig. 8. On the left-hand side, the aggregated interactions for each quarter-hour during the first day are reported. On the right-hand side, interactions taking place into those time intervals assigned to the same time cluster have the same form/colour. Two important things should be noticed:

1. The obtained clustering seems meaningful: the three time intervals with the highest interactions level are placed in the same cluster (blue), apart from all the others. More in general, each cluster is associated with a certain intensity level, so time intervals in the same

[6] More informations at http://www.ht2009.org/program.php.

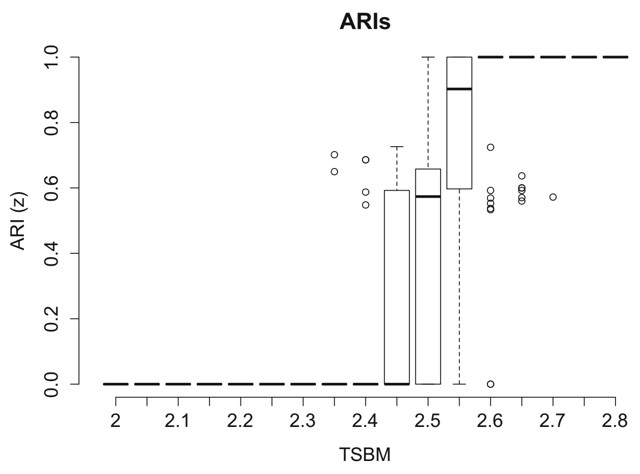


Fig. 5 *Box plots* of ARIs for different levels of contrast (ψ). Data have been sampled by non-homogeneous Poisson processes counting interactions in a dynamic graph whose nodes are grouped in three clusters and interactivity patterns vary across two time clusters

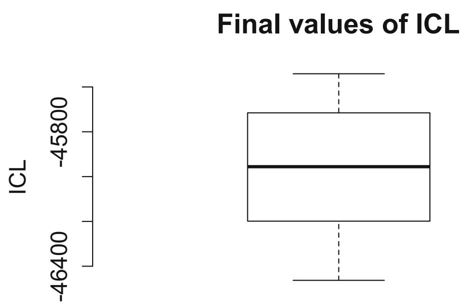


Fig. 6 *Box plot* of the ten final values of the ICL produced by the greedy ICL algorithm for different initializations

cluster, not necessarily adjacent, share the same global interactivity pattern.

2. There are not constraints on the number of abruptly changes connected with these five time clusters. In other words, time clusters do not need to be adjacent and this is the real difference between the approach considered in this paper (time clustering) and a pure segmentation one.

6 Conclusion

We proposed a non-stationary extension of the stochastic block model (SBM) allowing us to cluster nodes of a network is situations where the classical SBM fails. The approach we chose consists in partitioning the time interval over which interactions are studied into sub-intervals of fixed length. Those intervals provide aggregated interaction counts that are increments of non-homogeneous Poisson processes (NHPPs). In a SBM inspired perspective, nodes are clustered in such a way that aggregated interaction counts are homogeneous over clusters. We derived an exact integrated classification likelihood (ICL) for such a model and proposed to maximize it through a greedy search strategy. Finally, a nonparametric maximum likelihood estimator was developed to estimate the integrated intensity functions of the NHPPs counting interactions between nodes. The experiments we carried out on artificial and real-world networks highlight the capacity of the model to capture non-stationary structures in dynamic graphs.

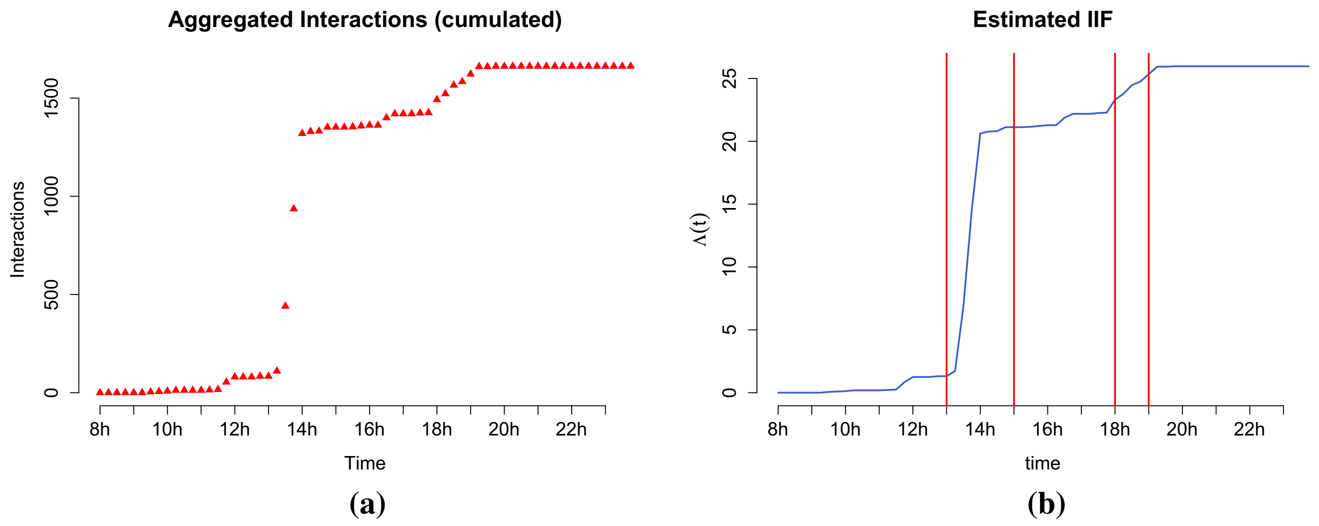
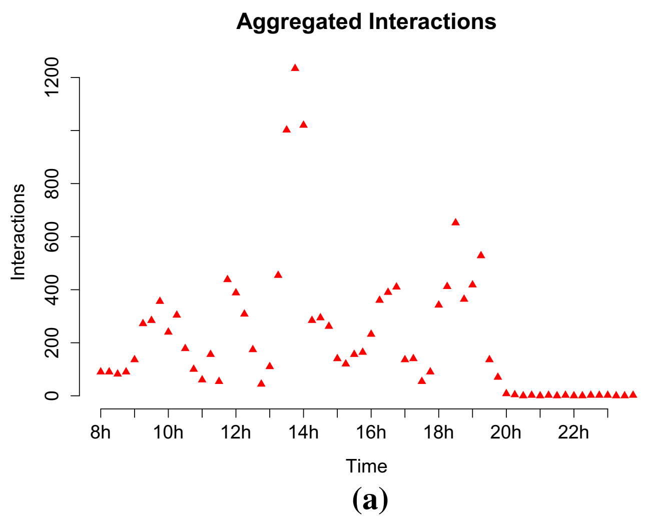


Fig. 7 **a** Cumulated aggregated connections for each time interval for cluster $\mathcal{A}_4$. **b** The estimated IIF for interactions inside cluster $\mathcal{A}_4$. *Vertical red lines* delimit the lunch break and the wine and cheese reception (colour figure online)



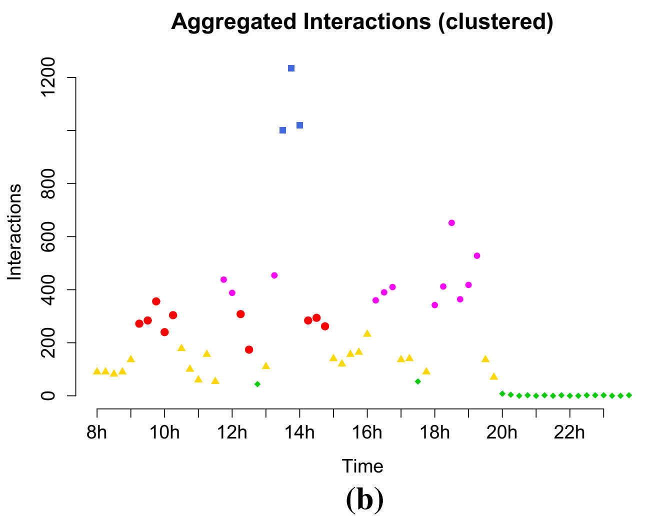


Fig. 8 **a** Aggregated connections for each time interval for the whole network.**b** Interactions of the same form/*colour* take place on time intervals assigned to the same cluster (model **B**) (colour figure online)

Appendix: Computational complexity

In this section, we provide details about the computational complexity of the main model presented in this paper, namely the model **A**. Assuming that the gamma function can be computed in constant time (see Press et al. 2007), we focus on the three statistics appearing in Eq. (9), namely

1. $S_{kgu} := \sum_{z_i=k} \sum_{z_j=g} Y_{ij}^{I_u}$,
2. $P_{kgu} := \prod_{z_i=k} \prod_{z_j=g} Y_{ij}^{I_u}!$,
3. $R_{kg} := |\mathcal{A}_k||\mathcal{A}_g|$.

The whole computation task consists in evaluating the increase in ICL induced by nodes exchanges and merges. Those computations involves the three quantities listed above. The tensor $\{S_{kgu}\}_{k,g \le K, u \le U}$ is stored in a three-dimensional array, never resized, occupying a $O(K_{\max}^2 U)$ memory space. Hence, at any time during the algorithm its elements can be accessed and modified in constant time. The tensor $\{P_{kgu}\}_{k,g \le K, u \le U}$ is handled similarly and clusters sizes (we recall that $|\mathcal{A}_k|$ corresponds to the size of cluster $\mathcal{A}_k$) are also stored in arrays. In order to evaluate the ICL changes, induced by an operation, we need to maintain aggregated interaction counts for each node: for a node i we have, e.g.

$$S_{igu} := \sum_{z_j=g} Y_{ij}^{I_u},$$

the number of interactions from node i to cluster $\mathcal{A}_g$ inside the time interval I_u. Similarly

$$S'_{igu} := \sum_{z_j=g} Y_{ji}^{I_u}$$

denotes the number of interactions from cluster $\mathcal{A}_g$ to node i inside the time interval I_u. Other related quantities are considered. These structures occupy a memory space of $O(N^2 U)$.

Exchanges

In order to evaluate the ICL increase induced by the switch of a node (say i) from cluster $\mathcal{A}_{k'}$ to cluster $\mathcal{A}_l$, we perform the following operations:

- $S_{k'gu}$ (respectively, $S_{gk'u}$) is reduced by S_{igu} (S'_{igu}) and S_{lgu} (S_{glu}) is increased by the same amount;
- $P_{k'gu}$ (respectively, $P_{g'ku}$) is reduced by P_{igu} (P'_{igu}) and P_{lgu} (P_{glu}) is increased by the same amount;
- $\mathcal{A}_{k'}$ ($\mathcal{A}_l$) is reduced (increased) by one.

Although these operations are in constant time, they are involved in a sum with (KU) elements (this can be seen in Eq. (22)), so that the total cost of the test is $O(KU)$. Since node i can be switched to $K-1$ remaining clusters and the graph has N nodes, the cost of a *full* exchange routine is $O(NK^2U)$.

Remark 6 When a node is actually switched from its cluster to another one, all data structures are updated, but the update cost is dominated by the cost of the testing phase described above.

Notice that we have evaluated the total cost of one full exchange routine, i.e. in the case where all nodes are considered once. Reductions in the number of clusters (very likely to be induced by exchanges in case $K_{\max}$ is high) are not taken into account.

Merges

The entire merge routine, consisting in a test phase and an actual merge, has a computational cost that is dominated by the cost of exchanges. Consider a cluster $\mathcal{A}_{k'}$. We first look for the cluster (say $\mathcal{A}_l$), leading to the best merge (highest increase in the ICL) with $\mathcal{A}_{k'}$. This operation has a cost of $O(K^2U)$: for each $\mathcal{A}_l$ the evaluation of the increase in ICL has a cost of $O(KU)$ (see Eq. (23)) and l can take $K-1$ possible values. Since we look for the best merge for all $k' \in \{1,\ldots,K\}$, the computational cost for a merge of two nodes clusters is $O(K^3U)$, where we recall that $D \leq N$.

Total cost

The worst case complexity for one iteration of the algorithm, with each node considered once, is $O(NK^2U)$. However, it is difficult to evaluate the actual complexity of the whole algorithm for two reasons. Firstly, we have no way to estimate the number of exchanges needed in the exchange phase. Secondly, nodes exchanges are very likely to reduce the number of clusters, especially at the beginning of the algorithm, when $K_{\max}$ is relatively high. Thus, the individual cost of an exchange reduces very quickly, leading to a vast overestimation of its cost using the proposed bounds. A detailed evaluation of the behaviour of the proposed algorithm, although outside the scope of the this paper, would be necessary to assess its use on large data sets.

References

Biernacki C, Celeux G, Govaert G (2000) Assessing a mixture model for clustering with the integrated completed likelihood. IEEE Trans Pattern Anal Mach Intell 22(7):719–725

Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech: Theory Experi 2008(10):10008. http://stacks.iop.org/1742-5468/2008/i=10/a=P10008

Côme E, Latouche P (2015) Model selection and clustering in stochastic block models based on the exact integrated complete data likelihood. Stat Model 15(6):564–589

Corneli M, Latouche P, Rossi F (2015) Modelling time evolving interactions in networks through a non stationary extension of stochastic block models. In: Pei J, Silvestri F, Tang J (eds) International conference on advances in social networks analysis and mining ASONAM 2015. IEEE/ACM, ACM, Paris, France, pp 1590–1591. https://hal.archives-ouvertes.fr/hal-01263540

Dubois C, Butts C, Smyth P (2013) Stochastic blockmodelling of relational event dynamics. In: International conference on artificial intelligence and statistics. Volume 31 of the Journal of Machine Learning Research Proceedings, pp 238–246

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174

Goldenberg A, Zheng X, Fienberg SE, Airoldi EM (2009) A survey of statistical network models. Mach Learn 2(2):129–133

Guigourès R, Boullé M, Rossi F (2015) Discovering patterns in time-varying graphs: a triclustering approach. Adv Data Anal Classif. doi:10.1007/s11634-015-0218-6

Guigourès R, Boullé M, Rossi F (2012) A triclustering approach for time evolving graphs. In: Co-clustering and applications, IEEE 12th international conference on data mining workshops (ICDMW 2012). Brussels, Belgium, pp 115–122

Holland P, Laskey K, Leinhardt S (1983) Stochastic blockmodels: first steps. Soc Netw 5:109–137

Isella L, Stehl J, Barrat A, Cattuto C, Pinton J, Van den Broeck W (2011) What's in a crowd? Analysis of face-to-face behavioral networks. J Theor Biol 271(1):166–180

Leemis LM (1991) Nonparametric estimation of the cumulative intensity function for a nonhomogeneous poisson process. Manag Sci 37(7):886–900. http://www.jstor.org/stable/2632541

Lorrain F, White H (1971) Structural equivalence of individuals in social networks. J Math Sociol 1:49–80

Matias C, Rebafka T, Villers F (2015) Estimation and clustering in a semiparametric Poisson process stochastic block model for longitudinal networks, HAL (preprint)

Noack A, Rotta R (2008) Multi-level algorithms for modularity clustering. CoRR arXiv:0812.4073

Nouedoui L, Latouche P (2013) Bayesian non parametric inference of discrete valued networks. In: 21th European symposium on artificial neural networks, computational intelligence and machine learning (ESANN 2013). Bruges, Belgium, pp 291–296

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007) Numerical recipes 3rd edition: the art of scientific computing, 3rd edn. Cambridge University Press, Cambridge

Rand WM (1971) Objective criteria for the evaluation of clustering methods. J Am Stat Assoc 66(336):846–850

Schaeffer SE (2007) Graph clustering. Comput Sci Rev 1(1):27–64

Wang Y, Wong G (1987) Stochastic blockmodels for directed graphs. J Am Stat Assoc 82:8–19

Wasserman S, Faust K (1994) Social network analysis: methods and applications, vol 506. Cambridge University Press, Cambridge

White HC, Boorman S, Breiger R (1976) Social structure from multiple networks: I. Blockmodels of roles and positions. Am J Sociol 81(4):730–780

Wyse J, Friel N, Latouche P (2014) Inferring structure in bipartite networks using the latent block model and exact icl. arXiv preprint arXiv:1404.2911

Xing EP, Fu W, Song L (2010) A state-space mixed membership blockmodel for dynamic network tomography. Ann Appl Stat 4(2):535–566

Xu KS, Hero III AO (2013) Dynamic stochastic blockmodels: statistical models for time-evolving networks. In: Proceedings of the 6th International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction, pp 201–210

Yang T, Chi Y, Zhu S, Gong Y, Jin R (2011) Detecting communities and their evolutions in dynamic social networks—a Bayesian approach. Mach Learn 82(2):157–189