myjournal manuscript No.
(will be inserted by the editor)

Dynamic stochastic block models

Parameter estimation and detection of changes in community structure

Matthew Ludkin · Idris Eckley · Peter Neal

Received: date / Accepted: date

Abstract The stochastic block model (SBM) is widely used for modelling network data by assigning individuals (nodes) to communities (blocks) with the probability of an edge existing between individuals depending upon community membership. In this paper we introduce an autoregressive extension of the SBM, based on continuous-time Markovian edge dynamics. The model is appropriate for networks evolving over time and allows for edges to turn on and off. Moreover, we allow for the movement of individuals between communities. An effective reversible jump Markov chain Monte Carlo algorithm is introduced for sampling jointly from the posterior distribution of the community parameters and the number and location of changes in community membership. The algorithm is successfully applied to a network of mice.

Keywords Stochastic Block Model · autoregressive Dynamic Network · Reversible jump MCMC · Continuous Time Network

We gratefully acknowledge the support of the EPSRC funded EP/H023151/1 STOR-i Centre for Doctoral Training

M. Ludkin
E-mail: m.ludkin1@lancaster.ac.uk
STOR-i Centre for Doctoral Training, Department of Mathematics and Statistics, Lancaster University, Lancaster, LA1 4YF, UK
I. Eckley
STOR-i Centre for Doctoral Training, Department of Mathematics and Statistics, Lancaster University, Lancaster, LA1 4YF, UK
P. Neal
Department of Mathematics and Statistics, Lancaster University, Lancaster, LA1 4YF, UK

1 Introduction

Network models play a key role in capturing and understanding population dynamics in a range of scenarios. Networks often show some form of structure rather than simple random interactions and this has led to a plethora of network models to capture such dynamics. Structures studied in the literature include: Barabsi-Albert model (Albert and Barabási 2002) (a scale-free model generated by preferential attachment), Watts-Strogatz model (Watts and Strogatz 1998) (small-world model), exponential random graph model (ERGM)(Frank and Strauss 1986) (specified frequencies of subgraphs) and the stochastic block model (SBM) (Frank and Harary 1982) (community model). This body of research covers a broad range of subject areas including the social sciences, statistics, physics and computational biology.

In this paper we consider the statistical detection of changes in the community structure of a dynamic network. The challenge of detecting changes in data sequences is well-known, receiving considerable attention in the statistics literature in recent years. Much of this effort has been focused on changepoint detection within univariate data sequences, for example, see Davis et al (2006); Fearnhead and Liu (2007); Picard et al (2007); Killick et al (2012); Fryzlewicz (2014); Haynes et al (2017). More recently, the literature has turned to focus on the detection of changes in more complex settings including multivariate time series (e.g. Matteson and James (2014); Xie and Siegmund (2013), spatial-temporal (Altieri et al 2015) and related challenges with network data (e.g. Fu et al (2009); Yang et al (2011); Xu and Hero (2014); Matias and Miele (2016)).

Within a network context, changing behaviour can arise in many different scenarios. This article focuses on movement of individuals from one community to an-

other with the interactions between individuals depending upon their community. Animals changing their mating partners is a prime example of such behaviour. Detecting changes in community structure in animal herds could help indicate the source of disease outbreaks and help with decisions such as targeted vaccination programs. In Section 6, we study the changes in community structure in a network of mice first presented in Lopes et al (2016b).

The different network models described above typically capture different network features. For example, the Watts-Strogatz model can create clusters whilst keeping a small distance between any two chosen nodes. This model has no simple parametric form, hence non-parametric methods are used to assess model fit (Kolaczyk 2009). The ERGM can create clusters of nodes with specified sub-graph properties but is known to suffer from identifiability problems (Chatterjee and Diaconis 2013) since two different parameterisations can lead to the same model. Given that our primary interest is in community dynamics, we focus on a dynamic, autoregressive extension of the SBM, introduced by Holland et al (1983). The general form of the SBM model is given by Snijders and Nowicki (1997) who discuss maximum likelihood estimation and an Expectation Maximisation algorithm for inferring the parameters for the SBM. The SBM aims to partition the set of nodes in a network in such a way that the proportion of edges between nodes in the same block is different to the proportion of edges between nodes in different blocks.

In this paper, the autoregressive stochastic block model (ARSBM) is introduced. This model is inspired by populations where the network of contacts (edges) between individuals evolve over time and depend upon the community (block) to which individuals belong; see, for example, the mice network data, Section 6 and Lopes et al (2016b). The edges are binary states 1/0 which alternate between being on (1) and off (0), spending time in a given state before transiting to the other state. The observed data consist of snapshots of the network over time with snapshots close together in time typically being more similar to those further apart. The correlation in the presence/absence of edges is a key feature of the data we want to explore and capture in our modelling. In addition, we seek to infer other important characteristics of the population such as the amount of movement of individuals between communities (blocks) and the interactions both within and between blocks.

There have been a number of extensions of the SBM to include temporal dynamics. Various authors have considered a continuous-time model based on an SBM where the edge processes are non-homogeneous Poisson point processes (DuBois et al 2013; Guigourès et al 2015; Corneli et al 2016; Xin et al 2017; Matias et al 2017). This is appropriate for event data such as sending emails or SMS. However, for edge processes which have a duration, such as phone calls and the status of friendships in a social network, a model which accounts for the time for which an edge lasts is required. Another direction which has attracted attention is discrete time dynamic extensions of the SBM (Fu et al 2009; Yang et al 2011; Xu and Hero 2014; Matias and Miele 2016). These papers have focused on discrete-time dynamics for both community membership and the network evolution over time. A key assumption of these works is that, conditional upon the community structure, the networks at each time point are independent SBMs. Relaxing the time-independence assumption is an important contribution of this work with a view to application domains with highly correlated edges. For example, in a computer network, knowing that two machines are currently connected means they are more likely to be connected in the near future. Moreover, as we show in this article, some community structures can only be detected by taking account of the temporal dependencies in the network dynamics. Finally, the continuous-time model handles irregularly observed or incomplete data far more easily than its discrete-time counterparts.

The remainder of the paper is organised as follows: in Section 2 we introduce the autoregressive stochastic block model (ARSBM), a time-dependent extension of the SBM. This includes the model definition of the process governing when nodes change community membership together with the autoregressive model for edge processes. Due to the complexity added by the continuous-time setting, knowledge of some edge states is needed at the changepoints, where individuals change community membership. To overcome this, an augmentation scheme is presented to aid inference for the ARSBM within a Bayesian framework. Since the number of changepoints is assumed to be unknown and the number of parameters of the ARSBM depends on the number of changepoints then a reversible-jump Markov chain Monte Carlo (RJMCMC) sampling scheme can be used to draw samples from the posterior distribution on the number of changepoints. In Section 3 an RJMCMC sampling scheme is described for the ARSBM. Whilst the primary focus of the paper is on the movements between communities, a useful by-product of the RJMCMC is an efficient algorithm for estimating the underlying network parameters. The performance of the RJMCMC sampler is sensitive to the initial community assignments and to combat this we give an effective mechanism for the initial assignment of nodes to communities in Section 4. In Sections 5 and 6 the RJMCMC sampling scheme is demonstrated on simu-

lated data sets and a data set involving monitoring social behaviour in mice (Lopes et al 2016b), respectively. Finally, in Section 7 we make some concluding remarks concerning directions for future research in this area.

2 The autoregressive stochastic block model

2.1 Model

The autoregressive stochastic block model (ARSBM) is built on a hierarchical structure as follows. Suppose a dynamic network consists of a fixed set of nodes, V ($|V| = N$), partitioned into a fixed number of communities, K. The community membership of the N nodes is modelled using N independent and identically distributed community membership processes. Let $C_i(\cdot)$ denote the community membership process for node i. It is assumed that $C_i(\cdot)$ is a continuous-time Markov chain (CTMC) (Norris 1997), which takes values in $\{1, 2, \ldots, K\}$, with $C_i(t) = k$ meaning that individual i is in community k at time t. We assume that, regardless of the current community to which it belongs, a node spends $\text{Exp}(\lambda)$ time in the community before moving to a new community chosen uniformly at random from the remaining communities. (This assumption can easily be relaxed.) Therefore the generator matrix for the CTMC governing $C_i(\cdot)$ has diagonal elements equal to $-\lambda$ and off-diagonal elements equal to $\lambda/(K-1)$. Using properties of CTMCs, the number of times node i changes community, $M_i \sim \text{Po}(\lambda)$ with the times of the changes $\boldsymbol{\tau}_i = (\tau_i^1, \ldots, \tau_i^{M_i})$ being ordered and uniformly distributed on $[t_0, t_T]$. The new community level, $C_i(\tau_i^d)$, is drawn uniformly at random from $\{k \neq c_i(\tau_i^{d-1}) : k = 1, \ldots\ K\}$, and individual i remains in that community until τ_i^{d+1}. Throughout we denote the stochastic process by $C_i(t)$ and a given realisation at time t by $c_i(t)$.

In the SBM the probability that an edge exists between two nodes depends only upon the communities to which the two nodes belong. In the ARSBM, we employ a similar model hierarchy with edge dynamics only depending upon the communities to which the two nodes belong. We introduce an autoregressive component which allows the state of the edge to switch "on" or "off" with Markovian dynamics. We make the additional assumption that all edges with end-nodes in different communities have similar dynamics, although this can easily be relaxed. Under this setting, there will be $K+1$ processes to govern the dynamics of edges in the network: one process for each community k (governing the edges (i, j) with $C_i(t) = C_j(t) = k$) and one process for edges between communities (governing the edges (i, j) where $C_i(t) \neq C_j(t)$). This reduces the number of parameters from $\mathcal{O}(K^2)$ to $\mathcal{O}(K)$.

In order to model edge dynamics, we first define the community membership of the edge, which is a deterministic function of the community membership of the end-nodes. Specifically, for the edge between nodes i and j, its community membership process $C_{ij}(\cdot)$ is defined to be k if both i and j are in community k and 0 otherwise, as in Equation (1).

$$C_{ij}(t) = \begin{cases} C_i(t) & \text{if } C_i(t) = C_j(t), \\ 0 & \text{if } C_i(t) \neq C_j(t), \end{cases} \qquad (1)$$
$$C_{ij}(t) \in \{0, 1, \ldots, K\}.$$

Since both $C_i(\cdot)$ and $C_j(\cdot)$ are piecewise constant processes, then $C_{ij}(\cdot)$ is a piecewise constant process with $M_{ij} \leq M_i + M_j$ step changes. Let $E_{ij}(\cdot)$ denote the edge status process for the edge between nodes i and j. Specifically, $E_{ij}(t) = 1$ if an edge exists ("on") between nodes i and j at time t and $E_{ij}(t) = 0$ if no edge exists ("off") between nodes i and j at time t. The edge process is assumed to follow a piecewise time-homogeneous CTMC. That is, whilst the $C_{ij}(t) = k$, the generator matrix for the edge process is

$$G(k) = \begin{pmatrix} -\alpha_k & \alpha_k \\ \delta_k & -\delta_k \end{pmatrix}.$$

The transition rates α_k, referred to as the *appearance rates*, govern the rate at which an edge appears (transitions from state 0 to 1) whilst in community k. Similarly, the transition rates δ_k are referred to as *deletion rates* and govern the reverse transition from state 1 to 0. Throughout, we denote the stochastic process by $E_{ij}(t)$ and a given realisation at time t by $e_{ij}(t)$.

Let $\pi_k = \alpha_k/(\alpha_k + \delta_k)$, the stationary probability of an edge being on in community k. This allows for a direct comparison with the static SBM. Furthermore, let $\rho_k = \alpha_k + \delta_k$ be the combined rate of change for the edge process with $\alpha_k = \pi_k \rho_k$ and $\delta_k = (1 - \pi_k)\rho_k$. It is helpful to use the parameterisation $\boldsymbol{\pi} = (\pi_0, \pi_1, \ldots, \pi_K)$ and $\boldsymbol{\rho} = (\rho_0, \rho_1, \ldots, \rho_K)$ for modelling the ARSBM.

2.2 Posterior distribution

We are now in position to construct the likelihood for the data and the posterior distribution of the parameters and community membership of the nodes.

Suppose that *network snapshots* of $\mathcal{N}$ are collected at time points $\mathbf{t} = (t_0, t_1, \ldots, t_T)$ in the observation interval $[t_0, t_T]$. In this way, the states $e_{ij}(t_s)$ are observed for $s = 0, 1, \ldots, T$ and $i \neq j \in \{1, \ldots, N\}$. For brevity, we let $e_{ij}^s = e_{ij}(t_s)$ be the state of the edge

between nodes i and j at the s^{th} observation. Similarly, $c_i^s = c_i(t_s)$ is the community membership of node i at observation time s; however, this is a latent variable. We also let $\Delta_s = t_s - t_{s-1}$ be the amount of time between observations $s-1$ and s. Let $\mathbf{e(t)} = \{e_{ij}^s | 1 \leq i < j \leq N, s = 0, 1, \ldots, T\}$ denote the set of all network snapshot data. Let $\mathbf{c}_i(\mathbf{t}) = \{c_i^s | s = 0, 1, \ldots, T\}$, the community membership of node i at every observation time, with $\mathbf{c(t)} = \{\mathbf{c}_i(\mathbf{t}) | i = 1, 2, \ldots, N\}$, the set of all community memberships. We are interested in the joint posterior distribution, which can be decomposed into the product of the observation likelihood, the distribution of the evolution of community assignments and a prior distribution on the parameters as in Equation (2).

$$\begin{aligned}\pi(\boldsymbol{\theta}, \mathbf{c(t)} | \mathbf{e(t)}) &\propto \pi(\mathbf{e(t)}, \mathbf{c(t)} | \boldsymbol{\theta}) \pi(\boldsymbol{\theta}) \\ &= \pi(\mathbf{e(t)} | \mathbf{c(t)}, \boldsymbol{\theta}) \pi(\mathbf{c(t)} | \boldsymbol{\theta}, \mathbf{c}(t_0)) \pi(\boldsymbol{\theta}, \mathbf{c}(t_0)),\end{aligned} \tag{2}$$

where $\boldsymbol{\theta} = (\lambda, \boldsymbol{\pi}, \boldsymbol{\rho})$ and $\mathbf{c}(t_0) = (c_1^0, c_2^0, \ldots, c_N^0)$. Note the dependence on the initial community structure.

We now provide equations for each term in Equation (2). Firstly, in Equation (3), the likelihood of the observed edge sequence, given the latent community memberships and model parameters, is computed.

$$\pi(\mathbf{e(t)} | \mathbf{c(t)}, \boldsymbol{\theta}) = \prod_{\substack{s,i \\ j \neq i}} \pi(e_{ij}^s | e_{ij}^{s-1}, c_i^{s-1}, c_i^s, c_j^{s-1}, c_j^s) \tag{3}$$

The computation of each factor in Equation (3) is non-trivial since it requires integrating over all possible community membership processes for all nodes between the times t_{s-1} and t_s. Since each community membership process is piecewise constant, it is sufficient to know the times of the changepoints in node i's community membership, $\boldsymbol{\tau}_i = (\tau_i^1, \ldots, \tau_i^{M_i})$, and the community membership of the nodes at the changepoints, $\mathbf{c}_i(\boldsymbol{\tau}_i)$ with $\mathbf{c}(\boldsymbol{\tau}) = (\mathbf{c}_1(\boldsymbol{\tau}_1), \ldots, \mathbf{c}_N(\boldsymbol{\tau}_N))$. Note that $\mathbf{c}_{ij}(\boldsymbol{\tau}_{ij})$ is a deterministic function of $\mathbf{c}_i(\boldsymbol{\tau}_i)$ and $\mathbf{c}_j(\boldsymbol{\tau}_j)$, where $\boldsymbol{\tau}_{ij} = (\tau_{ij}^1, \ldots, \tau_{ij}^{M_{ij}})$ is the set of combined changepoints in nodes i and j community memberships.

Therefore, given that the edge dynamics are governed by a CTMC with piecewise constant dynamics then, if $c_{ij}(t) = k$ for all time $t \in [t_{s-1}, t_s)$, then

$$\mathrm{P}\left(e^s = 1 | e^{s-1}, c(t)\right) = \pi_k + (e^{s-1} - \pi_k) e^{-\rho_k \Delta_s}, \tag{4}$$

where we drop the subscript ij for brevity.

The calculation of the probability of an edge being in state 1 becomes more involved if there is a change in community membership of the edge during an interval $[t_{s-1}, t_s)$. It is straightforward, in principle at least, to compute $\mathrm{P}\left(e_{ij}^s = 1 | e_{ij}^{s-1}, c_{ij}(t)\right)$ by summing over the possible states of the edge ij at each of the changepoints in the interval $[t_{s-1}, t_s)$. Specifically, if $\tau \in [t_{s-1}, t_s]$ is a changepoint with $c_{ij}(t) = k$ for $t \in [\tau, t_s)$, then

$$\begin{aligned}&\mathrm{P}\left(e^s = 1 | e^{s-1}, c(t)\right) \\ &= \sum_{l=0}^{1} \mathrm{P}(e^s = 1 | e(\tau) = l) \, \mathrm{P}\left(e(\tau) = l | e^{s-1}\right) \\ &= \sum_{l=0}^{1} \{\pi_k + (l - \pi_k) e^{-\rho_k (t_s - \tau)}\} \mathrm{P}\left(e(\tau) = l | e^{s-1}\right),\end{aligned} \tag{5}$$

where again, we drop the subscript ij for brevity.

Whilst it is possible to compute $\pi(\mathbf{e(t)} | \mathbf{c(t)}, \mathbf{c}(\boldsymbol{\tau}), \boldsymbol{\theta})$ from (5), it is far simpler to augment the data with $\mathbf{e}(\boldsymbol{\tau}) = \{e_{ij}(\tau_{ij}^d); 1 \leq i, j \leq N, d = 1, 2, \ldots, M_{ij}\}$. Let $\boldsymbol{\sigma}_i = \mathbf{t} \cup \boldsymbol{\tau}_i$, the ordered times at which the edges are observed or node i changes community membership. Similarly, let $\boldsymbol{\sigma}_{ij} = \boldsymbol{\sigma}_i \cup \boldsymbol{\sigma}_j$ denote the ordered times at which edge (i, j) is observed or changes community membership and contains $T_{ij} = T + M_{ij}$ elements. Thus, the likelihood of the observed and augmented edges, given the community structure, $\pi(\mathbf{e}(\boldsymbol{\sigma}) | \mathbf{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}, \boldsymbol{\theta})$ becomes

$$\prod_{i \neq j} \prod_{d=0}^{T_{ij}-1} \mathrm{P}\left(e_{ij}(\sigma_{ij}^{d+1}) | e_{ij}(\sigma_{ij}^d), c_{ij}(\sigma_{ij}^d)\right) \tag{6}$$

where, by letting $\Delta_{d+1} = (\sigma^{d+1} - \sigma^d)$, the factors can be written as:

$$\begin{aligned}&\left((1 - \pi_{c(\sigma^d)}) - (e(\sigma^d) - \pi_{c(\sigma^d)}) e^{-\rho_{c(\sigma^d)} \Delta_{d+1}}\right)^{1 - e(\sigma^{d+1})} \\ &\times \left(\pi_{c(\sigma^d)} + (e(\sigma^d) - \pi_{c(\sigma^d)}) e^{-\rho_{c(\sigma^d)} \delta_{d+1}}\right)^{e(\sigma^{d+1})}\end{aligned}$$

The computation of $\pi(\mathbf{c}(\boldsymbol{\sigma}), \boldsymbol{\tau} | \boldsymbol{\theta}, \mathbf{c}(t_0))$ is straightforward. Firstly, $\boldsymbol{\sigma}_i$ is deterministic given $\boldsymbol{\tau}_i$, so

$$\begin{aligned}&\pi(\mathbf{c}(\boldsymbol{\sigma}), \boldsymbol{\tau} | \boldsymbol{\theta}, \mathbf{c}(t_0)) \\ &= \prod_{i=1}^{N} \pi(\mathbf{c}_i(\boldsymbol{\tau}_i) | \boldsymbol{\tau}_i, \mathbf{c}(t_0)) \pi(\boldsymbol{\tau}_i | \lambda) \\ &= \prod_{i=1}^{N} \pi(\mathbf{c}_i(\boldsymbol{\tau}_i) | \boldsymbol{\tau}_i, \mathbf{c}(t_0)) \pi(\boldsymbol{\tau}_i | M_i) \pi(M_i | \lambda) \\ &= \prod_{i=1}^{N} \left(\frac{1}{k-1}\right)^{M_i} \times \frac{M_i!}{(t_T - t_0)^{M_i}} \\ &\quad \times \frac{\{\lambda(t_T - t_0)\}^{M_i}}{M_i!} \exp(-\lambda(t_T - t_0)) \\ &= \left(\frac{1}{k-1}\right)^{M} \lambda^M \exp(-\lambda N (t_T - t_0)),\end{aligned} \tag{7}$$

where $M = \sum_{i=1}^{N} M_i$ is the total number of changepoints. The three components on the right-hand side of (7) for node i correspond to; the density of the ordered M_i time points, the probability of the group transitions

which take place and the probability that there are M_i changes in node i's community membership.

Combining (6) and (7), we have an expression for $\pi(\mathbf{e}(\boldsymbol{\sigma}), \mathbf{c}(\boldsymbol{\sigma})|\boldsymbol{\theta}, \mathbf{c}(t_0))$, and therefore, an explicit expression for the right hand side of

$$\begin{aligned}\pi(\mathbf{c}(\boldsymbol{\sigma}), \mathbf{e}(\boldsymbol{\tau}), \boldsymbol{\tau}, \boldsymbol{\theta}|\mathbf{e}(\mathbf{t})) &\propto \pi(\mathbf{e}(\boldsymbol{\sigma})|\mathbf{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}, \boldsymbol{\theta}) \\ &\times \pi(\mathbf{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}|\mathbf{c}(t_0), \boldsymbol{\theta}) \times \pi(\mathbf{c}(t_0), \boldsymbol{\theta}).\end{aligned} \tag{8}$$

2.3 Identifiability

An important point to consider before introducing the RJMCMC sampler is the identifiability of the model. As is well known for SBMs, the parameters can only be obtained up to a label switching of the group nodes (Matias and Miele 2016). Letting $\rho_k \downarrow 0$ for $k = 0, 1, \ldots, K$ whilst keeping $\boldsymbol{\pi}$ fixed results in λ being unidentifiable. This is because the graph does not change through time and hence $\mathbf{E}(t) = \mathbf{E}(s)$ for all $0 \leq s < t$. Therefore the graph dynamics are invariant to how fast (or slow) the nodes switch between blocks since after the initial configuration, the block to which a node belongs becomes irrelevant. More generally, we observe that the dependence parameter ρ_k enters the likelihood through $\exp(-\rho_k \Delta_s))$, see Equation (4), and robust estimation of ρ_k is obtained when $\exp(-\rho_k \Delta_s))$ is not close to 0 (independence) or 1 (full dependence).

The graph parameters become unidentifiable as $\lambda \to \infty$, that is the nodes are constantly switching between blocks. In this case, for each $k, l = 1, 2, \ldots, K$, the nodes i and j will spend a proportion $1/K^2$ time in blocks k and l, respectively during any period of time. Consequently, regardless of the value of K, as $\lambda \to \infty$, the dynamic SBM resembles an SBM with a single block model, a dynamic Erdös-Rényi random graph, with stationary probability of an edge π_* and rate of change ρ_*, where

$$\rho_* = \frac{1}{K^2} \sum_{k=1}^{K} \rho_k + \frac{K-1}{K} \rho_0, \tag{9}$$

and

$$\pi_* = \frac{1}{\rho_*} \left\{ \frac{1}{K^2} \sum_{k=1}^{K} \rho_k \pi_k + \frac{K-1}{K} \rho_0 \pi_0 \right\}. \tag{10}$$

Letting $\lambda \to \infty$ removes any dependence in block membership of a node from one time point to the next. This is linked to the observation in Matias and Miele (2016) for the discrete time SBM models of Xu and Hero (2014) and Matias and Miele (2016) that independence in block membership from one time point to the next leads to non-identifiability of the parameters.

If $\lambda = 0$ and $(\rho_k, \pi_k) = (\rho_I, \pi_I)$ $(k = 1, 2, \ldots, K)$ (a dynamic affiliation model), then, following the approach of Frank and Harary (1982) and Allman et al (2011), it is straightforward to show that $E[E_{12}(0)]$ and $E[E_{12}(0)E_{13}(0)E_{23}(0)]$ give (π_I, π_0), as in the case of the static SBM. Moreover, considering $E[E_{12}(0)E_{12}(t)]$ and $E[E_{12}(0)E_{13}(0)E_{23}(0)E_{12}(t)E_{13}(t)E_{23}(t)]$ for some $t > 0$ is then sufficient to identify (ρ_I, ρ_0). By considering edge moments involving 4 nodes, we can show that this extends to small positive $\lambda > 0$ by ignoring $o(\lambda)$ terms. A further discussion of parameter identifiability is beyond the scope of this paper but note that we observe parameter estimation is robust to starting values in the simulations and application data set up to permutation of block labels, for moderate, positive ρ_k and small, positive λ.

3 Reversible jump MCMC

3.1 Sampling scheme

In this section a RJMCMC (reversible jump MCMC) algorithm is described for obtaining samples from the joint posterior distribution of $\boldsymbol{\theta} = (\lambda, \boldsymbol{\pi}, \boldsymbol{\rho})$ and $\boldsymbol{c}(\mathbf{t})$ given $\mathbf{e}(\mathbf{t})$ using (8) and data augmentation of the values $(\boldsymbol{\tau}, \mathbf{c}(\boldsymbol{\tau}), \mathbf{e}(\boldsymbol{\tau}))$. The updating of the parameters $\lambda, \boldsymbol{\pi}$ and $\boldsymbol{\rho}$ given $(\boldsymbol{\tau}, \mathbf{c}(\boldsymbol{\tau}), \mathbf{e}(\boldsymbol{\tau}))$ is straightforward using (6) and (7). Updating $\boldsymbol{\tau}_i$ and the associated augmented data is more involved as M_i, the number of elements in $\boldsymbol{\tau}_i$, is unknown. This naturally leads to a reversible jump sampler (Green 1995) to explore parameter spaces of differing dimensions.

An overview of the sampling scheme is given in Algorithm 1. For each step of the sampler, each of the parameters $\lambda, \boldsymbol{\pi}, \boldsymbol{\rho}$ and $\boldsymbol{\tau}$ (and $\boldsymbol{M} = (M_1, M_2, \ldots, M_N)$) are updated in turn.

By assigning a Gamma$(\lambda_{01}, \lambda_{02})$ prior to λ, it follows from (7) that, $\lambda|\boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\tau}, \mathbf{c}(\boldsymbol{\sigma}), \mathbf{e}(\boldsymbol{\sigma})$ is distributed as Gamma $(\lambda_{01} + M, \lambda_{02} + N(t_T - t_0))$.

For $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$ there is no closed form conditional distribution and for this reason, a random walk is proposed. Since π_k is bounded on $[0, 1]$, a random walk is proposed on a logit scale, $\text{logit}(\pi_k^*) \sim N(\text{logit}(\pi_k), \sigma_\pi^2)$. As for ρ_k, a random walk on the log scale is proposed, since $\rho_k > 0$, with $\log(\rho_k^*) \sim N(\log(\rho_k), \sigma_\rho^2)$. The priors for π_k and ρ_k are Beta and Gamma distributions respectively. By performing random walk updates on transformed scales, we need to take account of the proposal densities with

$$q(\pi_k^*|\pi_k) = \frac{\phi(\text{logit}(\pi_k^*)\,|\text{logit}(\pi_k), \sigma_\pi^2)}{\pi_k^*(1 - \pi_k^*)}, \tag{11}$$

Algorithm 1 RJMCMC Sampler

Inputs: parameters for Gamma prior for λ, prior distributions for π and ρ, nRuns and burn-in.
Draw λ, π, ρ from their respective priors. Set $M = 0$ and $\boldsymbol{\tau} = \emptyset$.
for h=1,...,nRuns **do**
  Draw $\lambda^{(h+1)}$ from its conditional distribution.

  for k in $1, \ldots, C$ **do**
    Propose $\pi_k^{(h+1)}$ by taking a random walk on the logit scale from $\pi_k^{(h)}$.
    Propose $\rho_k^{(h+1)}$ by taking a random walk on the log scale from $\rho_k^{(h)}$.
  if There are no changes in the current sampler state **then**
    Propose inserting a change
  else Draw X uniformly at random from $\{1, 2\}$
    if X=1 **then**
      Propose inserting a new change to the current state, with augmented edge states as required.
    else X=2
      Propose deleting a change from the current state, removing and adding affected augmented edge states as required.
  Given that $M > 0$, propose moving each changepoint into an adjacent observation interval and using a Gaussian random walk proposal.
  Resample the augmented edges.
Discard samples 1,...,burn-in.

and

$$q(\rho_k^*|\rho_k) = \frac{\phi(\log(\rho_k^*)\,|\log(\rho_k)\,, \sigma_\rho^2)}{\rho_k}, \tag{12}$$

where $\phi(y; \mu, \sigma^2)$ denotes the probability density function of a $N(\mu, \sigma^2)$ evaluated at y.

In this work, an adaptive scheme is used to adjust the variance of the proposal distributions to improve the efficiency of the sampler. The proposal variances σ_π^2 and σ_ρ^2 are set using an adaptive procedure as in Xiang and Neal (2014). By Roberts et al (1997), an acceptance rate of approximately 25% is optimal for random walk Metropolis sampling. To achieve this rate, a proposal variance σ^2 is adjusted at each step during the burn-in period by

$$\sigma_{h+1}^2 = \begin{cases} \sigma_h^2 \left(1 - \frac{\epsilon}{\sqrt{h}}\right) & \text{if move rejected,} \\ \sigma_h^2 \left(1 + \frac{3\epsilon}{\sqrt{h}}\right) & \text{if move accepted,} \end{cases}$$

where the step size ϵ is chosen as input.

3.2 Updating $(\boldsymbol{\tau}, \mathbf{c}(\boldsymbol{\tau}), \mathbf{e}(\boldsymbol{\tau}))$

The trans-dimensional sampler for updating $\boldsymbol{\tau}$, and consequently, $(\mathbf{c}(\boldsymbol{\tau}), \mathbf{e}(\boldsymbol{\tau}))$ is now described. These constitute birth-death moves: inserting a changepoint (insert a change of community membership in a node) and removing a changepoint (removes one of the changes from the current state of the sampler). In each iteration of the algorithm only one move is attempted. In the case that the current sampler state contains no changepoints, then an insert move is attempted. Otherwise, the insert move is chosen with probability 0.5. In addition, we propose moving the time of existing changepoints to obtain a posterior distribution of changepoint locations.

We begin by describing the process for proposing to insert a changepoint. Firstly a node i is chosen uniformly at random from $1, \ldots, N$ and a time τ^* is chosen uniformly at random from the interval $[t_0, t_T]$. This amounts to adding a step change at time τ^* in $C_i(\cdot)$. Let k denote the community membership of node i at time τ^* prior to the proposed addition of a changepoint at time τ^*. Since the initial community memberships are unknown, the sampler allows for adjusting $C_i(\cdot)$ either prior to, or after, time τ^*. A proposal "forwards" in time proposes a new community k^* and sets $C_i(\tau^*) = k^*$. Conversely, a proposal "backwards" in time proposes a new community k^* for the interval *preceding* τ^*. If this previous interval starts at time σ^*, then the sampler sets $C_i(\sigma^*) = k^*$, where $\sigma^* = t_0$ if there are no previous changes in node i's community membership with $C_i(\tau^*) = k$. See Figure 1 for an example of possible insert moves.

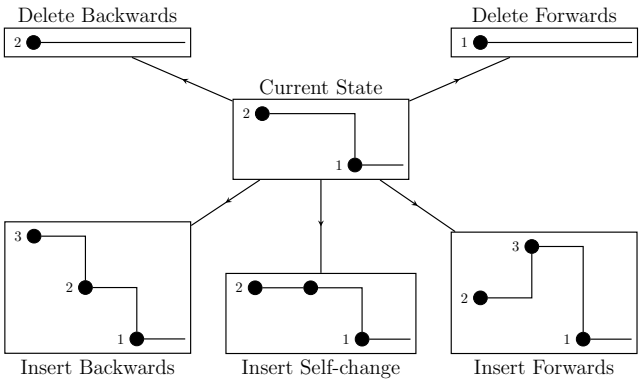

Fig. 1 Possible moves to insert or delete a changepoint for a node which currently has one change. After choosing to insert or delete, a model is proposed proportional to the likelihood.

To allow the sampler to explore the parameter space more freely, we allow the possibility of self-changes in community membership. That is, a change in which node i moves from community k to community k at time τ^*. Such changes are artificial and are used purely to allow the sampler to explore the parameter space. The directionality ("forwards" or "backward" in time) of

such a change is irrelevant since inserting a self change is symmetric in time.

There are therefore $2K-1$ ways to propose inserting a change in community membership at time τ^* for node i. Rather than drawing a change in community membership uniformly at random from the possibilities we consider the relative likelihood of the $2K-1$ changes in community membership and propose a change accordingly. In order to do this, we consider the set of edges affected by each of the proposed community changes. In all cases the unobserved states of edges affected by the change are a subset of $\mathbf{E}_i(\tau^*) = (E_{i1}(\tau^*), E_{i2}(\tau^*), \ldots, E_{iN}(\tau^*))$. For an edge (i,j) affected by the change in community membership, we augment the state space with $e_{ij}(\tau^*)$ and set $e_{ij}(\tau^*) = 1$ with probability,

$$P(e_{ij}(\tau^*) = 1 | e_{ij}(\sigma^* \wedge t^*), c_{ij}(\sigma^*) = \kappa, \pi_\kappa, \rho_\kappa), \qquad (13)$$

where t^* denotes the last observation prior to τ^*. Let $\mathcal{A}^{k_1,k_2}$ denote the set of additional edges proposed with the move to $c_i(\sigma^*) = k_1$ and $c_i(\tau^*) = k_2$, where at least one of k_1 or k_2 is equal to k. Let $\mathcal{A}^* = \cup_{k_1,k_2}\mathcal{A}^{k_1,k_2}$ and note that edge (i,j) can be included in more than one $\mathcal{A}^{k_1,k_2}$ with different values for $e_{ij}(\tau^*)$. We choose to move to community memberships $c_i(\sigma^*) = k_1$ and $c_i(\tau^*) = k_2$ for node i with probability

$$\frac{\mathrm{P}\left(\mathcal{A}^{k_1,k_2}|\boldsymbol{\theta}, \tau^*\right)}{\sum_{l_1,l_2}\mathrm{P}(\mathcal{A}^{l_1,l_2}|\boldsymbol{\theta}, \tau^*)}. \qquad (14)$$

Therefore the proposal distribution for the proposed changepoint in node i's community membership and $\mathcal{A}^*$ is

$$\frac{\mathrm{P}(M+1|M)}{N(t_T - t_0)} \cdot \mathrm{P}(\mathcal{A}^*|\boldsymbol{\theta}, \tau^*) \cdot \frac{\mathrm{P}\left(\mathcal{A}^{k_1,k_2}|\boldsymbol{\theta}, \tau^*\right)}{\sum_{l_1,l_2}\mathrm{P}(\mathcal{A}^{l_1,l_2}|\boldsymbol{\theta}, \tau^*)}. \qquad (15)$$

The reverse move is the deletion of a changepoint for which we require $M > 0$. Firstly, we select a changepoint τ^* to delete uniformly at random. Suppose that the changepoint occurs in node i's community membership. Suppose that σ^* denotes the previous changepoint in node i prior to τ^* and that $c_i(\sigma^*) = k_1$ and $c_i(\tau^*) = k_2$, then there are two choices (unless $k_1 = k_2$, τ^* represents a self-change), either set $c_i(\sigma^*) = k_1$ (change the future community membership from time τ^*) or $c_i(\sigma^*) = k_2$ (change the community membership prior to time τ^*). For both of these proposed changes it is possible that the set of augmented edges required changes at σ^* when setting $c_i(\sigma^*) = k_2$ and the changepoint in node i, should one exist, after τ^*. Let $\mathcal{B}^{k_1}$ and $\mathcal{B}^{k_2}$ denote the additional augmented edges required when setting $c_i(\sigma^*) = k_1$ and $c_i(\sigma^*) = k_2$, respectively. For generating edges in $\mathcal{B}^{k_l}$ $(l = 1, 2)$, we take the same approach as when inserting a changepoint simulating forward the state of an edge by modifying (13) to propose the edge state at the required time. Then we set $c_i(\sigma^*) = k_l$ $(l = 1, 2)$ with additional augmented edges $\mathcal{B}^{k_l}$ with probability

$$\frac{\mathrm{P}\left(\mathcal{B}^{k_l}|\boldsymbol{\theta}\right)}{\mathrm{P}(\mathcal{B}^{k_1}|\boldsymbol{\theta}) + \mathrm{P}(\mathcal{B}^{k_2}|\boldsymbol{\theta})}. \qquad (16)$$

Therefore, the proposal distribution for the proposed deletion of changepoint τ^* with associated changes and $\mathcal{B}^* = \mathcal{B}^{k_1} \cup \mathcal{B}^{k_2}$ is

$$\frac{\mathrm{P}(M-1|M)}{M} \cdot \mathrm{P}(\mathcal{B}^*|\boldsymbol{\theta}) \cdot \frac{\mathrm{P}\left(\mathcal{B}^{k_l}|\boldsymbol{\theta}\right)}{\mathrm{P}(\mathcal{B}^{k_1}|\boldsymbol{\theta}) + \mathrm{P}(\mathcal{B}^{k_2}|\boldsymbol{\theta})}. \qquad (17)$$

The generating of $\mathcal{A}^*$ and $\mathcal{B}^*$ in the above procedures are simply to assist with choosing community membership in an informed way and play no role in the posterior distribution (parameters and augmented states) once a set of augmented edges have been chosen. Therefore, we would ideally want to integrate out $\mathcal{A}^*$ and $\mathcal{B}^*$. This can effectively be done by working on an expanded state space incorporating all the possible community membership states of the nodes and all possible edge states. In this way we can show that the probability of accepting a proposed move to insert a changepoint in community i at time t^* is

$$\begin{aligned} &\frac{\pi(\mathbf{e}(\boldsymbol{\sigma}'), \mathbf{c}(\boldsymbol{\tau}'), \boldsymbol{\tau}', \boldsymbol{\theta}|\mathbf{e}(\mathbf{t}))}{\pi(\mathbf{e}(\boldsymbol{\sigma}), \mathbf{c}(\boldsymbol{\tau}), \boldsymbol{\tau}, \boldsymbol{\theta}|\mathbf{e}(\mathbf{t}))} \\ &\times \frac{P(M|M+1)N(t_T - t_0)}{P(M+1|M)(M+1)} \times \frac{\mathrm{P}(\mathcal{B}^*|\boldsymbol{\theta})}{\mathrm{P}(\mathcal{A}^*|\boldsymbol{\theta}, \tau^*)} \\ &\times \frac{\mathrm{P}\left(\mathcal{B}^{k_l}|\boldsymbol{\theta}\right)\sum_{l_1,l_2}\mathrm{P}\left(\mathcal{A}^{l_1,l_2}|\boldsymbol{\theta}, \tau^*\right)}{\mathrm{P}(\mathcal{A}^{k_1,k_2}|\boldsymbol{\theta}, \tau^*)\,\mathrm{P}(\mathcal{B}^{k_1}|\boldsymbol{\theta}) + \mathrm{P}(\mathcal{B}^{k_2}|\boldsymbol{\theta})} \end{aligned} \qquad (18)$$

where $\boldsymbol{\sigma}' = \boldsymbol{\sigma} \cup \tau^*$ and $\tau' = \tau \cup \tau^*$. The acceptance probability for deleting a changepoint is the reciprocal of (18).

The two procedures for moving a changepoint are straightforward. Firstly, each changepoint is moved at random either to the next observation interval or the previous observation interval. Secondly, the time of a changepoint is perturbed using a random walk move with a Gaussian proposal. The first such move allows for large changes in the location of a changepoint while the second allows for small, local moves refining the position of the changepoint. Suppose that the changepoint to be adjusted is τ which lies in the interval $[t_n, t_{n+1}]$. We propose a new time τ^* to lie in one of the intervals immediately before or after $[t_n, t_{n+1}]$. We propose that τ^* is positioned in the proposed interval proportional to the location of τ in the current interval such that:

$$\tau^* = \begin{cases} t_{n-1} + (t_n - t_{n-1})\frac{\tau - t_n}{t_{n+1} - t_n} & \text{w.p } 0.5 \\ t_{n+1} + (t_{n+1} - t_{n+2})\frac{\tau - t_n}{t_{n+1} - t_n} & \text{w.p } 0.5. \end{cases}$$

The second move allows for refinement of such times by making a small change in location of τ using a standard Metropolis-like move. Specifically a value τ^* is proposed via $\tau^* = \tau + N(0, \sigma_\tau)$ for σ_τ small.

Finally, each augmented edge states $A \in \mathcal{A}^*$ is resampled proportional to the relative likelihood using Equation (3) in the proposal distribution,

$$\mathrm{P}(A = 1) = \frac{\pi\left(A = 1, \mathbf{e}(\boldsymbol{\sigma})|\theta\right)}{\pi\left(A = 0, \mathbf{e}(\boldsymbol{\sigma})|\theta\right) + \pi\left(A = 1, \mathbf{e}(\boldsymbol{\sigma})|\theta\right)}.$$

In the case that a changepoint τ is close to an observation time t, the augmented edges at τ will most likely be resampled in the same state as at the observation time t.

4 Initialisation of sampler state

In this section some observations are made about the initial community membership vector $\boldsymbol{c}(t_0)$, which is key to the success of the sampler. Recall that the data $\mathbf{e}(\mathbf{t})$ concerns the state of edges in the network, which are assumed to be Markov chain distributed with parameters determined by the latent community membership of the end nodes. These community memberships are themselves Markov chain distributed conditional on the initial community assignment, $\boldsymbol{c}(t_0)$. This makes the initial community membership very influential on the entire model. As such, assigning nodes to the incorrect community can lead to poor estimates for parameters $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$, and slow convergence of the RJMCMC to the posterior distribution.

There are a number of possible ways to initialise $\boldsymbol{c}(t_0)$, the initial community membership. The simplest approach is to model the initial state using a static SBM to identify the initial block assignments. Given that a single snapshot of the ARSBM is informative about $\boldsymbol{\pi}$ but contains no information concerning $\boldsymbol{\rho}$, this works well if the π_ks $(k = 1, 2, \ldots, K)$ are significantly different from π_0. However, this approach fails if $\boldsymbol{\rho}$ is the primary determinant of block membership. Therefore, we propose and use throughout a robust approach based on clustering nodes using a distance metric. An alternative clustering using a Poisson SBM on the distances was also considered. In this case, the network snapshots were projected onto a matrix M^d with $M^d_{ij} = d(i, j)$ for each of the distances introduced in this section. Next, an SBM with Poisson emission distribution was fitted to each M^d to yield an initial assignment of nodes to communities labelled $\mathbf{c}^d$. Finally, the assignment with the highest likelihood (under the Poisson SBM) was chosen for the initialisation. The results for using a Poisson SBM are similar to the proposed clustering method; however, the clustering procedure is faster to compute.

The distance between two nodes is the weighted average of two measures. Firstly, $d_1(i, j)$ is the fraction of time that $e_{ij}(\cdot)$ is observed in the "on" state in the set of snapshots. Secondly, $d_2(i, j)$ is the number of times that $e_{ij}(\cdot)$ changes state in the set of snapshots. In essence, d_1 is a measure for $\boldsymbol{\pi}$ and d_2 is a measure for $\boldsymbol{\rho}$. The metric d is then a weighted average of these two distances as given in (19).

$$d(i, j) = \gamma d_1(i, j) + (1 - \gamma) d_2(i, j) \tag{19}$$

For networks where the community structure is more apparent in the ratio of edges within a community compared to the ratio of edges between communities, then setting $\gamma = 1$ in (19) gives a distance measure based only on this ratio. However, for networks where the community structure is embedded in the rate of transition of edge states, then $\gamma = 0$ is a more appropriate choice. This distance will work well in networks with disassortative community structures, since nodes which are less likely to be connected are close under this measure. Since no assumptions are made on the assortivity of a network, the distance used should not be fixed to one type of assortivity. A further three distances are used to measure the similarity of two nodes. All four distances are given in (20).

$$\begin{aligned}
d_{11}(i, j) &= \gamma_{11} d_1(i, j) + (1 - \gamma_{11}) d_2(i, j) \\
d_{10}(i, j) &= \gamma_{10} d_1(i, j) + (1 - \gamma_{10})(1 - d_2(i, j)) \\
d_{01}(i, j) &= \gamma_{01}(1 - d_1(i, j)) + (1 - \gamma_{01}) d_2(i, j) \\
d_{00}(i, j) &= \gamma_{00}(1 - d_1(i, j)) + (1 - \gamma_{00})(1 - d_2(i, j))
\end{aligned} \tag{20}$$

These distances are suited to different types of community structure. Firstly, d_{11} will minimise the distance between nodes in the same community in a network which is disassortative in both the fraction of edges and the number of times edges change state. Such networks have few edges between nodes in the same community, but such edges are persistent across time. Next, d_{10} will minimise the distance between nodes in the same community in a network which is disassortative in the fraction of edges and assortative in the number of times edges change state. Such networks have few edges between nodes in the same community and such edges change often. By contrast, d_{01} will minimise the distance between nodes in the same community in a network which is assortative in the fraction of edges and disassortative in the number of times edges change state. Such networks have more edges between nodes in the same community and such edges are persistent in time compared to edges between communities which are fewer in number and change more frequently. Finally, d_{00} will minimise the distance between nodes in the same community in a network which is assortative

in both the fraction of edges and the number of times edges change state. Such networks have more edges between nodes in the same community compared to edges between communities which are fewer in number, however the edges between communities are more persistent than edges within communities.

Using these distances, the k-means algorithm (Lloyd 1982) can be used to cluster the nodes. A good clustering should separate nodes which are in different communities. Based on this idea, the k-means algorithm aims to put nodes which are far apart under d into different communities. As a result, a measure for a good clustering is the ratio R of squared distances between nodes in different communities to the total squared distance between all nodes. The higher this ratio, the more separated the clusters are.

To set the initial community assignments $\boldsymbol{c}(t_0)$, the network is measured using each of the distances in (20). Each γ parameter is set by maximising R for each d by clustering the nodes using k-means. This gives four clustering which are respectively optimal under each distance. The clustering used to initialise $\boldsymbol{c}(t_0)$ is then chosen as the clustering which maximises R among these four clusterings. This procedure is very quick compared to the RJMCMC sampling scheme.

5 Simulation study

In order to assess the performance of the RJMCMC sampler, we conducted a simulation study over a range of parameter combinations. There are eight parameters which we varied and for each parameter we considered two settings (Low, High) giving $2^8 = 256$ parameter combinations. The parameter combinations are the number of nodes N, the number of communities, C, the size of each community n_c, the expected number of changes $E[M]$ (the rate of nodes moving λ) and the community parameters π and ρ. For the community parameters, we set all the within-community parameters to be the same, that is, for all $i, j = 1, 2, \ldots, C$, $\pi_i = \pi_j$ and $\rho_i = \rho_j$. The parameter values are given in Table 1. For equal community sizes, N/C nodes were placed in each community. The sizes of communities for other simulations are given in Table 2. We ran simulation for all parameter combinations with the exception of $\pi_k = \pi_0$ and $\rho_k = \rho_0$, where the resulting network is indistinguishable from a dynamic Erdös-Rényi random graph (a stochastic block model with only one community). This yielded 192 simulated data sets, each consisting of 30 snapshots of the network equally spaced in time.

The RJMCMC described in Section 3 was applied to each simulated data set for $H = 20,000$ steps. The prior

Table 1 Parameter settings for simulation study.

Parameter	Low	High
N	72	120
C	3	6
$E[M]$	$0.3N$	$1.0N$
π_k	0.1	0.5
π_0	0.1	0.5
ρ_0	0.2	1.2
ρ_k	0.2	1.2
n_c	Equal	Unequal

Table 2 Number of nodes per community for $n_c = unequal$.

N	$C = 3$	$C = 6$
72	12, 24, 36	7, 9, 11, 13, 15, 17
120	20, 40, 60	10, 14, 18, 22, 26, 30

distributions for λ, π and ρ were set as Gamma(1,1), Beta(1,1) and Gamma(2,1) respectively. The algorithm was initialised with no changepoints ($M = 0$) and the first 1000 steps were removed as burn-in. Trace-plots of the parameters showed that the burn-in was sufficient and test runs of 50,000 steps on a subset of the data sets gave similar parameter estimates, indicating that 20,000 steps is sufficient.

In order to assess the performance of the RJMCMC algorithm the modal values, a 95% credible interval and mean absolute percentage error against the true value (MAPE, see (21)) are computed for each of the parameters π, ρ, λ and τ. The MAPE of an estimate E from true value T is given by:

$$\text{MAPE}(E, T) = \sum_{i=1}^{n} \frac{|E_i - T_i|}{|T_i|} \tag{21}$$

Additionally, to assess the estimation of community assignments $\boldsymbol{c}(t)$, the v-measure (Rosenberg and Hirschberg 2007) was computed. V-measure is a score between 0 and 1 given to a clustering of a data set where true class labels are available Rosenberg and Hirschberg (2007). It is an information theoretic measure based on the harmonic mean of two different scores: homogeneity and completeness. A clustering is considered *homogeneous* if it assigns *only* those data points that are members of a single class to a single cluster, whereas a clustering is considered *complete* if it assigns *all* of those data points that are members of a single class to a single cluster. The v-measure lies in the interval [0,1] with a v-measure of 1 denoting perfect reconstruction of the classes. Alternative metrics such as the Adjusted Rand Index (ARI) can also be used for assessing community assignment.

The v-measure V_{iht} was computed for each data set $i = 1, \ldots, 192$ at each time point $t = 1, \ldots, T$ for each

step $h = 1001, \ldots, 20,000$ of the sampler. The mean v-measure $v_i = \sum_h \sum_t V_{iht}/(HT)$ was computed for each data set by averaging over time and sampler step. Across all sampler runs, v_i has mean 0.9131 and median 0.9294 with inter-quartile range [0.8856,0.9548]. Similar results were found using the ARI which had mean 0.9079, median 0.9404 and inter-quartile range [0.8644,0.9945]. The lowest v-measure was 0.6476, obtained for a data set with $\pi_0 = \pi_k = 0.1$ and $\rho_0 = 0.2$ and $\rho_k = 1.2$. This is a difficult data set for the sampler since the probability of seeing a given edge at any time is 0.1 and all the information on the community structure is encoded in the parameter ρ.

Although λ was estimated well in every simulation (the true value was in the HPD interval), the number of changepoints was sometimes underestimated. This generally occurred because changes close to the start or end of the observation period or that occur close to another change are difficult to detect, a well-known feature of changepoint problems. In such cases the sampler is performing model selection by selecting a more parsimonious model than the one simulated from. For example, in the simulation with combined v-measure of 0.6476, the change in community memberships of nodes 26 and 63 are missed at times 1.26 and 2.03, respectively, and instead the sampler assigns the community they move to as their initial community. Such an early change is thus difficult to detect but may not be important since the important structure (*i.e.* the community membership after time 2) is still captured. A similar boundary effect is present for changes late in the observation period.

Finally, we investigated in more detail how the algorithm scales with the amount of data ($N = 50, 100, 150$; $T = 20, 40, 60$) and number of blocks ($K = 2, 4, 6$). The RJMCMC sampler run-time per iteration scales linearly with the number of snapshots and quadratically in the number of nodes which is to be expected as doubling the number of nodes quadruples the number of potential edges to evaluate. The number of blocks in the model appears to have a negligible effect on the run-time of the algorithm. For a fixed number of iterations the effective sample sizes of the MCMC output decreases slightly as N and T increase. Therefore, the main additional computational cost from analysing larger data sets is the larger likelihood calculations required.

6 Application: Communities of mice

In this section we apply the RJMCMC sampling scheme to a data set of mice contacts presented in Lopes et al (2016b). We aim to show how the algorithm can identify changes in community structure of this dynamic network. In this study, 90% of a population of 257 mice were observed for a period of 54 days (Lopes et al 2016a). Each nest box was fitted with a sensor which recorded when two mice were cohabiting. The data were presented as aggregates of time spent in close proximity, mainly collected every other day but with some observations collected every third day. We use the data by setting the edge $Y_{ij}(t_s)$ to 1 if mice i and j had any contact on observation day t_s. Since the mice sleep in nests and are social animals, it is hypothesised that the contact network will show community structure. In Lopes et al (2016b) the authors stage an intervention in some of the subjects by treating them with either lipopolyaccharide (LPS) or a placebo saline injection. It is hypothesised that treatment with LPS makes subjects more introverted and thus less likely to contact other subjects. The authors found that the treatment, when compared to placebo injections, reduced the degree to which mice interacted with others. We ask if the mice change their community structure, hypothesising that the treated mice may change community membership.

A preliminary analysis (Lopes et al 2016b) shows that the network is split into some disconnected components. We take a subset of 107 mice to form a sub-network. This sub-network contains some almost disconnected components with some connections between components. This sub-network contained 12 mice who received the active treatment and 17 mice treated with a placebo. The remaining mice received no treatment.

Initial clustering of subjects was performed using the distance d_{11} in Equation (20). This measure is used, since there is prior knowledge available that the communities are assortative. The value for γ_{11} found was 0.999 so this indicates that only the density of edges within groups was needed to determine the initial community structure. To determine the number of communities, K, we consider the between-sum-of-squares (BSS) and total-sum-of-squares (TSS) ratio $R_K = BSS_K/TSS_K$. The BSS_k is the sum of the squared distances in the k-means clustering for all nodes in different communities, whereas TSS_K is the sum of squared distances between all node pairs. A higher value of R_K means the communities are better separated and as K increases, this measure R_K tends to one. Using an elbow plot in Figure 2 for R_K, we chose an analysis based on five communities, since five is the point where increasing the number of communities, K, does not substantially increase the separation between communities as measured by R_K. However, we also include a sixth community with parameters clamped at zero. This allows for the analysis to model mice which leave the nest for a period of time.

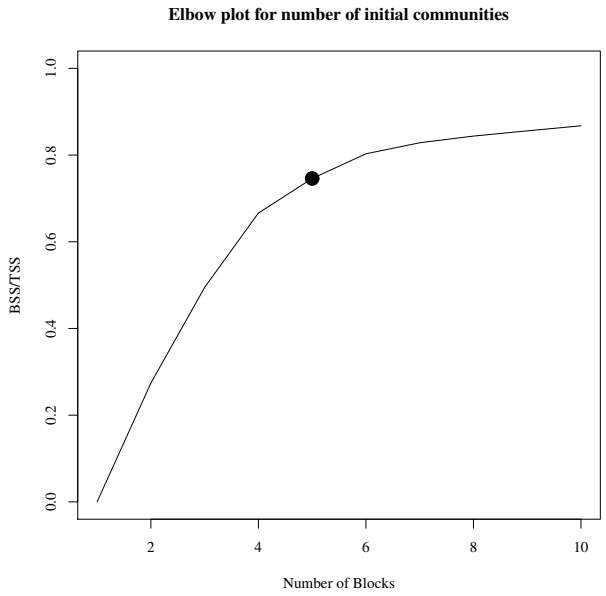

Fig. 2 Elbow plot for determining the number of communities with which to initialise the sampler.

We ran the RJMCMC sampler for 50,000 iterations discarding the first 10,000 as burn-in. This allows the number of changes to become stable, since the sampler starts with zero changepoints. The estimates for the community parameters are given in Table 3 with around 50 changes in community membership. Traceplots are available in the supplementary material. Notice that π_0 is low, showing that the communities are mainly disjointed. Contacts in communities 1, 2 and 5 are more likely than contacts in communities 3 and 4 with similar behaviour within these two groups. Note also that ρ is in the range (0.4, 0.6) for all communities giving a similar degree of autoregressive behaviour in the contact process for all mice. The higher value of ρ_0 corresponds to a more rapid turnover of contacts between mice in different communities, as one would expect.

Figure 3 shows the *a posteriori* most probable community membership through time for each mouse. The communities are coded by hatching, with the shading type z used at point (x, y) representing the highest posterior probability at time x of mouse y belonging to community z. The mice detected to have changed community were mainly mice which were absent from the nests over a short period. Such mice were detected to join the community labelled 6 in Figure 3. However, a few mice are more active. For example, the mouse with ID 97 leaves the nest from group 5 for some time then returns to group 4 and then leaves the nest again. For each of the 107 mice, we present plots of the posterior probabilities of a mouse belonging to each of the 6 communities over time in the supplementary material.

For comparison, the dynamic SBM (dynSBM) of Matias and Miele (2016) was fit to the same data. In this model, the nodes act independently and move between blocks via a discrete-time Markov chain. This gives similar dynamics for the nodes as for the ARSBM. The key difference is in the modelling of the edges. Under dynSBM, given the block memberships of the nodes, the edges are treated as independent Bernoulli random variables. Applying the dynSBM to the mice data set yields similar memberships to those found using ARSBM, as seen by comparing Figures 3 and 4. The mean parameter estimates for the dynSBM are given in Table 4 with some differences observed in the estimates of β_k and π_k, the probability of an edge existing between two nodes in community k in dynSBM and ARSBM, respectively. This is particularly the case for communities 1 and 2, reflecting the significant changes in community membership seen in the dynSBM between these two communities. The dynSBM method estimates 283 changes, more than five times the mean number of changes estimated using the ARSBM, with the latter maintaining a more consistent and coherent community structure.

Finally, we see no evidence that treating mice with LPS affects community structure of the network, (except by leaving the network). Even though mice are found to interact less by Lopes et al (2016b), those interactions are likely to be with the same group of mice.

Table 3 Parameter estimates for the mice community data set.

variable	5%	mean	95%	s.d.
M	49	52.57	56	2.0843
λ	0.0010	0.0019	0.0030	0.0006
π_0	0.0003	0.0004	0.0005	0.0001
π_1	0.6799	0.6994	0.7182	0.0118
π_2	0.6111	0.6660	0.7189	0.0328
π_3	0.4152	0.4349	0.4547	0.0121
π_4	0.4331	0.4473	0.4609	0.0084
π_5	0.6616	0.6821	0.7007	0.0119
ρ_0	1.1489	1.3164	1.5059	0.1092
ρ_1	0.4740	0.5138	0.5556	0.0248
ρ_2	0.3420	0.4104	0.4893	0.0453
ρ_3	0.5256	0.5650	0.6054	0.0244
ρ_4	0.3915	0.4077	0.4246	0.0102
ρ_5	0.6265	0.6851	0.7429	0.0357

7 Concluding remarks

In this paper we have introduced an autoregressive, continuous-time version of the stochastic block model

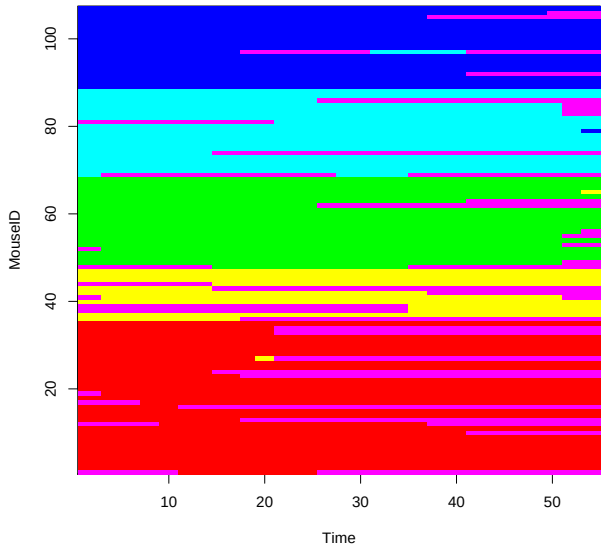

Fig. 3 ARSBM: Maximum a posteriori community membership of each mouse through time. Community labels are: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple. These labels match the parameter labels in Table 3.

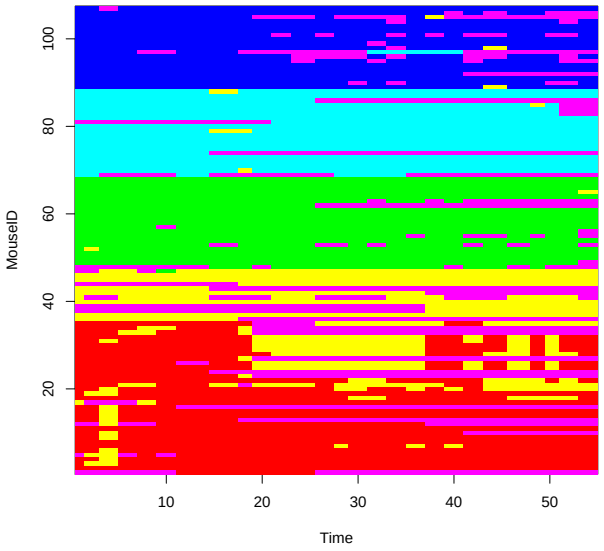

Fig. 4 dynSBM: Trace for community membership of each node. Community labels are: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple. These labels match the parameter labels in Table 4.

Table 4 Parameter estimates for the mice community data set from the dynSBM (β) and ARSBM (π, ρ).

k	β_k	π_k	ρ_k
0	0.0948	0.0004	1.3164
1	0.3645	0.6994	0.5138
2	0.7301	0.6660	0.4104
3	0.6415	0.4349	0.5650
4	0.5610	0.4473	0.4077
5	0.7602	0.6821	0.6851

and an effective RJMCMC algorithm to sample jointly from the posterior distribution of the parameters and the number and location of individuals' changes in community membership. The Markovian nature of the ARSBM makes it flexible and allows the model and RJMCMC algorithm to be trivially applied to irregularly observed data or data with gaps in the collection process, both of which are challenging problems for discrete-time models. The effectiveness of the RJMCMC algorithm is demonstrated through the simulation study with excellent detection of the changepoints in community membership. There are a number of exciting avenues for future research opened up by autoregressive stochastic block models. Firstly, whilst the initialisation procedure for community allocation worked well in the examples in this paper, alternative clustering algorithms could be considered, especially by estimating the community structure throughout the observation interval. This would enable the insertion of changepoints into the model at the start of the RJMCMC algorithm to reduce the potentially lengthy burn-in period. Secondly, it would be useful to allow the number of communities to be an unknown parameter which possibly varies over time. This would avoid the use of *ad hoc* methods such as an elbow plot to choose the number of communities and, more interestingly, allow the number of communities to vary through time, with the possibility of large global changes when communities split or merge. Further possible extensions include covariate information on edges or nodes and weighted edges. Both of these present challenges in efficient evaluation of the likelihood as in this paper we have been able to exploit the binary state of edges classified solely by the community membership of the nodes.

References

Albert R, Barabási AL (2002) Statistical mechanics of complex networks. Rev Mod Phys 74:47–97

Allman E, Matias C, Rhodes J (2011) Parameters identifiability in a class of random graph mixture models. Journal of Statistical Planning and Inference 141:1719–1736

Altieri L, Scott EM, Cocchi D, Illian JB (2015) A changepoint analysis of spatio-temporal point processes. Spatial Statistics 14:197–207

Chatterjee S, Diaconis P (2013) Estimating and understanding exponential random graph models. Ann Statist 41(5):2428–2461

Corneli M, Latouche P, Rossi F (2016) Block modelling in dynamic networks with non-homogeneous poisson processes and exact ICL. Social Network Analysis and Mining 6(1)

Davis RA, Lee TCM, Rodriguez-Yam GA (2006) Structural Break Estimation for Nonstationary Time Series Models. Journal of the American Statistical Association 101(473):223–239

DuBois C, Butts C, Smyth P (2013) Stochastic blockmodeling of relational event dynamics. In: Carvalho CM, Ravikumar P (eds) Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, PMLR, Scottsdale, Arizona, USA, Proceedings of Machine Learning Research, vol 31, pp 238–246

Fearnhead P, Liu Z (2007) On-line inference for multiple changepoint problems. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 69(4):589–605

Frank O, Harary F (1982) Cluster inference by using transitivity indices in empirical graphs. Journal of the American Statistical Association 77(380):835–840

Frank O, Strauss D (1986) Markov graphs. Journal of the american Statistical association 81(395):832–842

Fryzlewicz P (2014) Wild binary segmentation for multiple change-point detection. The Annals of Statistics 42(6):2243–2281

Fu W, Song L, Xing EP (2009) Dynamic mixed membership blockmodel for evolving networks. In: Proceedings of the 26th annual international conference on machine learning, ACM, pp 329–336

Green PJ (1995) Reversible jump markov chain monte carlo computation and bayesian model determination. Biometrika 82(4):711–732

Guigourès R, Boullé M, Rossi F (2015) Discovering patterns in time-varying graphs: a triclustering approach. Advances in Data Analysis and Classification

Haynes K, Eckley IA, Fearnhead P (2017) Computationally efficient changepoint detection for a range of penalties. Journal of Computational and Graphical Statistics 26(1):134–143

Holland PW, Laskey KB, Leinhardt S (1983) Stochastic blockmodels: First steps. Social networks 5(2):109–137

Killick R, Fearnhead P, Eckley IA (2012) Optimal Detection of Changepoints With a Linear Computational Cost. J Amer Statist Assoc 107(500):1590–1598

Kolaczyk ED (2009) Statistical Analysis of Network Data: Methods and Models. Springer New York

Lloyd S (1982) Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2):129–137

Lopes P, Block P, König B (2016a) Data from: Infection-induced behavioural changes reduce connectivity and the potential for disease spread in wild mice contact networks.

Lopes PC, Block P, König B (2016b) Infection-induced behavioural changes reduce connectivity and the potential for disease spread in wild mice contact networks. Scientific Reports 6:31,790

Matias C, Miele V (2016) Statistical clustering of temporal networks through a dynamic stochastic block model. Journal of the Royal Statistical Society: Series B (Statistical Methodology) p (to appear).

Matias C, Rebafka T, Villers F (2017) A semiparametric extension of the stochastic block model for longitudinal networks, working paper or preprintdata

Matteson DS, James NA (2014) A Nonparametric Approach for Multiple Change Point Analysis of Multivariate Data. Journal of the American Statistical Association 109(505):334–345

Norris JR (1997) Markov Chains. Cambridge University Press, cambridge Books Online

Picard F, Robin S, Lebarbier E, Daudin JJ (2007) A segmentation/clustering model for the analysis of array cgh data. Biometrics 63(3):758–766

Roberts GO, Gelman A, Gilks WR (1997) Weak convergence and optimal scaling of random walk metropolis algorithms. Ann Appl Probab 7(1):110–120

Rosenberg A, Hirschberg J (2007) V-measure: A conditional entropy-based external cluster evaluation measure. In: EMNLP-CoNLL, vol 7, pp 410–420

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

Watts DJ, Strogatz SH (1998) Collective dynamics of small-worldnetworks. nature 393(6684):440–442

Xiang F, Neal P (2014) Efficient mcmc for temporal epidemics via parameter reduction. Computational Statistics & Data Analysis 80:240–250

Xie Y, Siegmund D (2013) Sequential multi-sensor changepoint detetion. Annals of Statistics 41:670–692

Xin L, Zhu M, Chipman H (2017) A continuous-time stochastic block model for basketball networks. The Annals of Applied Statistics 11(2):553–597

Xu KS, Hero AO (2014) Dynamic stochastic blockmodels for time-evolving social networks. IEEE Journal of Selected Topics in Signal Processing 8(4):552–562

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