The autoregressive stochastic block model with changes in structure

Matthew Ludkin, M.Sci.(Hons.), M.Res

Submitted for the degree of Doctor of Philosophy at Lancaster University.

November 2017

STOR-i
excellence with impact

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

Matthew Robert Ludkin

A version of Chapter 3 has been published as:
Ludkin, M., Eckley, I., and Neal, P. (2017). Dynamic stochastic block models: parameter estimation and detection of changes in community structure. *Statistics and Computing*. https://doi.org/10.1007/s11222-017-9788-9

Acknowledgements

I would like to thank my supervisors: Peter Neal and Idris Eckley. Their help and guidance has been crucial to the completion of this thesis. I would like to thank the STOR-i Centre for Doctoral Training and the EPSRC for providing a vibrant community in which to conduct research. Thanks also to DSTL and Ralph Mansson who provided funding for the PhD program and helpful discussions about the research within this thesis. Finally, I would like to thank Helen Eden for the care and support provided during my PhD, without which I know it would never have been possible.

Abstract

Network science has been a growing subject for the last three decades, with statistical analysis of networks seing an explosion since the advent of online social networks. An important model within network analysis is the stochastic block model, which aims to partition the set of nodes of a network into groups which behave in a similar way. This thesis proposes Bayesian inference methods for problems related to the stochastic block model for network data. The presented research is formed of three parts. Firstly, two Markov chain Monte Carlo samplers are proposed to sample from the posterior distribution of the number of blocks, block memberships and edge-state parameters in the stochastic block model. These allow for non-binary and non-conjugate edge models, something not considered in the literature.

Secondly, a dynamic extension to the stochastic block model is presented which includes autoregressive terms. This novel approach to dynamic network models allows the present state of an edge to influence future states, and is therefore named the autoregresssive stochastic block model. Furthermore, an algorithm to perform inference on changes in block membership is given. This problem has gained some attention in the literature, but not with autoregressive features to the edge-state distribution as presented in this thesis.

Thirdly, an online procedure to detect changes in block membership in the autoregresssive stochastic block model is presented. This allows networks to be monitored through time, drastically reducing the data storage requirements. On top of this, the network parameters can be estimated together with the block

memberships.

Finally, conclusions are drawn from the above contributions in the context of the network analysis literature and future directions for research are identified.

Contents

Declaration ii

Acknowledgements iii

Abstract iv

List of Figures viii

List of Tables xi

1. Introduction 1

1.1. Modelling networks 2

1.1.1. Notation and definitions 3

1.1.2. Mathematical models for networks 4

1.1.3. Statistical network models 7

1.1.4. The stochastic block model 12

1.2. Dynamic stochastic block models 16

1.3. Contributions 19

2. Arbitrary edge-states and unknown number of blocks in the stochastic block model 24

2.1. Introduction 24

2.2. Model 28

2.2.1. Prior for block structure 31

2.3. Dirichlet process sampler 34

2.4. Split-merge sampler 36
2.5. Simulated data 48
2.5.1. Example networks 48
2.5.2. Assessing convergence 56
2.6. Real data 57
2.6.1. Macaque 57
2.6.2. Enron 61
2.6.3. Stack Overflow 63
2.7. Concluding remarks 64

3. Autoregressive stochastic block model with changes in block membership 67

3.1. Introduction 67
3.2. The autoregressive stochastic block model 71
3.2.1. Model 71
3.2.2. Posterior distribution 73
3.2.3. Identifiability 76
3.3. Reversible jump MCMC 78
3.3.1. Sampling scheme 78
3.3.2. Updating change points and augmented edge states 80
3.4. Initialisation of sampler state 84
3.5. Simulation study 87
3.6. Application: Communities of mice 90
3.7. Concluding remarks 94

4. Online monitoring of block membership in the autoregressive stochastic block model 98

4.1. Introduction 99
4.2. Model 101

4.3. SMC . . . 106
4.3.1. Sufficient statistics . . . 107
4.3.2. SMC algorithm . . . 111
4.4. Simulated data . . . 116
4.4.1. Comparison to offline methods . . . 120
4.5. Application to dynamic contact network . . . 123
4.5.1. Mice network . . . 123
4.6. Closing remarks . . . 126

5. Perspectives and future directions **129**

A. Appendix for arbitrary edge-states and unknown number of blocks in the stochastic block model **133**
A.1. Enron . . . 133

B. Appendix for autoregressive stochastic block model with changes in block membership **136**

C. Appendix for monitoring block membership in the autoregressive stochastic block model **144**
C.1. Posterior plots . . . 144
C.2. Trace of block membership per mouse . . . 148

Bibliography **148**

List of Figures

2.1. Comparison of block structures. Top left CRP(1), top right CRP(5). Bottom left: DMA(1,5), bottom right: DMA(5,5). 34
2.2. Posterior summaries for block membership in Bernoulli example network. 52
2.3. Posterior summaries for block membership in Poisson example network. 53
2.4. Posterior summaries for block membership in normal example network. 54
2.5. Posterior summaries for block membership in negative binomial example network. 55
2.6. Trace plots for number of blocks in example networks. Two chains are simulated in each case: The "lower chain" with all nodes initially in one block (orange line) and the "upper chain" with all nodes initially assigned to different blocks (blue line). 58
2.7. Posterior summaries for block membership in Macaque brain network using Dirichlet process sampler. 59
2.8. Posterior summaries for block membership in Macaque brain network using Dirichlet process sampler. 60
2.9. Posterior summaries for block membership in Enron network with Poisson edge-state model. 62
2.10. Posterior summaries for block membership in Enron network with negative binomial edge-state model. 62

2.11. Posterior summaries for block membership in Stack Overflow network. 64

3.1. Possible changes. 81
3.2. Elbow plot for determining the number of communities with which to initialise the sampler. 92
3.3. ARSBM: Maximum a posteriori community membership of each mouse through time. Community labels: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple. 95
3.4. dynSBM: Maximum a posteriori community membership of each mouse through time. Community labels: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple. 96

4.1. v-measure for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states. 118
4.2. Mean absolute deviation for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states. 119
4.3. Bias for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states. 120
4.4. v-measure against true λ values with method 2. 121
4.5. Comparison of v-measure for simulated networks under SMC and RJMCMC algorithms. 124
4.6. Comparison of bias in parameters under SMC (left) and RJMCMC (right) for example networks. 125
4.7. Maximum *a posteriori* block memberships of mice. 127

A.1. Posterior summaries for block membership in Enron network with Poisson edge-state model and strong prior. 134
A.2. Posterior summaries for block membership in Enron network with negative binomial edge-state model and strong prior. 134

B.1. Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1 137
B.2. Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1 138
B.3. Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1 139
B.4. Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1 140
B.5. Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1 141
B.6. Trace plots for $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$ parameters for mice data set. 142
B.7. Trace plots for λ and number of change-points for mice data set. . . 143

C.1. Posterior mean for λ at each time point in the mice network. 144
C.2. Posterior mean for $\boldsymbol{\omega}$ at each time point in the mice network. 145
C.3. Posterior mean for $\boldsymbol{\phi}$ at each time point in the mice network. 146
C.4. Posterior mean for $\boldsymbol{\rho}$ at each time point in the mice network. 147
C.5. Traces of block membership in mouse network. 148
C.6. Traces of block membership in mouse network. 149
C.7. Traces of block membership in mouse network. 150
C.8. Traces of block membership in mouse network. 151
C.9. Traces of block membership in mouse network. 152

List of Tables

2.1. Possible matching functions to ensure parameters lie in the correct space. 41
2.2. Simulated data parameter values for each edge-state distribution considered. 49
2.3. Mean and 95% credible interval (CI) for parameters of example networks. 51
2.4. Rubin-Gelman Statistics (and upper bound of 95% confidence interval) for each model with 30 independent chains of 5000 iterations for RJMCMC and Dirichlet Process samplers. 56
2.5. Parameter estimates and 95% highest posterior density interval for Macaque network. 60
2.6. Parameter estimates and 95% highest posterior density interval for Enron data set under both models. 62
2.7. Parameter estimates and 95% highest posterior density values for the Stack Overflow network. 65
2.8. Model block structure for the Stack Overflow network. 65

3.1. Parameter settings for simulation study. 88
3.2. Number of nodes per community for $n_c = unequal$. 88
3.3. Parameter estimates for the mice community data set. 94
3.4. Parameter estimates for the mice community data set from the dynSBM (β) and ARSBM (π, ρ). 94

4.1. Summary statistics for the ARSBM 108

4.2. Parameter settings for simulated data sets. 117

4.3. Posterior means for SMC and RJMCMC algorithms in simulated networks. 125

4.4. Final time posterior mean and variance for mice network under SMC and the corresponding estimates for the RJMCMC algorithm. . . . 126

A.1. Mean posterior parameter values and 95% credible interval for Enron network under Poisson and negative binomial models with vague and strong priors. 135

1. Introduction

Networks are ubiquitous in modern life: physical networks including roads, rails and pipes carry goods and services. Virtual networks, including telecommunications and the internet, enable sharing of files and messages. Social networks, both online and offline, describe the connections between people and animals. The field of network science is very broad, covering problems such as route-finding on a road network, designing telecommunications networks robust to the failure of connections or finding the most influential member of a social network. The field can be broadly divided based on the characteristics of the network under study. For example, in the network flow problem, the network is considered fixed (the roads are already in place) and an optimal route is sought. In the robust telecommunications example, the network is being designed (where should the intersections be placed?). Finally, in social network analysis, often the network structure is unknown and needs to be inferred to answer questions such as influence. This leads to quite different sub-fields of study.

This thesis is concerned with problems such as the latter example: data are recorded on a network of individuals and the structure of this network is to be inferred. Such problems occur in the sub-field of complex systems, where understanding the interactions between individual parts of a system lead to system-level behaviour. Furthermore, the structure of these interactions, and how such structure came about, is key to understanding such complex systems. For this, network modelling arose as a subject in its own right, dividing into two main classes: statistical and mathematical. Mathematical models in this domain aim to explain

a mechanism under which the network was generated, e.g. small-world networks Watts and Strogatz (1998), whereas statistical models aim to fit a probabilistic model with all the power of assessing fit using statistical inference procedures. The line between mathematical and statistical model is blurred but the literature on both has remained mainly distinct over the past two decades.

Much of the early work on network modelling focused on *static* networks, where information about a network is collected at one point in time, or is summarised to yield a single piece of information between pairs of individuals. A popular and well studied statistical model is the stochastic block model (SBM). This divides the set of nodes into groups, such that nodes in the same group have a similar behaviour in terms of their edge-states. Example application areas for the SBM include determining the friendship groups in social network analysis (Wang and Wong, 1987; Snijders and Nowicki, 1997; Zhao et al., 2011; Newman and Reinert, 2016) and separating the regions of the brain into a number of functional groups (Négyessy et al., 2006a).

In recent years, models for *dynamic* networks have been developed. These extend static network analysis and models to consider the state of networks through time. An interesting problem is thus to determine if a change in structure has occurred in the network.

This thesis is concerned with changes in the structure of a network. The topic lies at the intersection of network modelling, dynamic network models and changepoint detection. This chapter reviews the fields of network modelling for both static and dynamic networks, and introduces the contributions of each chapter and their position within the statistical literature.

1.1. Modelling networks

The modelling of networks mainly falls into two types: mathematical or statistical (Kolaczyk, 2009). In this thesis "mathematical models" refers to a class of models concerning network growth. In such growth models, the observed network

is assumed to be generated by some simple probabilistic rule with a small set of parameters. The estimation of such parameters from an observed network can indeed be viewed as a statistical problem, however the term "statistical models" refers to models designed to explain observed data. As such, statistical models allow for standard statistical tools such as goodness of fit tests and the evaluation of explanatory power of variables on the formation of edges within a network. This section reviews the literature on both these areas, with a focus on statistical models. Before proceeding with the review, firstly the mathematical concept of a network is defined and notation is introduced.

1.1.1. Notation and definitions

Networks are often conceptualised as mathematical *graphs*. In such a framework, the interactions of a network are denoted by *edges*, while the individuals or entities performing the interactions are designated as *nodes*. A static network $\mathcal{N}$, consisting of a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$, is written $\mathcal{N} = (\mathcal{V}, \mathcal{E})$. Statistical analysis of a network $\mathcal{N}$ involves models for the generation of the edges in $\mathcal{E}$. For ease of reference, let ij denote the edge between nodes i and j. Furthermore, denote by E_{ij} the state of edge ij in network $\mathcal{N}$; this is considered random and is the quantity to be modelled. Networks with values for the edges can be viewed as weighted graphs, so in the above notation, the weight of edge ij is the value of E_{ij}.

In the case of dynamic networks, a time index is introduced to the notation as $\mathcal{N}(t) = (\mathcal{V}(t), \mathcal{E}(t))$, denoting that the set of nodes or edges can change over time. Research presented in this thesis mainly concerns dynamic networks on a fixed set of nodes. As such, networks of the form $\mathcal{N}(t) = (\mathcal{V}, \mathcal{E}(t))$ are of interest. For ease of reference, an edge is always considered to exist between two nodes, whereas the edge-state may not. For example, in a dynamic social network the edge-states may represent if two people are in conversation. Under such a setting, the edge-state for a given pair can take values of true or false (people can either

be in conversation or not). By allowing every edge to exists, the phrase "the edge between nodes i and j switches state from true to false" is comprehensive.

Some graph-theoretic terms carry over into network modelling: in the case of binary edge-states, the *degree* of a node is the number of edges in state one for which i is an end-node. Furthermore, the *size* of a binary network is the number of edges in state one. A *degree sequence* is the (ordered) list of degrees $\{d_i : i \in \mathcal{V}\}$.

The types of interaction can vary also. In some cases the interaction process may allow for self-interactions. These are called *self-loops*. Symmetric interactions, such as "is friends with", create symmetric networks. In symmetric networks the edge-states have the following relationship: $E_{ij} = E_{ji}$ for all $i \neq j$. If a network is symmetric with no self-loops, then it is modelled as a *simple graph*. On the other hand, interactions may have a concept of direction, such as "node i sends node j an email"; such networks are modelled as *directed graphs*.

For a given network $\mathcal{N}$, it is convenient to consider the *adjacency matrix*. This is the matrix of edge-states $\boldsymbol{E}$. If a network contains symmetric edge-states, then $\boldsymbol{E}$ is a symmetric matrix. Furthermore, if there are no self-loops in $\mathcal{N}$, then $\boldsymbol{E}$ has 0 values on the diagonal.

1.1.2. Mathematical models for networks

A mathematical model for a network is mainly concerned with the mechanisms that gave rise to the network via simple rules with few parameters. As such they can be referred to as *network growth* models. There is an extensive literature on these models, mainly in the domain of statistical physics. Generally, a mathematical model takes some parameters $\boldsymbol{\theta}$ which drive some growth mechanism.

The observed network $\mathcal{N}$ is used to estimate $\boldsymbol{\theta}$, then comparisons between theoretical network features under $\boldsymbol{\theta}$ and features of the observed graph are compared. Examples of network features of interest include average degree, the number of transitive triples (complete sub-graphs on three nodes) and average shortest path length. The topics' breadth is down to the wide choice in growth mechanisms.

Since this thesis is mainly concerned with statistical models, only a few major mechanisms are reviewed here.

The first growth model to appear in the literature is arguably the simple random graph (Gilbert, 1959; Erdos and Rényi, 1960). In such a model, the set of N nodes, $\mathcal{V}$, is fixed and binary edge-states are assigned to the edges. In one version of the model, given the nodes, the edge-states are drawn uniformly with some probability p. An alternative version (which is easier to generalise), chooses a network at random from the set of all networks containing a given number of edges. The generalised Erdős-Rényi model chooses a network uniformly at random among among a set of networks with a given set of properties. This generalises the original model since the number of edges is such a property. Another common property is the degree sequence, hence $\mathcal{N}$ is chosen from the set of networks with a set degree sequence $\{d_1, d_2, \ldots, d_N\}$. Interest may lie in some counts of arbitrary sub-graphs called "motifs" for example, the number of connected triples (complete sub-graphs on three nodes).

The following procedure demonstrates the ideas behind the mathematical modelling paradigm. To analyse a given network $\mathcal{N}$, firstly a property or properties $\mathcal{P}$ are chosen and calculated as $\hat{p} = p(\mathcal{N})$ for each $p \in \mathcal{P}$. Secondly, the theoretical properties are calculated for the set of graphs obeying $\mathcal{P}$. Finally, $\hat{p}$ is compared to its theoretical value. For most properties of interest, the theoretical values are non-analytic, and hence MCMC algorithms to draw from the set of networks obeying $\mathcal{P}$ have been developed. This is a whole literature in itself, but key algorithms are given in Kolaczyk (2009). For the case of general motifs, first an underlying network generation model must be assumed, then the theoretical distribution for motif counts calculated. This quantity is non-analytic and difficult to approximate, even for the simple Erdős-Rényi model (Picard et al., 2008). Furthermore, without a robust theoretical distribution for motif counts, the use of p-values for model fit is difficult to justify.

The small-world model was popularised by Watts and Strogatz (1998). Such

networks have the following properties: (i) most nodes have few neighbours, (ii) the neighbourhoods of two neighbours has a large intersection, (iii) the average of the shortest path between any two nodes grows logarithmically in the number of nodes. The work of Watts and Strogatz (1998) classified networks based on the *clustering coefficient* and *shortest path length*. The clustering coefficient measures the tendency for nodes to group together into clusters and attempts to measure (i) and (ii) in the above list. If nodes group together, then the intersection of the neighbourhoods of two nodes in the same group will contain many nodes. The clustering coefficient is defined as the ratio of the number of triplets of nodes with all possible edges to the number of triplets of nodes with two possible edges. This can be seen in a graph as the number of triangles divided by the number of connected triplets.

In an Erdős-Rényi network model, the clustering coefficient is small, together with a short average shortest path length (ASPL). For example, with N nodes and probability of an edge appearing p, such that $Np > 1$, the clustering coefficient is p and ASPL grows as $\mathcal{O}(\log N)$ in the limit $N \to \infty$ (Bollobás, 1998). However, in small-world networks, the clustering coefficient is large with a small ASPL.

The mechanism to generate a small-world network as given by Watts and Strogatz (1998) is as shown in Algorithm 1.1. In the first stage a regular lattice network is created. This ensures the network starts with a high clustering coefficient. Secondly, by "rewiring" edges at random (avoiding already existing edges and self loops), the ASPL is greatly reduced. As $\beta \to 1$, a Watts-Strogatz network approaches an Erdős-Rényi network. For the case of $\beta = 0$, the Watts-Strogatz algorithm yields a lattice: each node has exactly K neighbours and consecutive nodes share $K - 2$ neighbours. In this case, the clustering coefficient is $\frac{3(K-2)}{4(K-1)}$ which approaches $\frac{3}{4}$ for large K whilst the ASPL is exactly $\frac{N}{2K}$. For the case where $\beta \in (0, 1)$, the ASPL decreases quickly with β, whilst the clustering coefficient remains close to the value at $\beta = 0$ and can be shown to be asymptotically equal to $\frac{3(K-2)}{4(K-1)}(1-\beta)^3$. Hence, for a large range of β values, the Watts-Strogatz algorithm

produces small-world networks.

Algorithm 1.1 The generation of a small world network via the Watts-Strogatz algorithm.

Require: Number of nodes N, mean degree K, $\beta \in [0, 1]$.
  Start with node set $\mathcal{V} = \{1, \ldots, N\}$ and an empty edge set $\mathcal{E}$.
  Label N nodes as $1, \ldots, N$.
  for $i = 1, , \ldots, , N$ **do**
    For nodes j such that $i - j = 1, , \ldots, , K \mod N$, add ij to $\mathcal{E}$.
  end for
  for $i = 1, , \ldots, , N$ **do**
    for $j = i, , \ldots, , N$ **do**
      With probability β choose a node $k \in \{j : j \neq i, ij \notin \mathcal{E}\} \neq i$.
      Delete ij from $\mathcal{E}$.
      Add ik to $\mathcal{E}$.
    end for
  end for
  return Network $\mathcal{N} = (\mathcal{V}, \mathcal{E})$.

Another popular mathematical model for networks is the preferential attachment model. The mechanism generating such networks is often dubbed "the rich get richer". A well-known algorithm for generating a preferential attachment network is described by Albert and Barabási (2002). To generate a network, start with a single edge between two nodes n_1 and n_2. At each step, a new node n_i is added, and an edge between n_i and n_j $(j < i)$ is created with probability proportional to the degree of n_j. In such a way, the more edges incident to a node, the more likely it will receive an edge from a new node. This is made concrete in Algorithm 1.2. As such, networks generated from Algorithm 1.2 are likely to contain some nodes with very high degrees, or *hubs*. Another property of the Albert and Barabási (2002) model is the degree distribution: it is scale-free. Specifically $\mathbb{P}[d(n_i) = k] \propto k^{-3}$. The ASPL can be shown to be asymptotically $\frac{\log N}{\log \log N}$.

1.1.3. Statistical network models

In Section 1.1.2, some famous mathematical models for generating networks were reviewed. These models have been used to compare against real-world networks. For example, co-authorship networks as analysed by Newman (2001, 2004b) with

Algorithm 1.2 The generation of a scale-free network via the Barabási–Albert algorithm.

Require: Number of nodes N, initial number of connected nodes N_0.
  Start with node set $\mathcal{V} = \{1, \ldots, N_0\}$ and empty edge set $\mathcal{E} = \emptyset$.
  for $i = 2, \ldots, N_0$ **do**
    for $j < i$ **do**
      Add ij to $\mathcal{E}$.
    end for
    Set the degree $d_i = N_0 - 1$.
  end for
  for $i = N_0 + 1, \ldots, N$ **do**
    Add node i to $\mathcal{V}$.
    for $j < i$ **do**
      Add ij to $\mathcal{E}$ with probability $p_{ij} = \frac{d_j}{\sum_{k<i} d_k}$.
    end for
    Update the degrees of nodes $1, \ldots, i$.
  end for
  return Network $\mathcal{N} = (\mathcal{V}, \mathcal{E})$ and the degree sequence $\{d_i : i \in \mathcal{V}\}$.

network properties such as average degree, ASPL and clustering coefficient reported and discussed.

In this section, some statistical models are reviewed. The change in focus between statistical and mathematical models is on estimation and representation: models must be able to be fit to the data and allow exploration of the effects that explain the data. None of the mathematical models are suited to this.

Three main categories of statistical model have been developed over the last few decades: exponential random graph models (ERGMs), latent space models, and stochastic block models (SBMs). These each mirror classical statistical methods: the ERGM can be viewed as a generalised linear model, latent space models use both observed and latent variables to model the probability of edge-states in the manner of classic latent space models, and SBMs are, at their heart, a mixture model.

Firstly, a review of ERGMs is presented. The Exponential Random Graph Model (ERGM) uses global properties of a graph to model the edges in a network and, hopefully, capture phenomena at the node level (Anderson et al., 1999). ERGMs have their origins in the social sciences with the so called p_1 and p^* mod-

els (Wasserman and Pattison, 1996). These are special cases of the more general ERGM which itself is analogous to classical generalised linear models (GLMs).

$$\mathbb{P}[\boldsymbol{E} = \boldsymbol{e}] = \frac{1}{J}\exp\left(\sum_{h\in H}\theta_h g_h(\boldsymbol{e})\right), \tag{1.1}$$

where g_h are functions counting the number of times configuration h appears in $\boldsymbol{e}$. Equation (1.1) shows the general form of the ERGM, an exponential family form, for the joint distribution of the adjacency matrix $\boldsymbol{E}$. The ERGM is based on *configurations* or sub-graphs within the network: for example, the appearance of edges or triangles or k-stars (a set of k edges sharing the same end-node). The hope is that simple structures can explain the observed adjacency matrix $\boldsymbol{e}$. In Equation (1.1), the configurations of interest are denoted by the set H. For each configuration h, there is an indicator function g_h, which counts the number of times configuration h appears in the adjacency matrix $\boldsymbol{e}$. If θ_h is non-zero, then the E_{ij} are dependent for all i, j in configuration h. This is the main appeal of the ERGM: a certain dependency structure on the appearance of edges in the network can be imposed through a small number of configurations. A further draw of the ERGM framework is the ease with which additional information $\boldsymbol{X}$ on the network can be included, simply specify the conditional distribution of $\boldsymbol{E}$ on $\boldsymbol{X}$ in exponential form with the addition of statistics g depending on $\boldsymbol{e}$ and $\boldsymbol{x}$. For example, if covariate information on the nodes is available as a vector $\boldsymbol{x}$ with x_i the data about node i, then a simple model would be to include additive effects: $g(\boldsymbol{e}, \boldsymbol{x}) = \sum_{i=j} e_{ij}(x_i + x_j)$. In this case, the log-odds of the edge ij appearing in the network increases with the covariate values for i and j. Second order terms can be included by comparing the values of x_i and x_j. In the simplest case, $\boldsymbol{x}$ is discrete and a matching criteria can be used: $g(\boldsymbol{e}, \boldsymbol{x}) = \sum_{i=j} e_{ij}\mathbb{I}[x_i + x_j]$.

The simplest such ERGM assumes that each edge appears independently, with some probability θ_{ij} for nodes i and j. As such the functions g_h for configuration including more than two nodes has $\theta_h = 0$ and the model reduces to: $\mathbb{P}[\boldsymbol{E} = \boldsymbol{e}] \propto \exp\left(\sum_{i=j}\theta_{ij}e_{ij}\right)$. Obviously this model is overly flexible: there is a parameter for

each data point. Setting each $\theta_{ij} = \theta$ makes the further assumption that edges are independent and identically distributed and, in this case, the Erdős-Rényi model is recovered with $\mathbb{P}[\boldsymbol{E} = \boldsymbol{e}] \propto \exp(\theta M(\boldsymbol{e}))$.

Although the ERGM has been demonstrated as a very flexible model, being able to incorporate covariate and dependency structures into the model, it does suffer from some problems. The configurations to include must be chosen carefully, since they easily conflict, leading to correlated estimates. For example, the number of triangles will be correlated to the number of edges since the number of edges is at least three times the number of triangles.

The work of Frank and Strauss (1986) posit ERGMs including only terms for the number of triangles and some k-stars (notice this includes the simple model of counts for edges and triangles since an edge is a 1-star). This model is simpler than the full model in Equation (1.1) which should lead to more interpretable results. In practise however, this model yields poor fit to real data due to model degeneracy (Handcock et al., 2003). To overcome such an issue, more terms could be included, but this leads to a large model. Various attempts to rectify this include making a parametric assumption on the k-star terms (Snijders et al., 2006; Robins et al., 2007).

Although the ERGM allows many potential specifications, fitting the model to data is a challenge. The maximum likelihood estimates for $\boldsymbol{\theta}$ in Equation (1.1) are well defined for appropriate models, but the estimation is non-trivial due to the normalisation term J. This term is only available for trivial ERGMs since it involves summing over all networks on N nodes for each possible $\boldsymbol{\theta}$: clearly a large problem. A pseudo-likelihood procedure can be used, however, Snijders et al. (2006) show in simulation studies that maximising the likelihood can lead to unbounded estimates or false maxima. A Markov chain Monte Carlo maximum likelihood procedure is available (Hunter and Handcock, 2006) which is the preferred method to fit ERGMs. However, unlike GLMs, the theory of ERGMs is not well established, with no asymptotic results available for a wide range of

ERGMs. Recent research (Bhamidi et al., 2011; Chatterjee and Diaconis, 2013) provides a framework for the asymptotic analysis of edge-and-triangle ERGMs, and give proofs of model degeneracy. Therefore, despite the potential of such models, ERGMs lack the theoretical underpinnings to be used with confidence.

The latent space models developed by Hoff et al. (2002); Handcock et al. (2007); Hoff (2008a,b); Krivitsky et al. (2009) treat the nodes as exchangeable in the absence of covariate information. This is motivated by the Aldous-Hoover theorem, since if the elements E_{ij} are exchangeable, then they can be expressed in the form:

$$E_{ij} = h(\mu, u_i, u_j, \xi_{ij})$$

with where h is a probability, μ a constant, u_i, u_j i.i.d. latent variables (with h symmetric in u_i, u_j) and ξ_{ij} i.i.d. pair-specific effects. This still leaves much flexibility via the specification of h. A popular approach is to let ξ_{ij} be standard Gaussian variates, μ to be augmented with covariate information and include latent terms $\boldsymbol{u}$ through some function α. This leads to a probit-like model in Equation (1.2). The choice of function α and the latent space U to which u_i, u_j belong determines the latent effects of the model, analogous to the choice of configurations in the ERGM. Letting $u_i, u_j \in \{1, , \ldots, , \kappa\}$ together with $\alpha(u_i, u_j) = m_{u_i u_j}$ for real valued m_{kl} is similar to the stochastic block model: if i and j are in groups k and l in the latent space U, then the probit model increases by m_{kl}. An interesting choice for α, U comes from the social science principle of homophily, where similar individuals are likely to associate with each other. In this case, α is chosen as a distance function on the latent space U, such that nodes close together in latent space are more likely to share an edge.

$$\mathbb{P}[E_{ij} = 1 | X_{ij} = x_{ij}] = \Phi\big(\mu + x'_{ij}\boldsymbol{\beta} + \alpha(u_i, u_j)\big) \tag{1.2}$$

Model fitting in the class of latent network models is achieved via a Bayesian approach, due to the hierarchical nature of the model. Latent network models

offer a variety of possible models but it can be difficult to interpret the latent space and the nodes' positions therein. Those models based solely on a distance-based α can also suffer in estimation, since the likelihood is invariant to isometric transformations of the latent space coordinates $\boldsymbol{u}$.

1.1.4. The stochastic block model

Lastly, the stochastic block model (SBM) is reviewed. This is the basis of the research presented in this thesis and, as such, a more rigorous introduction is given for the SBM. The SBM was first posed by Holland et al. (1983); Fienberg et al. (1985); Wasserman and Anderson (1987) as a random graph model. In these, the nodes are split into clusters or *blocks*, and, given the *block memberships*, the edge-states are generated from a mixture model. The mixture weights for a given edge-state are determined by the block membership of the end nodes. The main assumption under the SBM is the node-set $\mathcal{V}$ can be partitioned into κ blocks such that any node belongs to only one block. Furthermore, edge-states are assumed independent of other edge-states, given the block memberships. Letting $\boldsymbol{E}$ be an adjacency matrix for a network and $\boldsymbol{z}$ denote the block memberships, such that $z_{ik} = 1$ if node i is assigned to block k and $z_{ik} = 0$ otherwise, then the SBM may be written in hierarchical form in Equation (2.1).

$$\begin{aligned} \boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{\omega} &\sim \text{Multinomial}(\boldsymbol{z}|\boldsymbol{\omega}) \\ \Theta_{kl} = \theta_{kl} &\sim \text{Beta}(\theta_{kl}|\boldsymbol{\alpha}_{kl}) \\ E_{ij} = e_{ij}|\boldsymbol{\theta}, \boldsymbol{z} &\sim \text{Bernoulli}\big(e_{ij}|\theta_{z_i z_j}\big) \end{aligned} \tag{1.3}$$

The general form of the SBM has been well studied (Airoldi et al., 2005; Hastings, 2006; Picard et al., 2007; Daudin et al., 2008; Ambroise and Matias, 2012) including maximum likelihood and variational inference procedures. Other authors from statistical physics refer to the SBM as a community detection problem (Newman, 2004a; Girvan and Newman, 2002; Clauset, 2005). A full review of community methods including the SBM can be found in Fortunato (2010). Notice that, in

this case, "community" normally refers to a block structure where a node is more likely to share an edge with a node in the same block than one in another block ($\theta_{kk} \geq \theta_{kl}$). The most popular fitting procedure in the statistical physics literature is the maximisation of *modularity*. Modularity according to Girvan and Newman (2002) measures the difference between connectivity between blocks and within blocks. Specifically, let $M_{kl}(\boldsymbol{z}, \boldsymbol{E}) = \sum_{i=j} E_{ij}\mathbb{I}[z_{ik} = 1, z_{jl} = 1]$ be the counts of edges between block k and l under the block membership vector $\boldsymbol{z}$. Then the modularity of $\boldsymbol{z}$ is:

$$Q(\boldsymbol{z}, \boldsymbol{E}) = \sum_{k=1}^{\kappa} \frac{M_{kk}}{M} - \left(\frac{\sum_{l=1}^{\kappa} M_{kl}}{M} \right)^2 .$$

Maximising modularity with respect to $\boldsymbol{z}$ yields block structures where the density of edges within blocks is higher than between blocks. Notice that modularity maximising methods do not perform inference on the parameters $\boldsymbol{\theta}$, they only recover the block structure $\boldsymbol{z}$. It has been argued that modularity maximisation is biased compared to maximum likelihood estimation (Bickel and Chen, 2009) although recent work has shown its equivalence to a restricted form of the SBM (Newman, 2016).

A specific form of the SBM, dubbed the *affiliation model*, has been studied in depth by Snijders and Nowicki (1997); Nowicki and Snijders (2001); Copic et al. (2009). In this case, the parameters $\boldsymbol{\theta}$ are reduced to either between-block and within-block parameters as $\theta_{kk} = \theta_w$ and $\theta_{kl} = \theta_b$ for $k, l \in \{1, \ldots, \kappa\}$. The above references discuss technical issues such as fitting the affiliation model and the model degeneracy when all blocks are of the same size.

The classic SBM makes the assumption that the number of communities, κ, is known *a priori*. This can be considered too restrictive. Various methods have been considered to estimate the number of blocks within a network, and is an active research area. These include likelihood based methods using the Bayesian information criteria and its derivatives (Daudin et al., 2008; Latouche et al., 2012; Wang et al., 2017; Saldaña et al., 2017), information based methods using minimum

description lengths (Peixoto, 2013), sequential testing by embedding successive block models with increasing κ (Lei, 2016) and cross validation (Chen and Lei, 2016). These methods all fit an SBM model with a given κ, then do a post-hoc analysis to find an appropriate "final" κ.

An alternative approach is to let κ be random, and infer κ together with the block assignments $\boldsymbol{z}$ and parameters $\boldsymbol{\theta}$. Extending the SBM by allowing κ to vary leads to the *Infinite Relational Model* (IRM) (Mørup and Schmidt, 2013). The IRM extends the model hierarchy of the SBM by placing a prior on κ. Specifically in the case of the IRM, a joint prior is placed on $\kappa, \boldsymbol{z}$. This takes the form of the Chinese Restaurant Process (CRP) (Gershman and Blei, 2012). On top of this, the parameters $\boldsymbol{\theta}$ can be integrated out of the SBM model, leading to efficient collapsed Gibbs sampling algorithms (Mørup et al., 2011; Mørup and Schmidt, 2012, 2013; McDaid et al., 2013)

The IRM extends the SBM by placing a CRP prior on the number of blocks and block memberships. The CRP is a form of Dirichlet process used in clustering (Antoniak, 1974; Anderson, 1991; Escobar and West, 1995; Rasmussen, 2000; Neal, 2000). To cluster a set of points $e_1, \ldots, e_N$, the CRP assigns each point sequentially. Firstly, following the derivation of Gershman and Blei (2012), e_1 forms a cluster labelled 1. After $i-1$ points are assigned, suppose there are κ_i clusters. Then, e_i is assigned to cluster k proportional to the number of points in cluster k (for $k = 1, \ldots, \kappa_i$) or e_i starts a new cluster with probability γ (a model parameter). This process is exchangeable, so the partition defined by the CRP does not depend on the order in which the vertices are assigned to clusters (Gershman and Blei, 2012). The CRP has a similar property to the preferential attachment model in Section 1.1.2: a large cluster is more likely to gain new points. As such, the CRP tends to create partitions where one part is much larger than the others.

For use in network modelling, the CRP is used as a prior distribution on the block membership of nodes. In the above formulation, nodes are treated as points while blocks are treated as parts in the partition. In this way, the number of blocks

and the block memberships are treated as unknown *a priori*. With this prior, the SBM is extended to the IRM in Equation (1.4)

The IRM can be written as:

$$\begin{aligned} \kappa, \boldsymbol{z} &\sim \mathrm{CRP}(\gamma) \\ \theta_{kl} &\sim \mathrm{Beta}(\boldsymbol{\alpha}) \\ E_{ij}|\boldsymbol{z} &\sim \mathrm{Bernoulli}\big(\theta_{z_i z_j}\big) \end{aligned} \tag{1.4}$$

The posterior distribution of $\boldsymbol{z}$ can be found using a collapsed Gibbs sampler (Mørup and Schmidt, 2013). Note that under such an inference procedure, the $\boldsymbol{\theta}$ parameters are treated as nuisance parameters and are integrated out of the model.

So far, only binary edge-states have been considered. This is reflected in the literature, with arbitrary edge-states considered only recently (Jiang et al., 2009; Mariadassou et al., 2010; Ambroise and Matias, 2012). The generalisation to arbitrary edge-states is simple, given the block memberships $\boldsymbol{z}$, the edge-state $\boldsymbol{E}$ are assumed independent, and the distribution, G, of E_{ij} depends on $\boldsymbol{z}_i, \boldsymbol{z}_j$ and some parameters $\boldsymbol{\theta}$. These are the core assumptions of the SBM shown in Equation 1.5.

$$\begin{aligned} (E_{ij} \perp\!\!\!\perp E_{i'j'})\,|\boldsymbol{z} \text{ for all } i, i', j, j' \in \mathcal{V} \\ g(E_{ij}|\boldsymbol{z}) = g(E_{ij}|Z_i, Z_j, \boldsymbol{\theta}) \end{aligned} \tag{1.5}$$

For example, when considering edge-states representing count data, a Poisson distribution may be used for G (Mariadassou et al., 2010) or a Normal distribution for real-valued edge-states (Wyse and Friel, 2012).

Other authors consider non-binary data within the IRM model (i.e. an unknown number of blocks). Wyse and Friel (2012) and McDaid et al. (2013) extend the IRM to consider both Bernoulli and Poisson distributed edge-states with a collapsed Gibbs sampler for inference on $\kappa, \boldsymbol{z}$ similar to work by Mørup and Schmidt (2013).

1.2. Dynamic stochastic block models

There is a vast literature on dynamic network models, including dynamic extensions of the models introduced in Section 1.1.3. Early work in the field started with continuous-time Markov processes with edge-independence (Wasserman, 1980a,b) and stochastic actor oriented models (Snijders and van Duijn, 1997; Snijders et al., 2010). For a full overview see the review paper by Holme (2015). This thesis focuses on dynamic extensions to the stochastic block model, and thus a more comprehensive review of this field is given in this section.

Multiple temporal extensions of the stochastic block model exist. These can be classified in one important way: how the data are collected. In point-process-like models, edge-state data is assumed to be collected with a time-stamp. As such, data sets come in the form of a list of edges with a time point. For example, the list (i_s, j_s, t_s) for $s = 1, \ldots, S$ represents the S observed edges, with the s^{th} observed edge at time t_s between nodes i_s and j_s. In *snapshot models*, the edge-state of *all* edges is collected at predetermined observation times. As such, a series of network "snapshots" are taken at times say $t_0, t_1, \ldots, t_S$, hence, the data comes in the form of a list of adjacency matrices $\boldsymbol{E}_0, \boldsymbol{E}_1, \ldots, \boldsymbol{E}_S$.

Point process models for dynamic networks consider data in the form of instantaneous interactions between the nodes, such as sending an email. The stochastic block model with point process data aims to divide the nodes of the network based on their behaviour over time. Letting $E_{ij}(t)$ be the time-dependent edge-state for the pair of nodes i and j, a point process model, $F(\boldsymbol{\lambda})$, is placed on $E_{ij}(t)$ with some parameters $\boldsymbol{\lambda}$. The standard form of the stochastic block model can be applied by letting the parameter $\boldsymbol{\lambda}$ depend on the block membership of nodes i and j. For example, consider a network of employees at a company and data on when a pair of employees exchange emails. A block structure could form between the departments of the company, such that the rate of email exchange between employees in the same department is higher than across departments.

Research on point process stochastic block models includes the use of Cox's

multivariate hazard model with time-dependent covariates (Butts, 2008; Vu et al., 2011; Perry and Wolfe, 2013). These are Poisson process models with random intensity functions. Another special case of the Poisson process is the Hawkes process. A Hawkes process is a self-exciting point process, whereby the intensity function increases at times close to a point. In this way, the existence of a point makes future points more likely. Specifically, an impulse function is added to the intensity function in a region close to points. In the network case, a multivariate Hawkes process can be used to model the edge-states. Specifically each edge $E_{ij}(t)$ has a rate function $\lambda_{ij}(t)$. In the Hawkes process case, an impulse can be added to all intensity functions $\boldsymbol{\lambda}(t)$ when a point in a single edge is witnessed. Hawkes process models with the SBM have been developed by multiple authors including Blundell et al. (2012), where there is a Hawkes process per pair of blocks (rather than pair of nodes). In this way an edge-state $E_{ij}(t)$ follows a Hawkes process with parameters determined by the end-nodes. Furthermore, Blundell et al. (2012) use the IRM rather than the SBM, allowing inference to be performed on the number of blocks as well as the underlying processes. Cho et al. (2013) extend the Hawkes process idea further by allowing both a temporal relationship in events (via a Hawkes process) and a spatial component (via a spatial Gaussian mixture model). Linderman and Adams (2014) also propose an SBM with a Hawkes process in non-observed networks, with the view to inferring both the network and its structure through time. The above models all treat the edge-states as point processes, however, research has also been done where the nodes are modelled rather than the edges (Fox et al., 2016).

Point process models are appropriate for instantaneous interactions such as sending emails or instant messages, but not so appropriate for interactions with a duration such as phone calls or proximity. An alternative dynamic extension of the stochastic block model assumes that data is available in the form of snapshots. These snapshots record the state of all edges in the network at predefined times $t_0, \ldots, t_S$. A stochastic block model in this framework assigns nodes to blocks in

such a way that the dynamics of the edges between nodes i and j depends on the block membership of the pair i, j. Models in the literature mainly concern binary edge-states but allow the block membership of nodes to change through time. Therefore, given the block memberships z_t at time t, the edge-states are drawn from a Bernoulli SBM. On top of this, the block membership of nodes follows some Markov process. In the works of Fu et al. (2009); Xing et al. (2010); Yang et al. (2011); Xu and Hero (2014), the latent block memberships evolve as a discrete time Markov chain. Specifically, Fu et al. (2009); Xing et al. (2010) propose a mixed-membership SBM where the block memberships are represented as a vector $\boldsymbol{\pi}_{it}$, with π_{kit} denoting the amount to which node i belongs to block k at time t. The parameter $\boldsymbol{\pi}_{it}$ is drawn at each time point from a logistic-normal distribution with mean μ_t and variance Σ_t. Dynamics are added to the block memberships by assuming that these means μ_t evolve through time via an autoregressive Normal process. On top of this, the parameters governing edge-formation are assumed to follow a similar construction. This choice of a logistic normal distribution makes parameter estimation difficult since no conjugate prior is available and the authors appeal to variational inference. An alternative approach is presented in Yang et al. (2011), where the block memberships are allowed to change over time, but are not mixed-membership. Therefore, a node can belong to only one block per time, but can move between these blocks. In this way, the evolution of block membership $\boldsymbol{z}_t$ is modelled by a discrete time Markov chain with transition matrix A. As such, the probability that a node in block k remains in block k in consecutive time points is A_{kk}, and the probability of moving from block k to l is A_{kl}. Given the block memberships at time t, the edge-states are assumed to be generated from a static SBM. A variational expectation maximisation algorithm and a simulated annealing approach are presented to infer the block memberships at each time, the edge-state probabilities and the transition matrix A. Xu and Hero (2014) allow nodes to transition between blocks and the parameters of the SBM to change through time. Specifically, given the block memberships $\boldsymbol{z}_t$ at time

t, the edge-states are drawn from a static SBM with parameter $\boldsymbol{\theta}_t$, where θ_{kl} is the probability of an edge appearing between nodes in block k and l. The inference procedure transforms $\boldsymbol{\theta}_t$ to the real line (via a logit transform) to $\boldsymbol{\psi}_t$. The $\boldsymbol{\psi}_t$ are then treated as Gaussian variates with mean $A\boldsymbol{\psi}_{t-1}$ and variance Σ_t for parameters A and Σ. Given this structure, an extended Kalman filter is applied to get approximate parameter estimates for $\boldsymbol{\theta}_t$ at each time point, then a label switching procedure is used to infer the block memberships $\boldsymbol{z}_t$. There is no explicit model for node transitions.

More recent work by Matias and Miele (2017) extend the SBM by assuming the block memberships evolve as a discrete-time Markov chain. Given the block memberships at time t, the edge-states are drawn from a static stochastic block model. The authors also consider non-binary edge-states; in this case, the edge-state of ij at time t (E_{ijt}) is drawn from a distribution G with a parameter $\boldsymbol{\theta}_{kl}$ if $z_{it} = k$ and $z_{jt} = l$. A variational inference procedure is presented to find the maximum likelihood estimates for the parameters $\boldsymbol{\theta}$ and block membership $\boldsymbol{z}_t$ through time. Furthermore, the authors discuss model choice to determine the number of blocks and allow for nodes to exit and enter the network during the observation period. The authors discuss the problem of parameter identifiability in this model, where a permutation of the block labels leads to the same inference. As such it can be impossible to follow the path of a node through time, all that can be recovered is the groups at each time point. This problem affects all the models discussed above.

1.3. Contributions

This thesis consists of three chapters, each representing a contribution to the literature. Chapter 2 considers the SBM with an unknown number of blocks and arbitrary edge-states, Chapter 3 introduces a dynamic extension of SBM allowing for autoregressive behaviour in the edge-states. This is named the autoregressive stochastic block model (ARSBM). Chapter 4 considers online monitoring of block

structure in the ARSBM together with inferring the fixed model parameters. This section gives an overview of the contributions made in each of the above chapters.

In Chapter 2, two Markov chain Monte Carlo (MCMC) algorithms are presented to draw samples from the posterior distribution of the number of blocks, block memberships and edge-state parameters in a stochastic block model. Historically, research for inference in the stochastic block model has mostly treated the number of blocks as fixed. Of the research where the number of blocks κ is treated as unknown, those which incorporate inference on κ into the main inference (such as the infinite relational model), only conjugate models for the edge-states have been considered. Furthermore, only the Chinese restaurant process prior is used. This is shown to be a rather inflexible choice in Section 2.2.1.

Existing literature on determining the number of blocks in an SBM either uses a model selection criteria or includes the number of blocks as a parameter in the inference process. Both algorithms presented in Chapter 2 work in the latter framework. Research in this domain has only considered edge-state models for which conjugate priors are available. In such an approach, the edge-state parameters can be integrated out of the model, reducing the variance in the inference process. This leads to Gibbs sampling algorithms which concentrate on only the number of blocks and block memberships of nodes. While these points are appealing, there are some negatives: (i) only conjugate models can be considered for edge-states, (ii) Gibbs samplers can get stuck in local modes. McDaid et al. (2013) improve the Gibbs sampler by including some split-merge moves but still only applies to conjugate models. In Chapter 2 a Dirichlet process MCMC and a reversible jump MCMC algorithm are introduced which allow for any edge-state distributions (provided samples can be taken and a density computed). This greatly increases the flexibility of the SBM with arbitrary edge-states. Specifically, in Section 2.6.2 a negative binomial model with both parameters unknown is applied to the Enron email data set to analyse the structure within an email network. This is not possible using conjugate models. A comparison was made to an SBM with a Poisson

edge-state model, and the negative binomial finds additional structure, since the model is more flexible. Furthermore, a discussion in Section 2.2.1 considers the prior distribution of blocks and block memberships. Traditionally, this is taken as a Chinese Restaurant Process (CRP) which jointly models the number of blocks and block memberships. However, the CRP gives significant weight on block structures dominated by one larger block and multiple small blocks. In Section 2.2.1, it is argued that the Dirichlet Multinomial Allocation (DMA) prior, first used in cluster analysis (Green and Richardson, 2001), is a more flexible model for the number of blocks and block memberships. Specifically, the DMA is a hierarchical model allowing the number of blocks to be specified via an arbitrary distribution with support on a subset of the positive integers. Following this, a prior is placed on the block memberships given the number of blocks. As such, priors can be constructed where the number of blocks can be modelled separately to the block structures. These allow distributions where blocks are expected to be of equal size without influencing the number of blocks; an impossibility under the CRP.

In Chapter 3, a dynamic network model is introduced as an extension of the SBM. Research in this area has considered various dynamic extensions to the SBM including point process models for instantaneous events and snapshot models for events with a duration. These two techniques are complimentary: in general point process models fix block memberships for all time, but allow the rate of edge appearance to depend on time. In snapshot models, block memberships are allowed to change through time and generally the edge-state process has fixed parameters. In the snapshot model literature, model hierarchies consider dynamic block memberships, allowing nodes to change block membership over time. However, at each time point the edge-states are considered independent of the past, given the block memberships. This forces all dynamics into the unobserved block memberships. Chapter 3 considers a snapshot model where autoregressive terms are included for the edge-states, as such, this is named the autoregressive stochastic block model (ARSBM). The ARSBM makes some important contributions: (i) by allowing

autoregressive edges, a more realistic temporal model is provided, (ii) by setting the model in a continuous time framework dealing with missing data or irregular sampling is simple. The closest available method is the work of Matias and Miele (2017), where dynSBM is proposed. This model sets the block-membership process as a discrete-time Markov chain. Given the block memberships, at each time-point the edge-states are drawn independently from a static SBM model. Compared to Chapter 3, dynSBM cannot handle irregular sampling times without adaptation. Furthermore, no autoregressive behaviour is possible for the edge-states. In the ARSBM, the location of change points is inferred via a reversible jump Markov chain Monte Carlo sampling algorithm. This yields a distribution over change location, providing a quantification in uncertainty of the change points. Current methods, including dynSBM only provide point estimates for block membership and model parameters, or use a variational approximation technique. Such variational techniques are known to be over confident in their maximum likelihood estimates, and thus the uncertainty in parameter estimates provided by current methods is often underestimated (Blei et al., 2017), hence the ARSBM inference is more honest in its parameter uncertainty when compared to currently available methods for dynamic stochastic block models. The ARSBM allows more realistic treatment of dynamic network data by explicitly allowing temporal dependence on previous edge-states. On top of this, the problem of identifying evolutions in block membership is considered; specifically, the problem of detecting if nodes have changed group is tackled. This problem has been considered by other authors both in a Bayesian (Fu et al., 2009; Yang et al., 2011; Xu and Hero, 2014) and a frequentist context (Matias and Miele, 2017), yet these models do not allow for autoregressive terms.

Finally in Chapter 4 a sequential Monte Carlo (SMC) procedure for the online monitoring of block membership is considered for the ARSBM. Not only are the block memberships tracked through time, the fixed parameters are also estimated. To achieve this, a data augmentation scheme is utilised to allow the posterior

distribution of parameters given the edge-states and block structures to be separable. This scheme allows for an MCMC within particle filter algorithm (Storvik, 2002; Fearnhead, 2002). Specifically, a Gibbs sampler can be implemented to infer the parameters through time in a rigorous manner within the SMC algorithm. Therefore, the storage requirements of the algorithm are much smaller than the RJMCMC algorithm of Chapter 3. This problem has been attempted (Fu et al., 2009; Xing et al., 2010; Yang et al., 2011; Xu and Hero, 2014), but without the inclusion of autoregressive terms. Hence, as in Chapter 4, applying the ARSBM in a dynamic setting allows more flexible modelling of temporal data by explicitly allowing future edge-states to depend on the past.

A number of extensions to the research presented in this thesis are discussed in Chapter 5. This includes missing data in the edges-states, inclusion of covariate information on both the edges and nodes, and scalable inference procedures.

2. Arbitrary edge-states and unknown number of blocks in the stochastic block model

2.1. Introduction

Statistical analysis of networks has seen much growth in recent years with the increasing availability of network data. The term "network" is used in a broad range of research fields. In this paper, a network consists of a set of nodes, which can form pairwise interactions. Each possible interaction is referred to as an edge, with the value of that interaction being denoted as an edge-state. In similar work these are referred to as possible edges and edge-weights. By referring to each possible pair of nodes as an edge, the terminology is more succinct. The aim of statistical network modelling is to describe the edge-states with a probabilistic model, potentially performing inference for model parameters. Such models include the exponential random graph, the class of latent space models and the stochastic block model (SBM). Under the SBM, the set of nodes is partitioned into *blocks* such that the edge-state between two nodes depends on their block memberships. For example, an *assortative* block structure in a network with binary edge-states is formed when nodes in the same block are more likely to have an edge-state of one than between nodes in different blocks.

In this paper we aim to identify network structure via an extension of the

stochastic block model. The SBM has been studied at least since the 1980s (Holland et al., 1983; Frank and Harary, 1982), with attention turning to non-binary edge-states in the 2000s (Jiang et al., 2009; Mariadassou et al., 2010; Ambroise and Matias, 2012). There is a rich literature on the SBM including both Bayesian and Frequentist treatments. Extensions to the SBM include restricting the SBM to only within-block and between-block edge-state distributions in the affiliation network (Snijders and Nowicki, 1997; Nowicki and Snijders, 2001; Copic et al., 2009), multiple-block memberships in the mixed-membership SBM (Airoldi et al., 2008), degree-corrected SBM (Karrer and Newman, 2011), and the infinite relational model (IRM) (Kemp et al., 2006) where the number of blocks is treated as unknown. For a thorough review of the SBM and inference methods see Matias and Robin (2014).

The Bayesian inference procedure we present is applicable to networks with arbitrary edge-states, extending the work of Mørup and Schmidt (2012, 2013) and McDaid et al. (2013). The inference algorithms presented in this paper allow much more flexible modelling than previous work on networks with arbitrary edge-states. Previous authors have only considered conjugate models for edge-states, whereas we allow much more flexibility in the choice of prior by only assuming that (1) samples can be taken and (2) point-wise evaluation is computationally feasible. This greatly broadens the applicability of the stochastic block model to general network data with arbitrary edge-states.

Various methods have been considered to estimate the number of blocks within a network. Such methods fall into two main approaches: (1) a post-hoc analysis of multiple models (using model selection methodology) or (2) incorporating the number of blocks as a random variable. The model selection techniques in likelihood based methods using the Bayesian information criteria and its derivatives (Daudin et al., 2008; Latouche et al., 2012; Wang et al., 2017; Saldaña et al., 2017), information based methods using minimum description lengths (Peixoto, 2013), sequential testing by embedding successive block models with an increasing

number of blocks (Lei, 2016) and cross validation (Chen and Lei, 2016). These methods all fit multiple SBM models with differing numbers of blocks, then do a post-hoc analysis to choice a final number. Alternatively, treating the number of blocks as a random variable allows inference on the joint distribution of number of blocks, block membership of nodes and model parameters. Extending the SBM by allowing the number of blocks to vary leads to the *Infinite Relational Model* (IRM) (Mørup and Schmidt, 2013). The IRM extends the model hierarchy of the SBM by placing a Chinese Restaurant Process (CRP) prior (Gershman and Blei, 2012) on the number of blocks. On top of this, the parameters $\boldsymbol{\theta}$ can be integrated out of the SBM model, leading to efficient collapsed Gibbs sampling algorithms (Mørup et al., 2011; Mørup and Schmidt, 2012, 2013; McDaid et al., 2013)

In this paper, the number of blocks is estimated under the later paradigm. Two algorithms are presented to sample from the posterior distribution of the block parameters, block memberships and number of blocks in a stochastic block model. The first algorithm uses the Dirichlet process (DP) sampler (Neal, 2000) to create a Metropolis-within-Gibbs sampler. Given the block memberships, the parameters $\boldsymbol{\theta}$ can be updated using a Metropolis algorithm. The block memberships and number of blocks can be updated in turn given the parameter values. This is similar in spirit to the collapsed Gibbs sampler of McDaid et al. (2013) – for a given node i, the posterior probability of belonging to block k is computed with all other parameters fixed. Under the collapsed regime, considering assigning i to a new block k^* is simple, since the parameters $\boldsymbol{\theta}$ have been integrated from the model. In the case of non-conjugate mixture models, this parameter is required to evaluate the likelihood of node i belonging to block k^*. However, notice that in the SBM, if a node is the only member of a block, then such a block contains no edges – any parameter value θ_{k^*} does not affect the likelihood. By drawing θ_{k^*} from its prior, the proposal and prior densities will cancel, meaning θ_{k^*} has no effect on the acceptance probability either. This allows the sampler to create new blocks undeterred by the value of θ_{k^*} (contrast this to mixture models, where a

poor value of θ_{k^*} can decrease the likelihood substantially).

The Gibbs like nature of the DP sampler is not without its pitfalls: the sampler can get "stuck" in local maxima of the posterior. Specifically, in the case of block models, the sampler can be stuck such that two "true" blocks (k, l say with n_k and n_l nodes respectively) are assigned to block m. To reach a better local maxima, the sampler must separate the nodes currently in m to two new blocks matching the true labels k and l. Under the Gibbs sampler, to reach such a state requires moving at least $\min(n_k, n_l)$ nodes. Each of these moves is unlikely, meaning the series of such moves is very unlikely. However, if the sampler proposed all n_k nodes belonging to the "true" block k to be moved at once, the sampler could "jump" to a place of higher posterior density. Such moves are considered in the second sampler introduced.

The second sampler is inspired by Green and Richardson (2001) – a reversible jump MCMC (Green, 1995) scheme using split and merge proposals to explore the posterior by either combining two blocks, or splitting a block into two. Such split-merge moves avoid the local maxima of the DP sampler, but at the expense of proposing parameter values. Nobile and Fearnside (2007); McDaid et al. (2013) make use of a split-merge proposal, although due to the conjugate models considered, they do not require parameter values $\boldsymbol{\theta}'$. The difficulty in designing a good split-merge algorithm rests on ensuring that parameter values are "matched" when changing dimension.

The remainder of the paper is organised as follows: in Section 2.2 the specifics of the SBM are presented together with a discussion on the choice of prior distributions for the number of blocks and block memberships. Sections 2.3 and 2.4 introduce the Dirichlet process and split-merge samplers respectively. In Section 2.5 both samplers are applied to simulated data, whilst in Section 2.6 the split-merge sampler is used to analyse some real networks. Finally, closing remarks and extensions to the samplers are discussed in Section 2.7.

2.2. Model

The canonical SBM (Holland et al., 1983; Fienberg et al., 1985; Wasserman and Anderson, 1987) considers a network with a fixed number of nodes and blocks denoted as N and κ respectively. The nodes are then partitioned into blocks, with each node belonging to only one block. Let $\boldsymbol{z}$ be the block indicator matrix with $z_{ik} = 1$ if node i belongs to block k and 0 otherwise. As such $\boldsymbol{z}_i$ is a one-of-κ indicator vector. It is assumed that $\boldsymbol{z}_i$ is drawn from a Multinomial distribution with parameter $\boldsymbol{\omega}$. The parameter $\boldsymbol{\omega}$ governs the block memberships, with ω_k being the probability that a node joins block k.

For each pair of blocks, there is an associated parameter ϑ_{kl} which governs the probability of an edge-state between nodes in blocks k and l. These parameters can be arranged into a matrix, with diagonal elements ϑ_{kk} governing edge-states between nodes both in block k. In the case of undirected edges, the parameter matrix is symmetric with $\vartheta_{kl} = \vartheta_{lk}$. Finally, the edge-states are modelled as independent Bernoulli random variables with probability of success based on the block membership of the nodes. Specifically, the edge-state for edge ij, denoted E_{ij}, is drawn from a Bernoulli distribution with probability ϑ_{kl} where $z_{ik} = z_{jl} = 1$. Notice this can be written as the quadratic form $\boldsymbol{z}_i'\boldsymbol{\vartheta}\boldsymbol{z}_j$. This model is summarised in Equation (2.1), first the nodes are assigned to blocks; then given these block memberships, the edge-states are drawn with parameters depending on the block membership of the end-nodes.

$$\begin{aligned} \boldsymbol{z}|\boldsymbol{\omega} &\sim \text{Multinomial}(\boldsymbol{\omega}, \kappa) \\ E_{ij}|\boldsymbol{\vartheta}, \boldsymbol{z} &\sim \text{Bernoulli}(\boldsymbol{z}_i'\boldsymbol{\vartheta}\boldsymbol{z}_j) \end{aligned} \tag{2.1}$$

Given a data set consisting of the edge-states in a network, recovery of the block memberships is of interest. However, when performing inference, the values for parameters $\boldsymbol{\vartheta}$ are likely to be unknown. In the case of binary edge-states, a natural extension to Equation (2.1) is to allow ϑ_{kl} to be a Beta random variable. Therefore, in applications of the SBM, when only the edge-states are observed,

inference must be performed on both $\boldsymbol{z}$ and $\boldsymbol{\vartheta}$.

The structure of the SBM can be applied to non-binary edge-states, such as count data or a continuous value. In this way, the parameters $\boldsymbol{\vartheta}$ apply to some edge-state distribution other than the Bernoulli. As such, a prior distribution other than the Beta is required for the parameters in order to perform inference. A further extension allows $\boldsymbol{\omega}$ to be treated as unknown by assigning it a Dirichlet prior. Letting G and G_0 be the distribution on the edges-states and parameters respectively yields the model in Equation (2.2).

$$\begin{aligned} \boldsymbol{\omega} &\sim \text{Dirichlet}(\boldsymbol{\gamma}) \\ \boldsymbol{z}|\boldsymbol{\omega} &\sim \text{Multinomial}(\boldsymbol{\omega}, \kappa) \\ \vartheta_{kl} &\sim G_0(\alpha_{kl}) \\ E_{ij}|\boldsymbol{\vartheta}, \boldsymbol{z} &\sim G\big(\vartheta_{z_i z_j}\big) \end{aligned} \tag{2.2}$$

The IRM treats the number of blocks as unknown. Multiple authors have considered Bayesian inference for the IRM with conjugate models for the edge-states In this case, the parameters governing the edge-states can be integrated out of the likelihood, resulting in a collapsed Gibbs sampler (McDaid et al., 2013). Such an approach is reliant on the conjugate assumptions; hence, only a limited number of edge-state models can be fitted using such algorithms.

In the canonical SBM, there are κ blocks. When considering the block membership of the end-nodes there are $\binom{\kappa+1}{2}$ possibilities: the end nodes can be in the same block (κ possibilities) or in different blocks ($\binom{\kappa}{2}$ possibilities). Allowing every pair of blocks k, l to be governed by different parameters $\boldsymbol{\vartheta}_{kl}$ say, then the number of parameters grows quadratically in κ. Therefore, to control this growth, this paper considers a restricted form of the SBM. Other authors have considered reduced forms of the SBM. The affiliation model (Snijders and Nowicki, 1997; Nowicki and Snijders, 2001; Copic et al., 2009), only has two parameters: one for edges between nodes in the same block and another for edges between nodes in different blocks. Therefore the number of parameters is always two, no matter

how many blocks the are. This paper considers a parameterisation between these two extremes: letting $\boldsymbol{\theta}_k$ be the parameters governing edges between nodes in the same block k (of which there are κ), and a global parameter $\boldsymbol{\theta}_0$ for edge-states between nodes in different blocks. In this way the number of parameters is $\kappa + 1$, and grows linearly in the the number of blocks. This model is appropriate for networks where between block connections are relatively homogeneous; for example in ecological contact networks, where herds of animals remain close together for most of the time, with some between herd interactions. For comparison to the generic SBM from Equation (2.2), let $\boldsymbol{\theta}$ be the matrix with parameters with $\theta_{kl} = \theta_0$ and $\theta_{kk} = \theta_k$, then the quadratic form $\boldsymbol{z}_i'\boldsymbol{\theta}\boldsymbol{z}_j$ picks the parameter governing the edge-state E_{ij}. This may be extended to edge-state distribution G with multiple parameters. Note that $\boldsymbol{\theta}_k$ is a vector of parameters for the edge-state distribution for edges in block k (or between blocks if $k = 0$). For example, if G represents the Gaussian distribution, then $\boldsymbol{\theta}_k = (\mu_k, \sigma_k^2)$ represents the mean and variance of the edge-states in block k. In this case, an additional subscript is required on $\boldsymbol{\theta}_k$ such that θ_{kp} is the p^{th} parameter for block k. In the Gaussian example θ_{k1} is the mean value of edges in block k.

Since the number of blocks κ is considered unknown in this paper, a prior must be placed on both the number of blocks and block memberships. Choices for this prior are discussed in Section 2.2.1. Prior parameters $\boldsymbol{\alpha}$ are assigned to the block parameters $\boldsymbol{\theta}$. Meanwhile, let F be a joint distribution for $(\kappa, \boldsymbol{z})$ with parameters $\boldsymbol{\gamma}$, hence the restricted form of the SBM considered in this paper is shown in Equation (2.3).

$$\begin{aligned} \kappa, \boldsymbol{z} &\sim F(\boldsymbol{\gamma}, \delta) \\ \boldsymbol{\theta}_k &\sim G_0(\boldsymbol{\alpha}) \\ E_{ij}|\boldsymbol{\theta}, \boldsymbol{z} &\sim G(\boldsymbol{z}_i'\boldsymbol{\theta}\boldsymbol{z}_j) \end{aligned} \tag{2.3}$$

2.2.1. Prior for block structure

Under a canonical SBM, the number of blocks κ are assumed known. In this case, the standard prior to place on the block allocations $\boldsymbol{z}$ is a Multinomial($\boldsymbol{\omega}$). In the case where κ is unknown, this can be extended by setting a prior on both κ and $\boldsymbol{z}$. A hierarchical prior distribution can be achieved by first setting an arbitrary prior distribution F_0 for κ, and then setting a prior for $\boldsymbol{z}$ given κ. One approach is to let $\boldsymbol{z} \sim$ Multinomial($\boldsymbol{\omega}$), where $\boldsymbol{\omega} \sim$ Dirichlet(γ, κ), the symmetric Dirichlet distribution on the $\kappa - 1$ simplex. Such a hierarchical prior is referred to as a Dirichlet Multinomial Allocation (DMA) prior (Green and Richardson, 2001). Since a symmetric Dirichlet distribution with parameter γ is used, the parameter $\boldsymbol{\omega}$ can be marginalised out to obtain a prior density for block memberships depending only on κ and γ as shown in Equation (2.4).

$$\begin{aligned} f(\boldsymbol{z}|\gamma,\kappa) &= \int_{\boldsymbol{\omega}} f(\boldsymbol{z}|\boldsymbol{\omega})\, \pi_0(\boldsymbol{\omega}|\gamma)\, d\boldsymbol{\omega} \\ &= \int_{\boldsymbol{\omega}} \prod_{k=1}^{\kappa} \prod_{i=1}^{N} \omega_k^{z_{ik}+\gamma+1} \frac{\Gamma(\kappa\gamma)}{\Gamma(\gamma)^{\kappa}} d\boldsymbol{\omega} \\ &= \frac{\Gamma(\kappa\gamma)}{\Gamma(\gamma)^{\kappa}} \frac{\prod_{k=1}^{\kappa} \Gamma(N_k+\gamma)}{\Gamma(\sum_{k=1}^{\kappa} N_k + \kappa\gamma)} \end{aligned} \tag{2.4}$$

where N_k is the number of nodes in block k.

By taking the limits $\kappa \to \infty, \gamma \to 0$ while $\kappa\gamma \to \hat{\gamma}$, the Dirichlet process named the Chinese Restaurant Process (CRP) is obtained (Aldous, 1985). This has the interpretation of a block model with infinitely many blocks, while the sum of the Dirichlet parameters remains equal to $\hat{\gamma}$. For further discussion on the connections between DMA and DPs see Green and Richardson (2001). The CRP is a popular choice of prior for the SBM (Mørup et al., 2011; Mørup and Schmidt, 2012; Cha and Cho, 2012), used in topic modelling (Blei et al., 2003; Broderick et al., 2013) and includes extensions such as the Indian buffet process (Ghahramani and Griffiths, 2006). It is appealing since it jointly models the number of blocks and block memberships via one parameter γ. The expected number of blocks is available in closed form in Equation (2.5) where Ψ is the digamma function.

$$\mathbb{E}[\kappa] = \gamma \left(\Psi(\gamma + N) - \Psi(\gamma) \right) \tag{2.5}$$

However, the marginal distribution on the number of blocks, $p(\kappa)$, depends on the Stirling numbers of the first kind. The computation of Stirling numbers is non-trivial (Antoniak, 1974). As such, prior specification is more precise under the DMA specification, since it allows a distribution to be placed on the number of blocks independent to the block size distribution.

For use in the following samplers, both the density, and marginal densities for a single node are required. These are available in Equation (2.6) for the DMA and Equation (2.7) for the CRP.

$$\begin{aligned}
f(\boldsymbol{z}_i = l | \bar{\boldsymbol{z}}, \eta) &= \sum_{\kappa^*=\eta}^{\eta+1} \frac{f(\boldsymbol{z}|\kappa^*)\, \pi_0(\kappa^*|\eta)}{\pi_0(\bar{\boldsymbol{z}}|\eta)} \\
&= \frac{\Gamma(\gamma)^\eta}{\Gamma(\gamma\eta)} \frac{\Gamma(\gamma\eta + M)}{\prod_{k=1}^{\eta} \Gamma(\gamma + M_k)} \\
&\quad \times \left[\frac{\Gamma(\gamma\eta)}{\Gamma(\gamma)^\eta} \frac{\prod_{k=1}^{\eta} \Gamma(\gamma + M_k + z_{ik})}{\Gamma(\gamma\eta + M + 1)} \pi_0(\eta|\eta) \right. \\
&\quad \left. + \frac{\Gamma(\gamma\eta + \gamma)}{\Gamma(\gamma)^{\eta+1}} \frac{\prod_{k=1}^{\eta+1} \Gamma(\gamma + M_k + z_{ik})}{\Gamma(\gamma\eta + \gamma + M + 1)} \pi_0(\eta + 1|\eta) \right] \\
&= \begin{cases} \pi_0(\eta + 1|\eta)\, \frac{\boldsymbol{B}(\gamma+1, M+\eta\gamma)}{\boldsymbol{B}(\gamma, \eta\gamma)} & \text{if } l = \eta + 1 \\ \pi_0(\eta|\eta)\, \frac{M_l + \gamma}{M + \eta\gamma} + \pi_0(\eta+1|\eta)\, \frac{\boldsymbol{B}(\gamma+1, M+\eta\gamma)}{\boldsymbol{B}(\gamma, \eta\gamma)} \left(1 + \frac{M_l}{\gamma}\right) & \text{o.w.} \end{cases}
\end{aligned} \tag{2.6}$$

Where $\bar{\boldsymbol{z}}$ is the set of nodes without node i, $M = N - 1$ is the number of nodes without i, η is the number of blocks defined by the set $\bar{\boldsymbol{z}}$ and M_k is the number of nodes in block k from the set $\bar{\boldsymbol{z}}$. As for the CRP:

$$\begin{aligned}
f(\boldsymbol{z}, \kappa|\gamma) &= \frac{\Gamma(\gamma)\, \gamma^\kappa}{\Gamma(\gamma + N)} \prod_{k=1}^{\kappa} \Gamma(N_k) \\
f(z_{ik} = 1 | \boldsymbol{z}_{-i}, \kappa, \gamma) &= \begin{cases} \frac{N'_k}{N-1+\gamma} \text{ if } k \leq \kappa, \\ \frac{\gamma}{N-1+\gamma} \text{ if } k \text{ a new block.} \end{cases}
\end{aligned} \tag{2.7}$$

2. Arbitrary edge-states and unknown number of blocks in the SBM

This section closes with a comparison between the CRP and DMA priors. One downside to using the CRP is that generated blocks structures are skewed towards one large block with multiple smaller blocks. By keeping an explicit DMA prior the possible block structures are more flexible and interpretation of parameters is easier, since they are independent features of the model. Firstly, by choosing a specific prior distribution F_0 for κ, much more can be said about the number of blocks; this allows more informative priors when modelling. Secondly, explicit choices can be made for the distribution of nodes to these blocks via γ. Therefore, for flexible modelling, the DMA is to be preferred. In this work, the DMA prior is used with F_0 the distribution obtained by adding one to a Poisson random variable. As a consequence, $f_0(K = \kappa) = \frac{\delta^{\kappa-1}\exp(-\delta)}{(\kappa-1)!}$ for $k = 1, 2, \ldots$. A comparison of block structures generated under each model is shown in Figure 2.1. In each diagram, six block structures have been generated for 100 nodes from two settings in each of the CRP and DMA model. Each structure is represented as a set of rectangles, with each rectangle denoting one block. The height of the rectangle depicts the number of nodes assigned to that block. In all cases, the blocks have been sorted by size, hence, the more uneven the block structure, the more curved the plot. As such, notice that the CRP generates uneven block structures, especially for smaller γ. However, to generate more even blocks, a high value for γ is required; this generally generates more blocks as well. This is due to the low probability of generating even blocks under the CRP. As for the DMA, in both cases, the expected number of blocks is 5, but γ is set as 1 or 5. The block structures generated with $\gamma = 1$ are in the lower left. The size of blocks in this case is drawn uniformly across all block sizes with the given number of blocks. In the lower right, a DMA(5,5) is shown. In this case the block sizes are drawn from a symmetric Dirichlet(5) distribution. This concentrates the block size distribution to more even blocks, yielding plots with a straight diagonal.

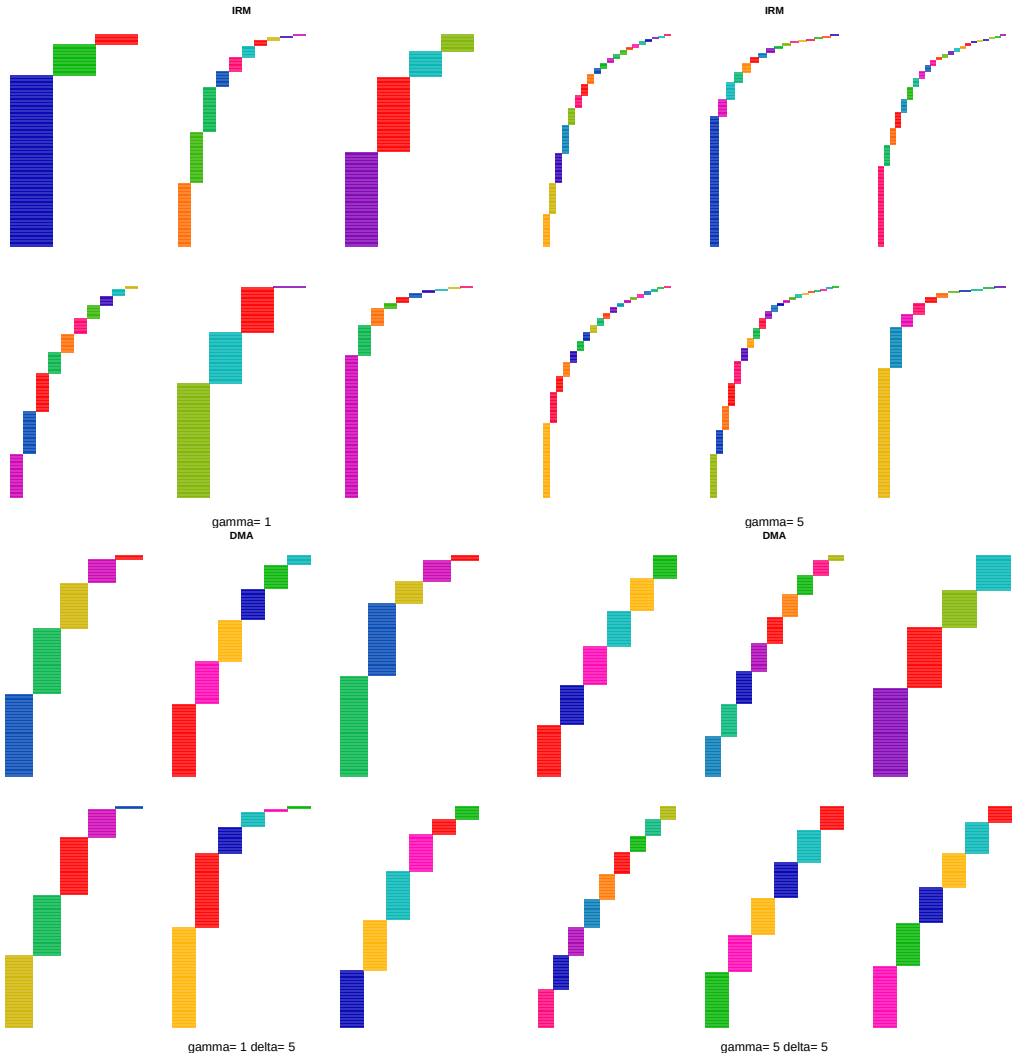


Figure 2.1.: Comparison of block structures. Top left CRP(1), top right CRP(5). Bottom left: DMA(1,5), bottom right: DMA(5,5).

2.3. Dirichlet process sampler

In this section a Dirichlet process sampler for the restricted SBM is given. This adapts the Dirichlet process sampler for clustering of Neal (2000) for the SBM. Running such a sampler for a large number of steps will draw samples from the posterior distribution of $(\kappa, \boldsymbol{z}, \boldsymbol{\theta})$. In such a procedure, the block membership of each node is updated in turn. When considering a node i, it is either assigned to one of the current κ blocks, or starts a new block denoted by k^*. For the current blocks, a parameter $\boldsymbol{\theta}_k$ is used from the current state of the sampler when calculating the likelihood. For the new block however, a parameter $\boldsymbol{\theta}_{k^*}$ is simply drawn from the prior.

2. Arbitrary edge-states and unknown number of blocks in the SBM

Neal (2000) introduce a Dirichlet process sampler for clustering with non-conjugate models. When clustering, assigning a data point to a new cluster requires a new parameter $\boldsymbol{\theta}_{k^*}$. However, in the case of the SBM, there is a key difference: under the clustering problem each data point belongs to one component, in the SBM each node i belongs to one block, which in turn decides the distribution which the $N-1$ edges with end-node i follow. This will greatly influence the likelihood of the point being assigned to the new cluster k^*. An interesting property for the restricted SBM is that assigning a single node to a new block leaves no within-block edges, hence, any value of $\boldsymbol{\theta}_{k^*}$ will not affect the likelihood. This can help the sampler in exploration, since creating a new block is not down-weighted by poor parameter values $\boldsymbol{\theta}_{k^*}$, such values may be updated by other moves of the sampler. The specifics of the algorithm are now discussed.

Firstly, let $(\kappa^s, \boldsymbol{z}^s, \boldsymbol{\theta}^s)$ be the state of the sampler after s steps. The process by which the next state $s+1$ is generated is described for each of the components $\kappa, \boldsymbol{z}$ and $\boldsymbol{\theta}$. The update process for parameter values $\boldsymbol{\theta}$ has been chosen as a Metropolis-Hastings random walk on a transformed scale. In the following this is referred to as "Update", which takes the current state of the sampler and proposes new values $\boldsymbol{\theta}^{s+1}$. In the analysis, a Metropolis-Hastings procedure is applied with symmetric Gaussian proposals on a transformed scale. That is, $\boldsymbol{\theta}'_k = m^{-1}(m(\boldsymbol{\theta}_k) + \sigma\xi)$ where ξ is a draw from a standard normal distribution and m is an isomorphism from $\boldsymbol{\Theta}$ to $\mathbb{R}$.

The difficult part of the Dirichlet process sampler is the update for block memberships $\boldsymbol{z}$ and, as a consequence, the number of blocks κ. This makes use of a Metropolis update for each node in turn. Let i be the node whose block membership, $\boldsymbol{z}_i$, is to be updated. By choosing the marginal prior as the proposal distribution for the block membership of i, this will cancel in the acceptance probability. Therefore, the proposed block membership of i is drawn as $\boldsymbol{z}'_i \sim \text{Multinomial}(\boldsymbol{p})$, where $p_k = f(z_{ik} = 1 | z_{-i})$. Note that the prior here is marginalised over κ as well as the parameters $\boldsymbol{\theta}$. Therefore, the DMA prior cannot be used in this form.

Therefore, all analysis with the DP sampler will use a CRP prior in this paper. The marginal density for the CRP is given in Equation (2.7). Next, if i is proposed to start a new block k', then a value for $\boldsymbol{\theta}_{k'}$ is required for this new block. In this case, denote $\boldsymbol{\theta'} = \boldsymbol{\theta} \cup \theta_{k'}$ as the proposed set of parameters. By drawing $\boldsymbol{\theta}_{k'}$ from its prior distribution G_0, this also cancels in the acceptance probability. On the other hand, if z_i is currently a member of a singleton block l and $z_i' \neq l$, then block l is removed from the model. In this case $\boldsymbol{\theta'} = \boldsymbol{\theta} \backslash \theta_l$. In the case where the number of blocks is unchanged when proposing z_i', the proposed parameter values $\boldsymbol{\theta'} = \boldsymbol{\theta}$. Finally, $\boldsymbol{\theta}^{s+1}, \boldsymbol{z}_i^{s+1}$ is set to $\boldsymbol{\theta'}, \boldsymbol{z}_i'$ with probability A in Equation (2.8), and to $\boldsymbol{\theta}^s, \boldsymbol{z}_i^s$ otherwise. It remains to compute the acceptance probability. Since updates are proposed from the prior distribution, this cancels in the posterior leaving a likelihood ratio. Recall that G is the distribution for the edge-states with density function g, and thus $g(\boldsymbol{E}|\boldsymbol{z}, \boldsymbol{\theta})$ is the likelihood function then:

$$\begin{aligned} A(\boldsymbol{z}_i \to \boldsymbol{z}_i') &= \frac{\pi\big(\boldsymbol{z}_i'|\boldsymbol{E}, \boldsymbol{z}_{-i}^s, \boldsymbol{\theta'}\big)}{\pi\big(\boldsymbol{z}_i^s|\boldsymbol{E}, \boldsymbol{z}_{-i}^s, \boldsymbol{\theta}^s\big)} \frac{q(\boldsymbol{z}_i^s|\boldsymbol{z}_i')}{q(\boldsymbol{z}_i'|\boldsymbol{z}_i^s)} \\ &= \frac{g\big(\boldsymbol{E}|\boldsymbol{z}_i', \boldsymbol{z}_{-i}^s, \boldsymbol{\theta'}\big)}{g\big(\boldsymbol{E}|\boldsymbol{z}_i^s, \boldsymbol{z}_{-i}^s, \boldsymbol{\theta}^s\big)}. \end{aligned} \tag{2.8}$$

The procedure is summarised in Algorithm 2.3. Note that at a given step a node may start a new block, increasing κ. A node may also be the only node in a block, but then be reassigned to a block containing other nodes, decreasing κ. An important case to consider is when a node i is currently a member of a singleton block, say k: a block with only one node. In such a situation, assigning i to k' is the same as i starting a new block in terms of the model, since block labels have no impact on the model. A simple remedy is to set the probability of reassigning node i to block k as zero.

2.4. Split-merge sampler

In this section a split-merge sampler, based on reversible jump Markov chain Monte Carlo is provided. This is an alternative approach to the Dirichlet process sampler

Algorithm 2.3 Metropolis-within-Gibbs sampler for restricted SBM with unknown κ: Dirichlet process sampler.

Inputs: Edge-states $\boldsymbol{E}$, prior parameters $\boldsymbol{\alpha}, \gamma, \delta$.
Draw $\kappa^0, \boldsymbol{z}^0 \sim F_0(\cdot|\gamma, \delta)$.
Draw $\boldsymbol{\theta}^0 \sim G_0(\cdot|\boldsymbol{\alpha})$.
for $s = 1, \ldots, S$ **do**
  Draw $\boldsymbol{\theta}^s \sim \text{Update}(\cdot|\boldsymbol{E}, \kappa^{s-1}, \boldsymbol{z}^{s-1}, \boldsymbol{\theta}^{s-1}, \boldsymbol{\alpha})$
  Let $\kappa^s = \kappa^{s-1}$
  for $i = 1, \ldots, N$ **do**
    for $k = 1, \ldots, \kappa^s + 1$ **do**
      Let $p_k = f(z_{ik} = 1|z_{-i})$
    end for
    if i currently belongs to a singleton block k **then**
      Let $p_k = 0$
    end if
    Draw $\boldsymbol{z}_i' \sim \text{Multinomial}(\boldsymbol{p})$
    if $\boldsymbol{z}_i' = \kappa^s + 1$ **then**
      Draw $\boldsymbol{\theta}_{\kappa^s+1} \sim G_0(\boldsymbol{\alpha})$
    end if
    Calculate A from Equation (2.8)
    Draw $Y \sim \text{Bernoulli}(A)$
    if $Y = 1$ **then**
      Let $\boldsymbol{z}_i^s = \boldsymbol{z}_i'$
    else
      Let $\boldsymbol{z}_i^s = \boldsymbol{z}_i^{s-1}$
    end if
    Let $\kappa^s = \sum_{k=1}^{\infty} \mathbb{I}\left[\sum_{i=1}^{N} \boldsymbol{z}_{ik}^s > 0\right]$
  end for
  Store sample $(\boldsymbol{z}^s, \boldsymbol{\theta}^s)$.
end for

introduced in Section 2.3. Both methods draw samples from the posterior distribution of $(\kappa, \boldsymbol{z}, \boldsymbol{\theta})$. However, the split-merge sampler can perform more drastic changes to the state space compared to the Dirichlet process sampler. This can have major benefits when exploring the posterior distribution. For example, the block membership of each node is updated in turn under the Dirichlet process sampler; this can lead to cases where it is difficult to separate nodes which should belong to different blocks. Consider two "true" blocks k and l with $n_k \geq n_l$ nodes. Furthermore, consider a state s of the DPS where all nodes in blocks k and l are assigned to one block k^s (which contains only these nodes). To separate the nodes within k^s to the true blocks k and l requires at least n_l steps, each of which takes the nodes assigned to k^s and assigns them to a new block l^s. However, each of these moves is quite unlikely: especially if the parameters $\boldsymbol{\theta}_k, \boldsymbol{\theta}_l$ are close to $\boldsymbol{\theta}_0$. On the other hand, if all nodes could be moved at once, then the proposal would be more likely to be accepted. This is a common problem with Gibbs sampling algorithms: the one-at-a-time nature of the procedure means large changes in posterior space are unlikely, even if the combined changes increase the posterior considerably. This phenomenon can cause the Gibbs sampler to get "stuck" in local modes of the posterior. One way to address this problem is to use a split-merge sampler.

Split-merge samplers have been developed for general mixture models (Green and Richardson, 2001), with emphasis on the canonical mixture of normal densities. In a standard parametric mixture model, each data point is believed to be drawn from a component of the mixture. Each component has a different form, either different distributions or different parameter values. A split-merge sampler applied to such a data set explores the possible assignments of data points to components by successively proposing to either merge two components together, or splitting one component into two. Care must be taken in designing such proposals: it must be an isomorphism and differentiable to ensure the validity of the underlying Markov chain. Furthermore, to be efficient, the proposal must ensure that a proposed structure has similar posterior support to the current structure to

ensure a reasonable acceptance probability.

In this paper, a split-merge sampler for the SBM is introduced. Firstly, notice that the SBM differs from the standard mixture model considered above. In the above, each data point belongs to one component; in the SBM each node belongs to one block, however the data in a network is the set of edge-states. These are influenced by the block membership of the nodes. Therefore, when considering split-merge samplers for the SBM, multiple edge-states are affected by changing the block membership of one node.

On top of this, in a mixture model a data point may belong to a component to which no other data points are assigned. If a split move proposes a split such that a component contains only one data point, it will be penalised for creating an extra component, but rewarded for finding a good fit in terms of likelihood. In the mixture of normal examples, if a data point z_i is proposed to start a component with form $N(z_i, \epsilon)$ (for ϵ small), the likelihood will be increased. Similarly, a node i may be the only member of a block, say k. Such nodes are referred to as *singletons*. However, in this case, there are no edge-states governed by the k^{th} block (since self-edges are not considered in this paper). As such, the likelihood is not influenced by a parameter value for block k. On the contrary, node i is a singleton, thus all its edges are between-block edges, and thus the edge-states E_{ij} for $j \neq i$ are governed by parameter $\boldsymbol{\theta}_0$.

The differences highlighted above make a split-merge sampler for the SBM different from a standard split-merge sampler. These differences need to be considered for the design of a successful sampler.

The remainder of this section introduces the split-merge sampler for the restricted SBM, taking account of the issues highlighted above. The sampler consists of four moves:

- resampling parameter values,
- splitting or merging blocks,

- adding or deleting an empty block,
- reassigning nodes to the current set of blocks.

Let $(\kappa^s, \boldsymbol{z}^s, \boldsymbol{\theta}^s)$ be the current state of the parameters in step s of the sampler. Resampling parameter values is identical to the process under the DP sampler of Section 2.3. The difficult proposals are the trans-dimensional: splitting and merging.

Firstly, the merge proposal is described. This takes the state $(\kappa^s, \boldsymbol{z}^s, \boldsymbol{\theta}^s)$ and proposes a new state $(\kappa', \boldsymbol{z}', \boldsymbol{\theta}')$ Such a move will reduce the number of blocks by one; hence, $\kappa' = \kappa^s - 1$. Two block must be chosen to merge, k and l, say. There is freedom of choice in the mechanism for choosing k and l: random, proportional to block size, inversely proportional to block size, etc. In this paper, k and l are chosen at random for simplicity. Given k and l, the parameter and block memberships must be proposed. Block memberships are deterministic: any node belonging to either block k or block l at step s is now assigned to block k'. Finally, the parameter values are proposed. Following the recommendations of Green and Richardson (2001), proposing a value $\boldsymbol{\theta}'_{k'}$ with similar explanatory power as $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_l$ should ensure that $\boldsymbol{\theta}'_{k'}$ is well supported in the posterior. A simple approach is to simply take the mean value: $\boldsymbol{\theta}'_{k'} = \boldsymbol{\theta}_k/2 + \boldsymbol{\theta}_l/2$. However, to allow more flexibility in the sampler, an uneven merge is considered with a sampler parameter $w \in [0, 1]$ giving a weighted mean of parameter values as in Equation (2.9).

$$\boldsymbol{\theta}'_{k'} = w\boldsymbol{\theta}_k + (1 - w)\boldsymbol{\theta}_l \tag{2.9}$$

The split move is the inverse of a merge. Hence, when splitting block k' back into block k and l, the inverse of Equation (2.9) is required. On top of this, an auxiliary variable u is need to match the dimension of the parameter space. In the examples u is drawn from a Normal$(0, \sigma^2)$ distribution for $\sigma = 1$. The value of σ dictates the extent to which the new block parameters can differ, with larger values forcing the proposed parameters further apart. Letting u be the weighted

difference of the parameters k and l yields a pair of simultaneous equations, as in Equation (2.10). Solving these gives values for the new block parameters $\boldsymbol{\theta}_k$ and $\boldsymbol{\theta}_l$ in Equation (2.11).

$$\begin{aligned} \boldsymbol{\theta}'_{k'} &= w\boldsymbol{\theta}_k + (1-w)\boldsymbol{\theta}_l \\ u &= w\boldsymbol{\theta}_k - (1-w)\boldsymbol{\theta}_l \end{aligned} \tag{2.10}$$

$$\begin{aligned} \boldsymbol{\theta}_k &= \frac{\boldsymbol{\theta}'_{k'} + u}{2w} \\ \boldsymbol{\theta}_l &= \frac{\boldsymbol{\theta}'_{k'} - u}{2(1-w)} \end{aligned} \tag{2.11}$$

However, Equation (2.11) does not guarantee values in the parameter space. For example, if $\boldsymbol{\theta}'_{k'}$ is a rate parameter (which must be positive), $\boldsymbol{\theta}_l$ is not guaranteed to remain positive. Rather than choose u and w conditional on a suitable $\boldsymbol{\theta}$, a matching function $m : \boldsymbol{\theta} \to \mathbb{R}$ is used. This ensures that, when splitting, the inverse m^{-1} yields parameter values in the correct space. Possible matching functions for a given parameter space are shown in Table 2.1.

Table 2.1.: Possible matching functions to ensure parameters lie in the correct space.

Range for $\boldsymbol{\theta}$	Possible matching function m
(∞, ∞)	$m(x) = x$
$[0, \infty)$	$m(x) = \log(x)$
$[0, 1]$	$m(x) = \text{logit}(x) = \log(x) - \log(1-x)$

As a consequence, the sampler uses Equation (2.12) with sampler parameter w to merge blocks k and l into block k'. Whereas to split a block k', equations (2.13) and (2.14) are used together with simulation parameters w' and u'. In the examples w and w' are drawn from Unif$(0, 1)$ and u' from Normal$(0, 1)$. Note that the dimension matching criteria of RJMCMC (Green, 1995) is achieved since the vectors $(\boldsymbol{\theta}'_{k'}, u', w')$ and $(\boldsymbol{\theta}_k, \boldsymbol{\theta}_l, w)$ contain the same number of elements.

$$m(\boldsymbol{\theta}'_{k'}) = wm(\boldsymbol{\theta}_k) + (1-w)m(\boldsymbol{\theta}_l) \tag{2.12}$$

$$m(\boldsymbol{\theta}_k) = \frac{m(\boldsymbol{\theta}'_{k'}) + u'}{2w'} \tag{2.13}$$

$$m(\boldsymbol{\theta}_l) = \frac{m(\boldsymbol{\theta}'_{k'}) - u'}{2(1-w')} \tag{2.14}$$

Therefore, the merge proposal takes two blocks k, l, proposes a parameter value $\boldsymbol{\theta}'_{k'}$ based on $\boldsymbol{\theta}_k, \boldsymbol{\theta}_l, w$ and u, reassigns nodes in block k or l to block k' and decreases the value of κ by one. The proposed state $(\kappa', \boldsymbol{z}', \boldsymbol{\theta}')$ is then accepted as the new state of the RJMCMC sampler with probability A_{merge} computed in the following and shown in Equation (2.18).

It remains to describe the split proposal in detail. Again let the current state of the sampler be $(\kappa^s, \boldsymbol{z}^s, \boldsymbol{\theta}^s)$; as such the proposed state is $(\kappa', \boldsymbol{z}', \boldsymbol{\theta}')$. A split move will divide a single block into two blocks, thereby increasing the number of blocks by one; hence $\kappa' = \kappa^s + 1$. Firstly, a block k is chosen to split at random from the current set of κ^s blocks. Given k, the parameter and block memberships must be proposed with new blocks k', l'. Discussion of how to generate parameters can be found above in the merge procedure, parameters $\boldsymbol{\theta}'_{k'}, \boldsymbol{\theta}'_{l'}$ are given by Equation (2.13) and (2.14) respectively.

Finally, the proposed block structure $\boldsymbol{z}'$ is considered. In Green and Richardson (2001), data points are assigned to block k' or l' proportional to the likelihood. That is, for a density f and data $\boldsymbol{y}$:

$$q(z'_i = k') = \frac{f(y_i|z'_i = k')}{f(y_i|z'_i = k') + f(y_i|z'_i = l')}$$

Taking a similar approach in the SBM is not possible due to the fact that edge-states exist between all nodes. For example, under the SBM, $g(E_{ij}|z'_i = k')$ can only be calculated with knowledge of z'_j for $j \neq i$; this is not known for the set of nodes being split. The quantity can be calculated in principle by looking at all the possible allocations of the nodes in block k to k' and l'. Let $\mathcal{I}$ be the set of

nodes to be split, and $\sigma(\mathcal{I})$ be a permutation of $\mathcal{I}$. Each of the nodes in $\mathcal{I}$ will be assigned in the order $\sigma(\mathcal{I})$. Let "$< i$" represent indices preceding i and hence $\sigma(\mathcal{I})_{<i}$ is the set of nodes already assigned before i. Therefore, when assigning i, the following can be calculated:

$$q(z'_i = k') = \frac{f\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)}{f\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big) + f\big(E|z'_i = l', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)}. \tag{2.15}$$

Taking the product of this over each $i \in \sigma(\mathcal{I})$ yields the probability of assigning the nodes in the sequence $\sigma(\mathcal{I})$:

$$q(\boldsymbol{z}'_{\mathcal{I}}) = \prod_{i \in \sigma(\mathcal{I})} \frac{g\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)}{g\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big) + g\big(E|z'_i = l', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)}. \tag{2.16}$$

However, this depends on the order σ, hence to obtain the probability of a given sequence, this value is averaged over all permutations of $\mathcal{I}$. Let $\Sigma(\mathcal{I})$ be the set of all permutations over the set $\mathcal{I}$ and $\sigma(\mathcal{I})$ be a given permutation, then the probability of the allocation $\boldsymbol{z}'$ is:

$$q(\boldsymbol{z}') = \frac{1}{|\mathcal{I}|!} \sum_{\sigma \in \Sigma(\mathcal{I})} \prod_{i \in \sigma(\mathcal{I})} \frac{g\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)}{g\big(E|z'_i = k', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big) + g\big(E|z'_i = l', \boldsymbol{z}'_{<i}, \boldsymbol{z}'_{-\mathcal{I}}, \boldsymbol{\theta}'\big)} \tag{2.17}$$

This is computationally infeasible to calculate. Notice that it is a mean over all permutation of the set $\mathcal{I}$, hence, an unbiased estimate can be obtained by taking a sample average over some subset of permutations, specifically a single permutation. Therefore, in practise, to split a block k, the nodes to be split are permuted at random. Next, they are sequentially allocated to the new blocks k' or l' with the probability given in Equation (2.15). This contributes to the acceptance probability by the amount in Equation (2.17), which is an unbiased estimate of the value in Equation (2.16). Finally, the proposed split is accepted as the next state of the RJMCMC sampler with probability A_{split} in Equation (2.18).

Now each of the proposal distributions are described, it remains to compute

the acceptance probabilities A_{merge} and A_{split}. Since a merge move is the inverse of a split move, $A_{merge} = 1/A_{split}$, hence only A_{split} is derived. The acceptance probability can be split into the following parts: posterior density ratio, proposal density ratio, ratio of densities of auxiliary variables, and the Jacobian; as such A_{split} has the general form:

$$\begin{aligned} A_{split} &= \frac{\pi(\kappa+1, \boldsymbol{z}', \boldsymbol{\theta}'|E)}{\pi(\kappa, \boldsymbol{z}, \boldsymbol{\theta}|E)} \frac{q(\kappa, \boldsymbol{z}, \boldsymbol{\theta}|\kappa+1, \boldsymbol{z}', \boldsymbol{\theta}')}{q(\kappa+1, \boldsymbol{z}', \boldsymbol{\theta}'|\kappa, \boldsymbol{z}, \boldsymbol{\theta})} \frac{q(w)}{q(u', w')} J_{split} \\ &= \frac{\pi(\kappa+1, \boldsymbol{z}', \boldsymbol{\theta}'|E)}{\pi(\kappa, \boldsymbol{z}, \boldsymbol{\theta}|E)} \frac{q(merge|\kappa+1)}{q(split|\kappa)} \frac{q(k', l')}{q(k)} \frac{q(w')}{q(w, u)} \frac{1}{q(\boldsymbol{z}'|\boldsymbol{\theta}')} J_{split} \end{aligned} \tag{2.18}$$

where $q(split|\kappa)$ and $q(merge|\kappa)$ are the probabilities of proposing a split or merge move given the current state of the sampler contains κ blocks. These are chosen as $1/2$ where possible. That is $q(split|\kappa = 1) = 1$ and $q(merge|\kappa = 1) = 0$ since merging is impossible when there is only one block.

Notice the sampler allows splitting of singleton blocks and leaving a block empty during a split. Due to the reversible nature of the sampler, these empty blocks can then be merged back with other blocks. In the mean time nodes can be assigned to such empty blocks via the Gibbs-like allocation move, described in the following.

Lastly, J_{split} is the Jacobian of the split proposal given in Equation (2.19) and p is the dimensionality of each $\boldsymbol{\theta}_k$.

$$J_{split} = \begin{vmatrix} \frac{\partial \boldsymbol{\theta}'_{k'}}{\partial \boldsymbol{\theta}_k} & \frac{\partial \boldsymbol{\theta}'_{l'}}{\partial \boldsymbol{\theta}_k} \\ \frac{\partial \boldsymbol{\theta}'_{k'}}{\partial u'} & \frac{\partial \boldsymbol{\theta}'_{l'}}{\partial u'} \end{vmatrix} = \left| \frac{\nabla m(\boldsymbol{\theta}'_{k'}) \, \nabla m(\boldsymbol{\theta}'_{l'})}{\nabla m(\boldsymbol{\theta}_k) \, (2w(1-w))^p} \right| \tag{2.19}$$

Hence, in the examples, where specific choices for u', w', w and $q(merge)\,, q(split)$

have been made, the acceptance probabilities reduce to:

$$\begin{aligned} A_{split} =& \frac{\pi(\kappa+1, \boldsymbol{z}', \boldsymbol{\theta}'|E)}{\pi(\kappa, \boldsymbol{z}, \boldsymbol{\theta}|E)} \frac{1}{1+\mathbb{I}[\kappa=1]} \frac{2}{\kappa+1} \\ & \times \frac{1}{\phi(u'|0,\sigma^2)} \frac{1}{q(\boldsymbol{z}'|\boldsymbol{\theta}')} \left| \frac{\nabla m(\boldsymbol{\theta}'_{k'}) \nabla m(\boldsymbol{\theta}'_{l'})}{\nabla m(\boldsymbol{\theta}_k) (2w(1-w))^p} \right| \\ A_{merge} =& \frac{\pi(\kappa-1, \boldsymbol{z}', \boldsymbol{\theta}'|E)}{\pi(\kappa, \boldsymbol{z}, \boldsymbol{\theta}|E)} (1+\mathbb{I}[\kappa=2]) \frac{\kappa}{2} \\ & \times \phi\big(u|0,\sigma^2\big)\, q(\boldsymbol{z}|\boldsymbol{\theta}) \left| \frac{\nabla m(\boldsymbol{\theta}'_{k'}) (2w(1-w))^p}{\nabla m(\boldsymbol{\theta}_k) \nabla m(\boldsymbol{\theta}_l)} \right| \end{aligned}$$

To allow the sampler to explore the parameter space, an additional two moves are included. A Gibbs like move, which allocates each node to a block proportional to the posterior density, and a move that allows the addition and deletion of empty blocks.

The Gibbs-like allocation move takes a node i and computes the marginal posterior for i being a member of each of the κ blocks in the current state of the sampler. Since κ is finite, at any given step, a vector $\boldsymbol{p}_i$, of length κ, can be calculated with p_{ik} the probability that node i is proposed to move to block k. Thanks to the structure of the restricted SBM, p_{ik} can be written as the product of two densities: the posterior density of edge-states to nodes in the block k and the posterior density of edge-states to nodes in other blocks as in Equation (2.20).

$$\begin{aligned} p_{ik} &= \pi(z_{ik}=1|\boldsymbol{z}_{-i}, E, \boldsymbol{\theta}) \\ &\propto f(\boldsymbol{z}_{ik}=1|\boldsymbol{z}_{-i}) \prod_{j\neq i} g(E_{ij}|\boldsymbol{z}_j, bz_{ik}=1, \boldsymbol{\theta}) \\ &= f(\boldsymbol{z}_{ik}=1|\boldsymbol{z}_{-i}) \prod_{j\neq i} g(E_{ij}|\boldsymbol{\theta}_k)^{\boldsymbol{z}_{jk}}\, g(E_{ij}|\boldsymbol{\theta}_0)^{1-\boldsymbol{z}_{jk}} \end{aligned} \tag{2.20}$$

Note that proposing to assign i back to the block to which it currently belongs is allowed. Furthermore, since κ is finite, dividing by the sum of $\boldsymbol{p}_i$ is trivial to obtain a probability vector.

The second extension allows the deletion and addition of empty blocks. Notice that the Gibbs allocation move or the split move can leave blocks empty. Waiting for the sampler to merge an empty block with another block can leave empty

blocks in the sampler state for some time (since blocks are chosen to merge with probability $2/\kappa$). Also, due to the need to have values for $\boldsymbol{\theta}$ for empty blocks in case of merging, a merge may not even be successful, due to the parameter averaging in Equation (2.12). Therefore, a "delete empty block" proposal is included. To ensure the steady-state properties of the sampler, the inverse "add empty block" proposal is also required. When choosing a block to delete, it is chosen at random from the current set of empty blocks. When adding an empty block, it is simply labelled $\kappa + 1$. For simplicity, when an add/delete move is attempted, the probability of adding a block is chosen proportional to an algorithm parameter ρ. The probability of choosing to delete an empty block is proportional to the number of empty blocks in the current state, $N_\emptyset$. In this way, steps are not wasted attempting to delete an empty block that does not exist. Note that the likelihood of the edge-states does not change with the addition of empty blocks since all node structure remains identical. When adding a block, a parameter $\boldsymbol{\theta}^*$ is drawn from the prior distribution. Given these points, the acceptance probabilities of the add and delete empty block moves can be calculated as in Equation (2.21).

$$\begin{aligned} A_{add} &= \frac{g(E|\kappa+1,\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\theta}^*)}{g(E|\kappa,\boldsymbol{z},\boldsymbol{\theta}|E)} \frac{\pi_0(\kappa+1,\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\theta}^*)}{\pi_0(\kappa,\boldsymbol{z},\boldsymbol{\theta})} \frac{q(\kappa,\boldsymbol{z},\boldsymbol{\theta}|\kappa+1,\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\theta}^*)}{q(\kappa+1,\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\theta}^*|\kappa,\boldsymbol{\theta})} \\ &= \frac{\pi_0(\kappa+1,\boldsymbol{z})}{\pi_0(\kappa,\boldsymbol{z})} \frac{\rho+N_\emptyset}{\rho(\rho+N_\emptyset+1)} \qquad (2.21) \\ A_{del} &= \frac{\pi_0(\kappa-1,\boldsymbol{z})}{\pi_0(\kappa,\boldsymbol{z})} \frac{\rho(\rho+N_\emptyset)}{\rho+N_\emptyset-1} \qquad (2.22) \end{aligned}$$

The full split-merge algorithm is given in Algorithm 2.4, including the split-merge moves, Gibbs-like reallocation of nodes and the addition or deletion of empty blocks.

Algorithm 2.4 Reversible jump Markov Chain Monte Carlo sampler for the restricted SBM with unknown κ: split-merge algorithm.

Inputs: Edge-states $\boldsymbol{E}$, prior parameters $\boldsymbol{\alpha}, \gamma, \delta$.
Draw $\kappa^0, \boldsymbol{z}^0 \sim F_0(\cdot|\gamma, \delta)$.
Draw $\boldsymbol{\theta}^0 \sim G_0(\cdot|\boldsymbol{\alpha})$.
for $s = 1, \ldots, S$ **do**
  Draw $\boldsymbol{\theta}^s \sim \text{Update}(\cdot|\boldsymbol{E}, \kappa^{s-1}, \boldsymbol{z}^{s-1}, \boldsymbol{\theta}^{s-1}, \boldsymbol{\alpha})$
  Let $\kappa^s = \kappa^{s-1}$
  if κ^s=1 **then**
    Propose a split
  else
    with probability 1/2 propose a split or a merge
  end if
  if There are no empty blocks **then**
    Propose adding an empty block
  else
    with probability $\frac{N_\emptyset}{N_\emptyset+\rho}$ attempt deleting an empty block.
    or with probability $\frac{\rho}{N_\emptyset+\rho}$ attempt adding an empty block.
  end if
  for $i = 1, \ldots, N$ **do**
    for $k = 1, \ldots, \kappa^s$ **do**
      Let $p_k = g(E_{i\cdot}|\boldsymbol{z}_{-i}, z_{ik} = 1, \boldsymbol{\theta})\, f(z_{ik} = 1|\boldsymbol{z}_{-i})$
    end for
    Draw $\boldsymbol{z}_i' \sim \text{Multinomial}(\boldsymbol{p})$
  end for
  Store sample $(\boldsymbol{z}^s, \boldsymbol{\theta}^s, \kappa^s)$.
end for

2.5. Simulated data

In this section, the Dirichlet process sampler from Section 2.3 and split-merge sampler of section 2.4 are demonstrated on simulated data.

2.5.1. Example networks

In this section example data sets are considered with a range of edge-state distributions. Specifically the Bernoulli, Poisson, normal and negative binomial distributions are considered. The negative binomial distribution as used in this paper is a discrete distribution over the number of successful independent Bernoulli trials with probability p which occur until a specified number of failures are obtained. This can be extended to a generalised "number of failures" $r > 0$. The probability mass function is thus:

$$\mathbb{P}[X = x] = \frac{\Gamma(x + r)}{\Gamma(r)\, x!} p^r (1 - p)^x, \text{ for } x = 0, 1, \ldots, r > 0 \text{ and } p \in (0, 1]$$

Notice that the first three edge-state distributions admit conjugate priors. Therefore, existing samplers, such as those introduced by Mørup and Schmidt (2012) and McDaid et al. (2013), could be applied. However, for the negative binomial with both parameters unknown, no conjugate model exists. Each of these networks are formed on the same block-structure of 100 nodes. The block sizes are 19, 23, 27 and 31.

In each network, the DP and split-merge samplers were implemented with reference prior distributions applied to each parameter $\boldsymbol{\theta}$ (where possible). As for the prior on κ and $\boldsymbol{z}$, a CRP(5) is used for the DP sampler whereas a DMA(1,6) prior was used with a Poisson distribution on $\kappa - 1$ for the split-merge sampler. As discussed in Section 2.2.1, this ensures that the support for κ is the positive integers, hence, the prior expected number of blocks is 5 in both cases. The parameter values used for each of the edge-state models is given in Table 2.2. For the network with Bernoulli distributed edge-states, the reference prior Beta(1/2,

1/2) was applied to each parameter $\boldsymbol{\theta}$. In the Poisson network, the reference prior of Gamma(1/2, 0) was used for the rates $\boldsymbol{\theta}$. In the case of normal distributed edge-states, two parameters are required. The reference prior used is proportional to the reciprocal of the variance. In practice, a small value is used in place of a 0 to make the above distributions proper. Finally, in the negative binomial case with both parameters unknown, no reference prior exists, instead a Beta(1/2, 1/2) distribution is placed on the probability parameter p and a Gamma(0.5, 0) distribution for r is used.

Table 2.2.: Simulated data parameter values for each edge-state distribution considered.

Parameter	θ_0	θ_1	θ_2	θ_3	θ_4
Bernoulli(p)	0.05	0.4	0.5	0.6	0.7
Poisson(λ)	1	1	5/3	7/3	3
Normal(μ, σ)	(0, 0.5)	(0.4, 0.5)	(0.4, 0.5)	(4.0, 0.5)	(5.0, 0.5)
Negative binomial(r, p)	(1, 0.5)	(1, 0.5)	(3, 0.5)	(3, 0.5)	(3, 0.5)

In the case of the DP sampler, 5000 steps were taken, with 2500 discarded as burn-in. At each step, the block membership of each node is updated, therefore 250,000 updates to individual block memberships are made in total. This has the possibility to create a new block by moving a single node out of a current block. For the split-merge sampler, 5000 steps were taken with 2500 discarded as burn-in. Notice that at each step, a single split or merge move is proposed which can affect multiple nodes and a Gibbs step is taken to update the membership of each node to the available blocks. Therefore new blocks can be created with multiple nodes by splitting a current block into two new blocks. In all cases, a random walk Metropolis-Hastings step was used on the parameters with variance equal to 0.1 A draw from the prior is taken as the starting point, with both samplers using the same start for a fair comparison.

To evaluate the algorithms' performance the joint probabilities that two nodes

belong to the same block are calculated, after burn-in, as:

$$p_{ij} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathbb{I}[z_{is} = z_{js}] \,, \tag{2.23}$$

where $\mathcal{S}$ are the indices of samples remaining after burn-in. The posterior joint probability that two nodes are in the same block (after burn-in) is displayed for each algorithm for the Bernoulli network in Figure 2.2a and 2.2b, the Poisson network in Figure 2.3a and 2.3b, the normal network in Figure 2.4a and 2.4b, and the negative binomial network in Figure 2.5a and 2.5b,

Another point to consider when evaluating the ability of the samplers is the estimated parameter values. The posterior means of the parameters are shown in Table 2.3, together with a 95% the highest posterior density credible interval. Notice that the values of θ_k, for $k = 0, \ldots, 4$, are good estimates to the true values in Table 2.2.

Notice that the Bernoulli case is rather easy, the parameter values within blocks are quite different to the between-block parameter θ_0, and as such both algorithms perform well here. As for the Poisson case, notice that $\theta_1 = \theta_0$, hence block 1 is not a block as defined in Section 2.2. This is apparent in the plots in Figure 2.3a and 2.3b. In both cases, the nodes simulated with a block membership of 1 do not have a high probability of being in a block with other nodes. This is a case where the DP sampler is performing slightly better than the split-merge sampler: the probabilities are smaller for the DP since it can assign a node to a singleton block more easily. In the split-merge sampler, to assign a node to a singleton requires splitting a block with n nodes into two block with 1 and $n - 1$ nodes respectively; this is a low probability move so is rarely taken by the split-merge sampler.

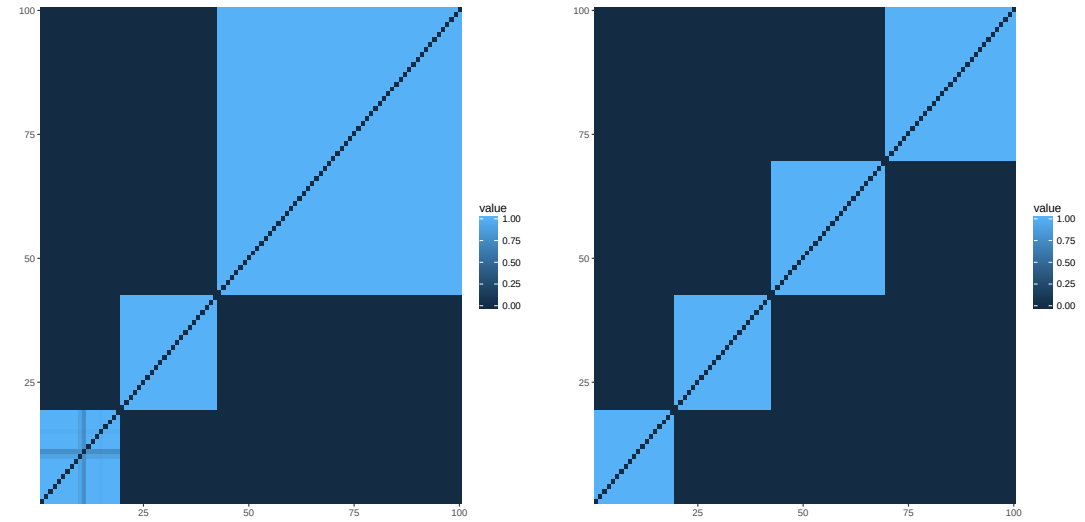
For the network with normally distributed edge-states, the two blocks labelled 3 and 4 are represented in the upper right of the plots in Figure 2.4a and 2.4b. Notice that the DP sampler fails to split these blocks and prefers a posterior where the two blocks are joined as one. However, the split-merge sampler does much better in this case, splitting the two blocks early in the RJMCMC chain. This is a problem

with the DP sampler as mentioned in Section 2.1: due to its Gibbs-like nature it easily gets stuck in local maxima. To split these blocks the DP must assign every node in block 3 to a new block. Each of these moves is unlikely and when taken as a sequence is incredibly unlikely, even though the end of the sequence (with block 3 and 4 split) has high posterior density. The split-merge sampler can propose a split move which separates block 3 and 4 in one move, jumping over the sequence of unlikely moves straight to the high density region of the posterior.

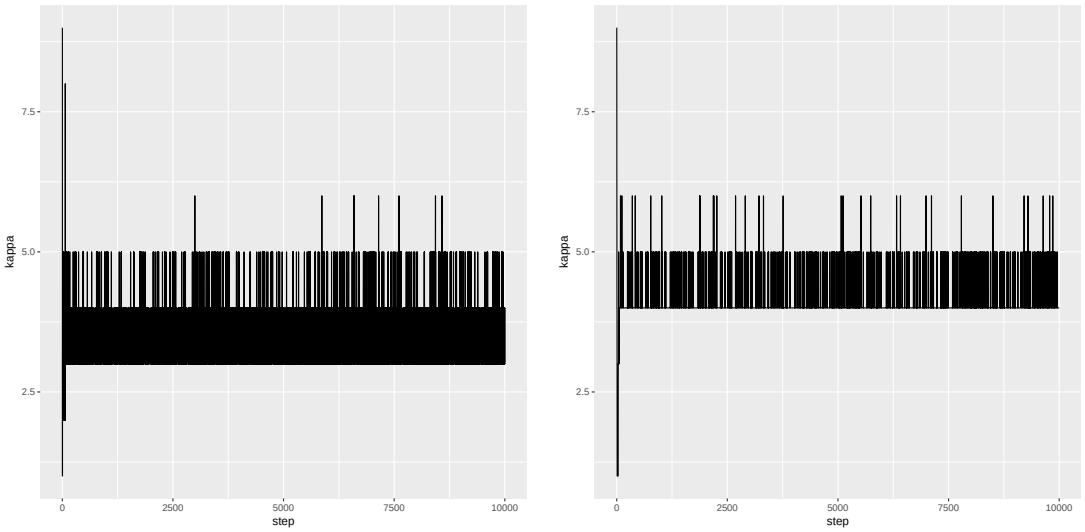
Finally, both samplers perform well in the network with negative binomial distributed edge-states. This network again has a "false" block labelled 1, where $\theta_0 = \theta_1$. Both samplers are able to explore regions where the nodes in simulated block 1 are separate from the other nodes, as seen by the low probability region in both plots in Figures 2.5a and 2.5b.

Table 2.3.: Mean and 95% credible interval (CI) for parameters of example networks.

DP	Bernoulli(p)	Poisson(λ)	Normal(μ)
θ_0	0.056 (0.048, 0.063)	0.982 (0.952, 1.010)	0.017 (0.017, 0.017)
θ_1	0.441 (0.360, 0.526)	43.089 (0.000, 173.426)	0.410 (0.326, 0.463)
θ_2	0.507 (0.448, 0.568)	1.642 (1.432, 1.803)	0.422 (0.373, 0.487)
θ_3	0.524 (0.011, 1.000)	2.198 (2.063, 2.324)	3.962 (3.942, 3.969)
θ_4	0.346 (0.326, 0.372)	3.086 (2.902, 3.232)	5.004 (4.983, 5.049)
DP	NegBin(r)	NegBin(p)	Normal(σ)
θ_0	0.953 (0.894, 0.986)	0.484 (0.473, 0.497)	0.502 (0.502, 0.502)
θ_1	92.693 (0.000, 525.346)	0.863 (0.164, 1.000)	0.505 (0.488, 0.555)
θ_2	2.992 (2.025, 4.020)	0.526 (0.448, 0.617)	0.497 (0.442, 0.571)
θ_3	5.336 (3.984, 7.129)	0.566 (0.498, 0.640)	0.520 (0.488, 0.527)
θ_4	4.607 (3.429, 6.379)	0.466 (0.410, 0.559)	0.484 (0.472, 0.508)
RJ	Bernoulli(p)	Poisson(λ)	Normal(μ)
θ_0	0.052 (0.045, 0.060)	0.976 (0.942, 1.005)	0.015 (0.002, 0.027)
θ_1	0.426 (0.352, 0.487)	0.964 (0.748, 1.155)	0.413 (0.368, 0.462)
θ_2	0.507 (0.447, 0.566)	1.650 (1.468, 1.871)	0.369 (0.289, 0.428)
θ_3	0.640 (0.595, 0.686)	2.201 (2.031, 2.376)	3.981 (3.962, 3.986)
θ_4	0.678 (0.639, 0.722)	3.086 (2.889, 3.241)	4.985 (4.973, 5.019)
RJ	NegBin(r)	NegBin(p)	Normal(σ)
θ_0	0.931 (0.819, 1.065)	0.485 (0.455, 0.513)	0.504 (0.478, 0.521)
θ_1	16.606 (0.000, 58.087)	0.589 (0.314, 0.999)	0.508 (0.489, 0.541)
θ_2	2.892 (1.999, 3.988)	0.520 (0.436, 0.613)	0.502 (0.477, 0.522)
θ_3	5.313 (3.639, 8.187)	0.562 (0.503, 0.686)	0.529 (0.523, 0.551)
θ_4	4.609 (3.887, 5.397)	0.469 (0.418, 0.512)	0.490 (0.476, 0.509)

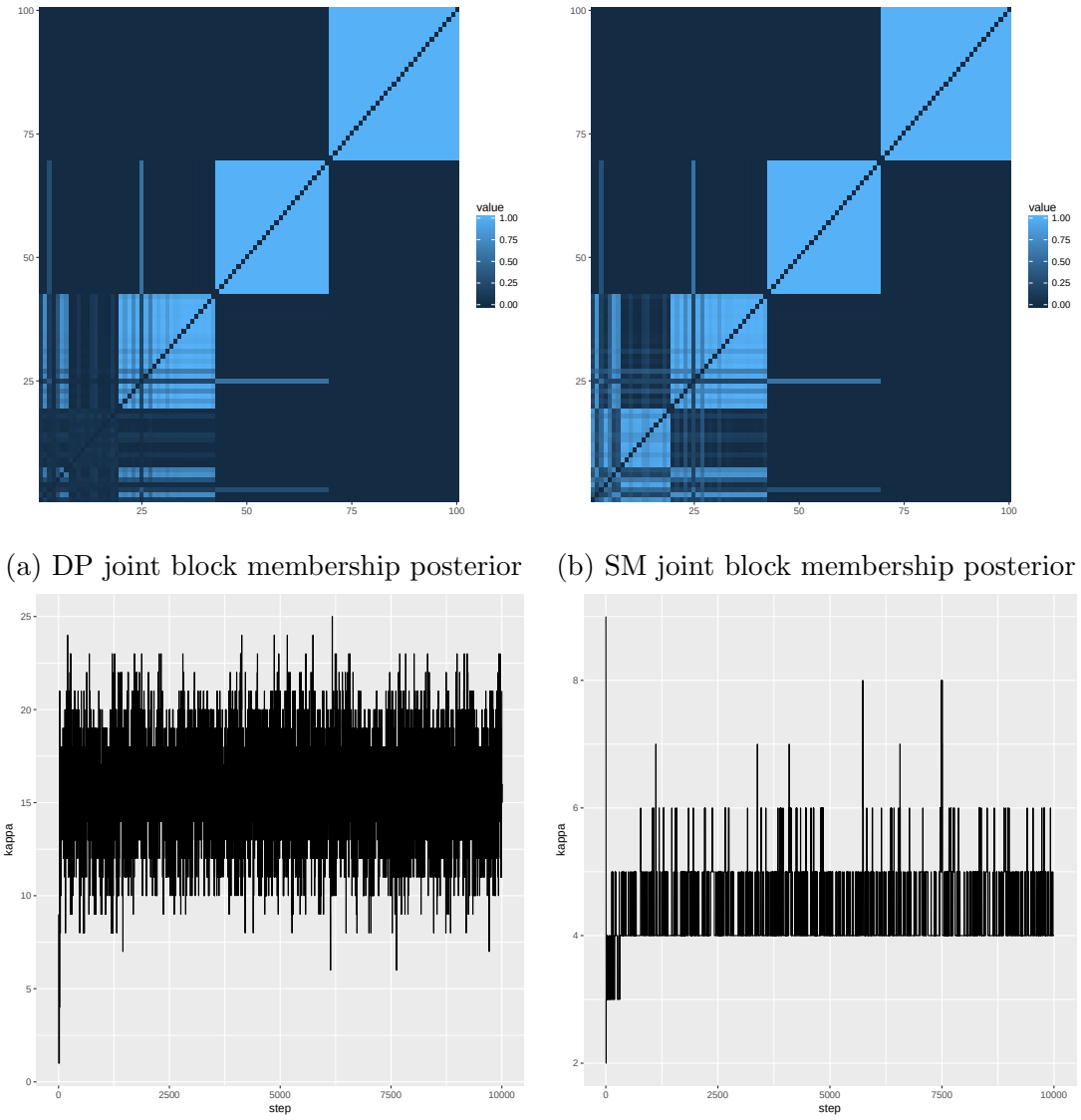


(a) DP joint block membership posterior (b) SM joint block membership posterior



(c) DP trace plot for κ (d) SM trace plot for κ

Figure 2.2.: Posterior summaries for block membership in Bernoulli example network.



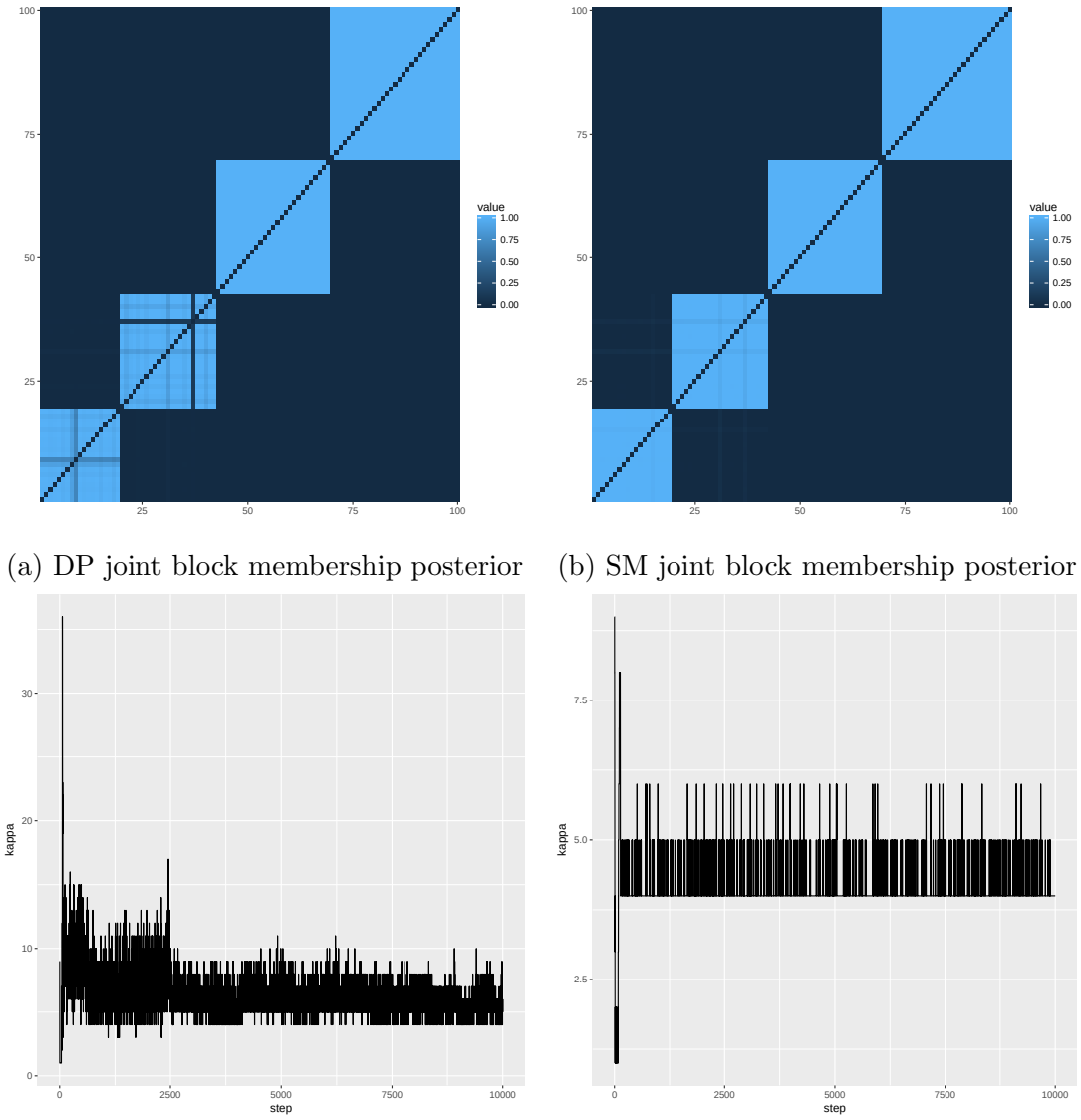
(a) DP joint block membership posterior

(b) SM joint block membership posterior

(c) DP trace plot for κ

(d) SM trace plot for κ

Figure 2.3.: Posterior summaries for block membership in Poisson example network.



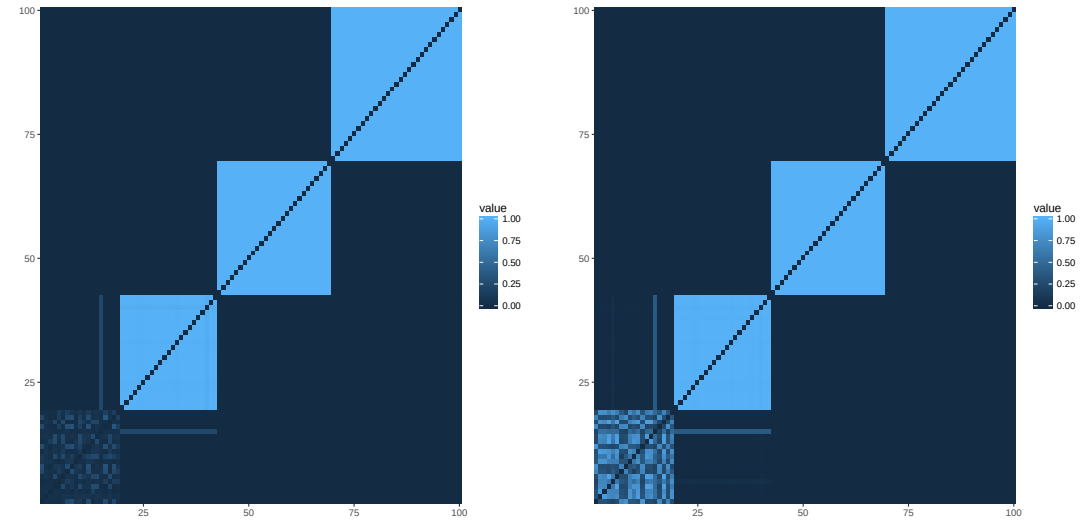
(a) DP joint block membership posterior

(b) SM joint block membership posterior

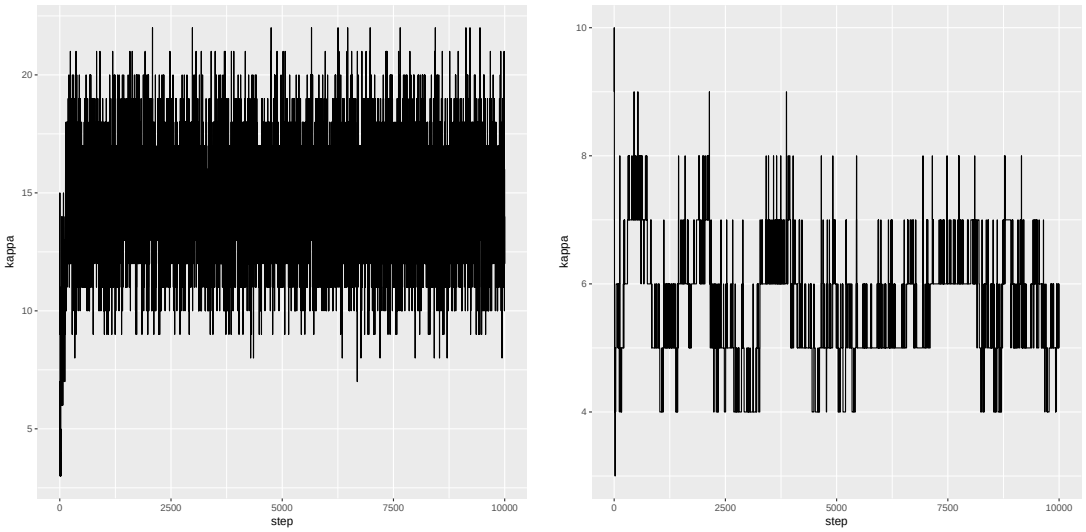
(c) DP trace plot for κ

(d) SM trace plot for κ

Figure 2.4.: Posterior summaries for block membership in normal example network.



(a) DP joint block membership posterior (b) SM joint block membership posterior



(c) DP trace plot for κ (d) SM trace plot for κ

Figure 2.5.: Posterior summaries for block membership in negative binomial example network.

2.5.2. Assessing convergence

Assessing the convergence of a reversible jump Markov chain is non-trivial. Two techniques are applied in this section: (1) applying the Rubin-Gelman convergence statistic (Gelman and Rubin, 1992) to a summary statistic and (2) starting two independent samplers, one with all nodes assigned to one block and the other with each node assigned to different blocks.

In the first case, the mean and variance of the parameter values are used as summary statistics of the sampler performance. These are recorded at every iteration of the sampler. The Rubin-GelmanGelman and Rubin (1992) statistics for the RJMCMC sampler for each model are shown in Table 2.4 based on 30 independent chains. These values are mostly close to 1, indicating that convergence can be expected after a few thousand iterations in the examples networks of Section 2.5.1. The only case where the chain fails to converge is the Bernoulli model under the DP sampler. This is due to the chain failing to split two blocks, and as such spends many iterations with large parameter values drawn from the prior. These in turn lead to large estimates of the variance under the Rubin-Gelman statistic. The DP sampler gave similar results for 20,000 iterations (1.883 for mean of parameter values and 1.228 for variances), indicating that the sampler struggles in the Bernoulli model.

Model	Bernoulli	Poisson
RJ – Mean	1.0004 (1.0007)	1.0090 (1.0094)
RJ – Variance	1.0008 (1.0013)	1.0222 (1.0226)
DP – Mean	2.5885 (3.2001)	1.0001 (1.0004)
DP – Variance	1.3593 (1.5734)	1.0003 (1.0004)
Model	Normal	Negative binomial
RJ – Mean	1.0212 (1.0334)	1.0197 (1.0295)
RJ – Variance	1.0032 (1.0046)	1.0161 (1.0194)
DP – Mean	1.0093 (1.0142)	1.0062 (1.0097)
DP – Variance	1.0347 (1.0542)	1.0010 (1.0016)

Table 2.4.: Rubin-Gelman Statistics (and upper bound of 95% confidence interval) for each model with 30 independent chains of 5000 iterations for RJMCMC and Dirichlet Process samplers.

The second technique for assessing convergence is inspired by perfect simula-

tion: considering two samplers starting at opposite extremes and observing both converging to the same area gives an indication that the underlying Markov chains have converged. This process was used for the simulated data sets, trace-plots for the number of blocks in each case are shown in Figure 2.6.

2.6. Real data

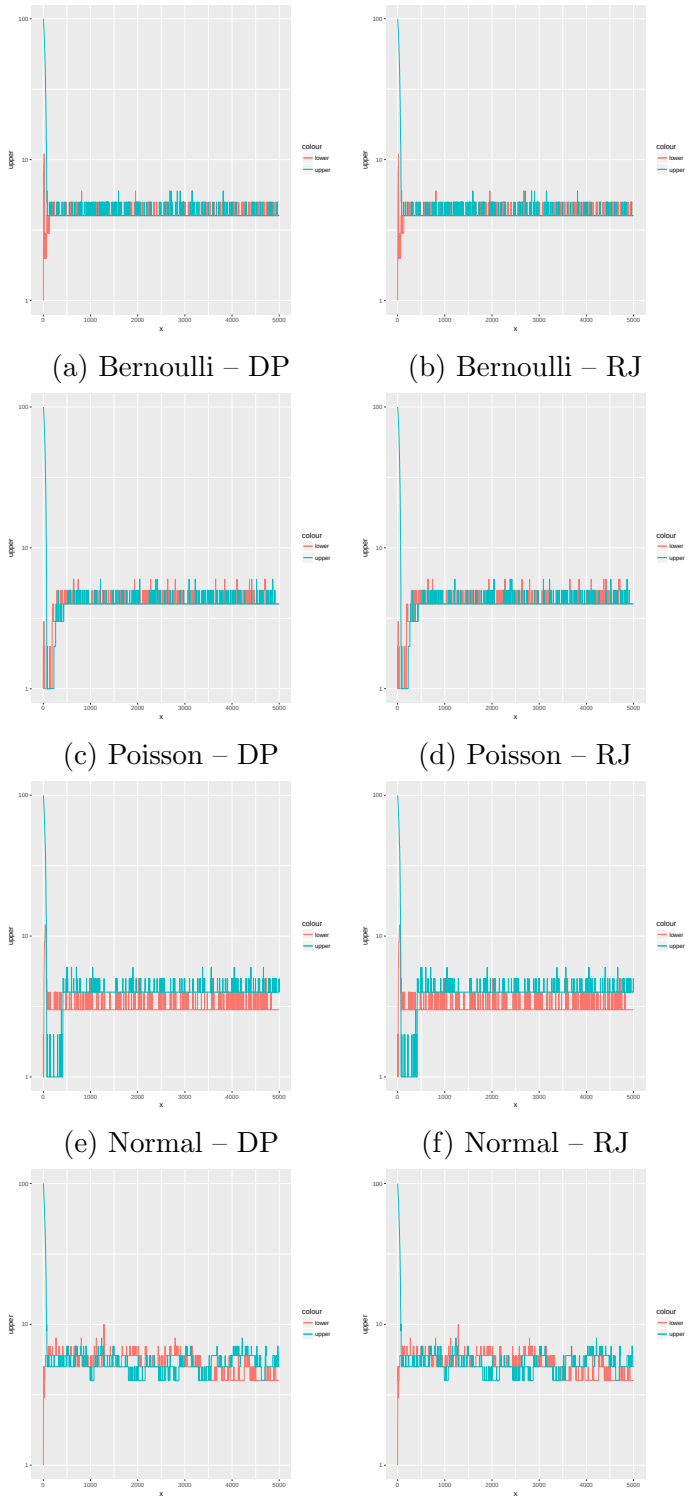
The DP and RJMCMC samplers are demonstrated on real networks. These include a network of brain connectivity with binary edge-states in Section 2.6.1 and a network of emails with edge-state consisting of counts is analysed in Section 2.6.2. In both cases the edge-states are directed. Finally, in Section 2.6.3 a network with symmetric real-valued edge-states is considered.

2.6.1. Macaque

The first data set analysed concerns the brain of a macaque monkey (Négyessy et al., 2006b). Regions of the cortex were deemed connected, or not, during a sensory task. In total, 45 regions of the brain were analysed as a network. Figure 2.7a shows the data set as an adjacency matrix.

In this paper a block model is proposed to partition the regions of the brain. This model assigns regions of the brain to the same block if their neural activity is similar. Since the data only provides binary edge-states, a Bernoulli SBM is applied. A Beta(0.5,0.5) reference prior was placed on the edge probability parameters θ_k. The Dirichlet process sampler and split-merge algorithm were run for 5,000 iterations to provide samples from the posterior distribution of both block membership and parameter values. In the DP sampler a CRP(5) prior was used, whilst the RJMCMC sampler used a DMA(1, 6) prior for block memberships. In both cases the expected number of blocks is five.

Figure 2.7 displays posterior summaries of the macaque network using the Dirichlet process sampler whilst Figure 2.8 displays summaries for the split merge



(a) Bernoulli – DP (b) Bernoulli – RJ

(c) Poisson – DP (d) Poisson – RJ

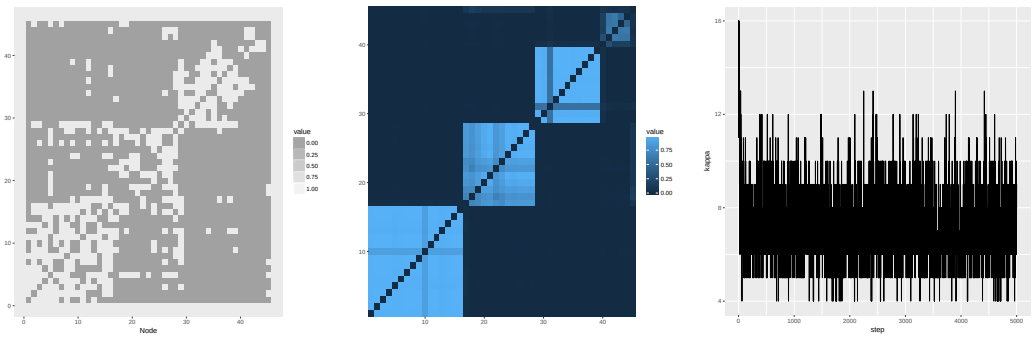
(e) Normal – DP (f) Normal – RJ

(g) Negative binomial – DP (h) Negative binomial – RJ

Figure 2.6.: Trace plots for number of blocks in example networks. Two chains are simulated in each case: The "lower chain" with all nodes initially in one block (orange line) and the "upper chain" with all nodes initially assigned to different blocks (blue line).

algorithm. Notice that the joint posterior probability of two nodes being members of the same block (presented in Equation (2.23)) for the Dirichlet process sampler is more confident than the Split merge algorithm. This is shown in Figure 2.7b, where four clear blocks are visible, whereas Figure 2.8b shows three blocks, however, the block in the top right is more uncertain, and could be interpreted as two blocks with some probability of merging as one. This can be seen in the adjacency matrix in Figure 2.8a, where the nodes in the top right could form one or two groups with a high density of connections. The number of blocks in the sampler state is plotted against sampler step in Figure 2.8c, this shows that the sampler quickly splits blocks until between 4 and 7 blocks exist (for the split-merge algorithm) and between 4 and 12 blocks (for the Dirichlet process sampler).

Parameter estimates for both methodologies are shown in Table 2.5 together with a 95% highest posterior density interval. Note that parameter θ_4 under the RJMCMC sampler is more uncertain, this is due to the fact that the sampler is unsure whether block four exists, or if it should be merged with block 3.

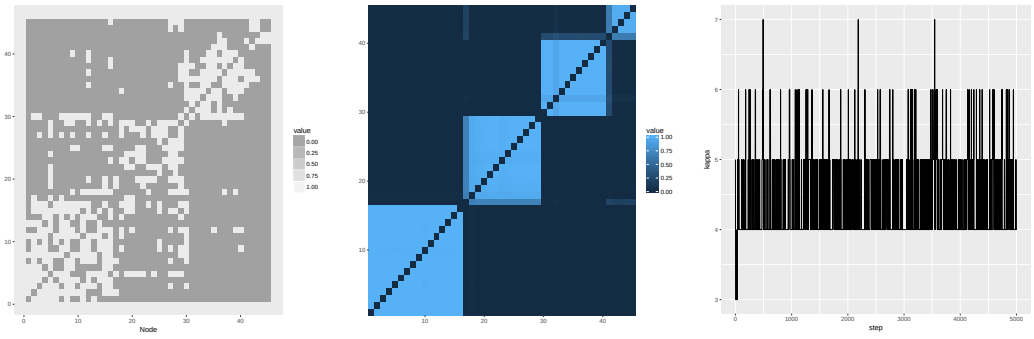


(a) Edge-states sorted by most likely block assignments – DP

(b) Posterior joint block membership – DP

(c) Trace plot for κ – DP

Figure 2.7.: Posterior summaries for block membership in Macaque brain network using Dirichlet process sampler.



(a) Edge-states sorted by most likely block assignments – RJ

(b) Posterior joint block membership – RJ

(c) Trace plot for κ – RJ

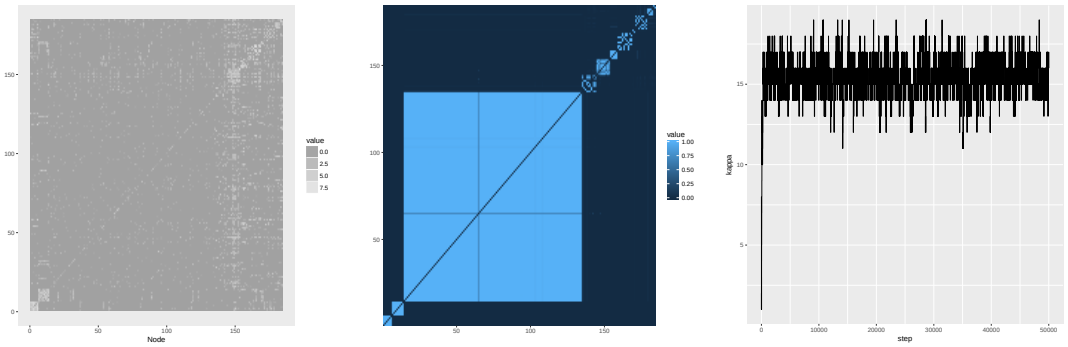
Figure 2.8.: Posterior summaries for block membership in Macaque brain network using Dirichlet process sampler.

Table 2.5.: Parameter estimates and 95% highest posterior density interval for Macaque network.

Parameter	DP	RJ
θ_0	0.099 (0.078, 0.117)	0.091 (0.076, 0.107)
θ_1	0.704 (0.643, 0.764)	0.702 (0.649, 0.765)
θ_2	0.533 (0.394, 0.665)	0.481 (0.379, 0.580)
θ_3	0.766 (0.643, 0.878)	0.719 (0.612, 0.829)
θ_4	0.634 (0.148, 1.000)	0.576 (0.239, 0.996)

2.6.2. Enron

The Enron data set is the second network to be analysed in this section. The Enron corporation was declared bankrupt in 2001 and later multiple employees were found guilty of accounting fraud. As a result of the trial, a corpus of emails leading up to the closure of the company were released as a public data set (Klimt and Yang, 2004). In this paper the aggregate counts of emails between any two employees are analysed over the entire period available. Note that these are directed, and contain self-loops (since some emails are sent to mailing lists, to which the sender belongs). Therefore, the edge-states are integer based. A Poisson model could be used here with a Gamma prior, however, on a first analysis, the mean number of emails sent by any one employee is 3.7, whilst the variance is 4753. Therefore, for more flexibility, a negative binomial distribution is assumed for the edge-state distribution. As for the priors, r is given a Gamma prior and p a Beta prior. The split-merge algorithm of Section 2.4 was applied with 50,000 steps with both a negative binomial and Poisson edge-state model. In both cases a DMA(1,6) joint prior is placed on $\kappa, \boldsymbol{z}$ whilst a Gamma(1/2, 0) reference prior is placed on the rate for the Poisson model, whereas a Gamma(1/2, 0) and a Beta(0.5, 0.5) are placed on r and p respectively for the negative binomial model. The resulting modal block structure under the Poisson model is shown in Figure 2.9b. The adjacency matrix is plotted with the same ordering of nodes in Figure 2.9a (on a log scale). The corresponding results for the negative binomial are shown in Figure 2.10. The negative binomial model shows more uncertainty in the block membership of nodes (Figure 2.10b) compared to the Poisson model (Figure 2.9b). This is due to the lack of flexibility in the Poisson model, unlike for the Poisson. In both models, the large group has a low incidence of emails sent. Furthermore this block is smaller in the negative binomial model than the Poisson. This demonstrates the flexibility in the negative binomial approach, which can detect more structure in the network. The parameter values for each model are given in Table 2.6 together with a 95% highest posterior density interval.

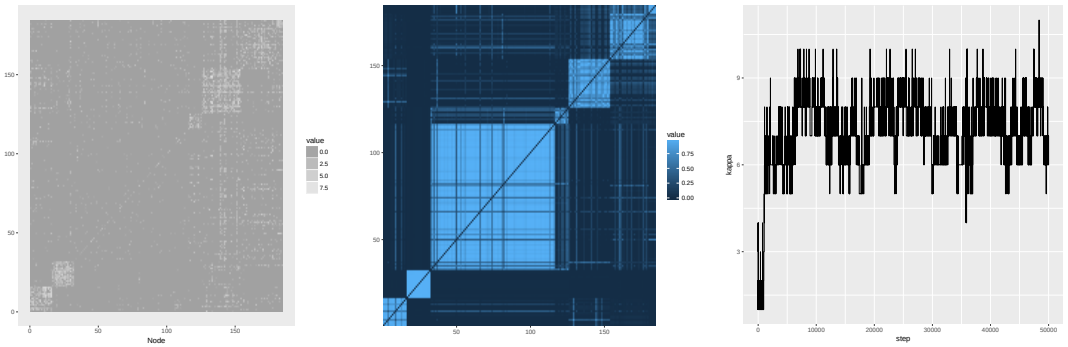


(a) log(edge-states) permuted (b) SM joint block membership posterior (c) SM trace plot for κ

Figure 2.9.: Posterior summaries for block membership in Enron network with Poisson edge-state model.

Table 2.6.: Parameter estimates and 95% highest posterior density interval for Enron data set under both models.

Block	Poisson(λ)	NBin(r)	NBin(p)
0	2.948 (2.832, 3.011)	0.016 (0.014, 0.018)	0.012 (0.009, 0.014)
1	348.494 (343.838, 356.793)	0.189 (0.116, 0.519)	0.002 (0.001, 0.009)
2	106.433 (103.874, 109.264)	0.313 (0.268, 0.374)	0.007 (0.005, 0.010)
3	0.368 (0.354, 0.384)	0.009 (0.007, 0.012)	0.035 (0.022, 0.052)
4	116.907 (0.562, 120.995)	0.059 (0, 0.267)	0.006 (0, 0.010)
5	202.968 (70.603, 265.529)	0.142 (0.076, 0.229)	0.003 (0, 0.003)
6	141.434 (94.038, 157.022)	0.077 (0.056, 0.158)	0.004 (0, 0.007)
7	671.401 (661.435, 682.603)	0.130 (0, 3737.000)	0.006 (0, 0.978)
8	88.199 (78.024, 182.127)	0.177 (0, 19001.158)	0.004 (0n, 0.996)
9	86.587 (0.002, 638.540)		
10	86.948 (0.015, 725.435)		
11	83.801 (0.003, 453.429)		
12	84.072 (0.299, 221.739)		
13	85.986 (0.644, 232.241)		
14	85.503 (0.012, 279.680)		



(a) log(edge-states) permuted (b) SM joint block membership posterior (c) SM trace plot for κ

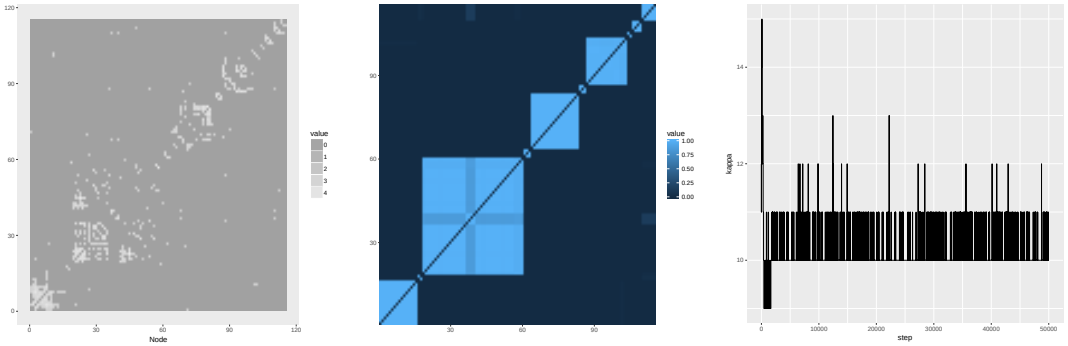
Figure 2.10.: Posterior summaries for block membership in Enron network with negative binomial edge-state model.

2.6.3. Stack Overflow

In 2017 the company Stack Overflow at stackoverflow.com released data from their developer stories. Developer stories are online curriculum vitae for developers on the stackoverflow.com platform. When creating a developer story, the developer may add "tags" to highlight which technologies they use. For example, a developer using tags such as "html", "javascript" and "css" would likely be a website designer. Stack Overflow released a network based on the frequency of co-occurrence of these tags in developer stories. The network was constructed with each tag represented by a node, and the edge-states represent the co-occurrence of tags. This network gives a positive value to edge-states from the set of non-negative real numbers. The network was analysed with the split-merge algorithm of Section 2.4. Since the network is formed on co-occurrence, prior distributions reflecting an assortative block structure were chosen. The edge-states were modelled using a Gaussian distribution with unknown mean and variance. The prior for $\boldsymbol{\theta}_0 = (\mu_0, \sigma_0)$ was $\mu_0 \sim \text{Normal}(0, 1)\,, \sigma_0 \sim \text{Gamma}(10, 10)$ whilst for $\boldsymbol{\theta}_k = (\mu_k, \sigma_k)$ was $\mu_k \sim \text{Normal}(20, 3)\,, \sigma_k \sim \text{Gamma}(30, 10)$. After 50,000 iterations, with 25,000 discarded as burn-in, the estimate of the joint posterior block membership was obtained as shown in Figure 2.11b. The block structure found indicates that nine blocks exist. The model block structure is given in Table 2.8. This shows the partition of technologies based on the SBM with Normally distributed edge states. The table has been annotated by the authors to describe the technologies in each block. Considering the partition induced by the block structure in the posterior, developers on Stackoverflow.com seem to specialise in one area, with a low frequency of co-occurrence for tags in different blocks, as indicated by θ_0. The mean posterior parameter values are given in Table 2.7 together with a 95% credible interval with the corresponding block labels from Table 2.8. This modal block structure is used to order the nodes in the plots in Figure 2.11. In Figure 2.11b block one is in the lower left corner and block 10 in the upper right.

The adjacency matrix is displayed in Figure 2.11a, with row and columns per-

muted to match Figure 2.11b. Qualitatively, the blocks found do indeed have an assortative structure. Notice in Figure 2.11b that there is uncertainty on if some nodes in the 2nd block should be merged with the 7th block (counting from the bottom left). These tags are "c++", "embedded", "qt" and "c". These are technologies with a broad use case so fit equally well with the "Web Apps" block (2) or "Scientific" block (7). The trace plot for the number of blocks is given in Figure 2.11c. This shows the algorithm exploring different block structures as the algorithm progresses, before settling on a mean of ten blocks (with some exploration to 11 or 12 blocks). The block structure found partitions the technology tags into groups such that tags in the same group are likely to appear together in the same developer story on stackoverflow.com, and as such, technologies in the same block should be similar.



(a) log(edge-states) permuted (b) SM joint block membership posterior (c) SM trace plot for κ

Figure 2.11.: Posterior summaries for block membership in Stack Overflow network.

2.7. Concluding remarks

This paper has considered the stochastic block model with arbitrary edge-state distributions and an unknown number of blocks. Two Bayesian inference algorithms were proposed: a Dirichlet process sampling procedure in Section 2.3 and a split-merge algorithm facilitated by a reversible jump Markov chain Monte Carlo sampler in Section 2.4. Unlike previous Bayesian treatments of the stochastic block

Table 2.7.: Parameter estimates and 95% highest posterior density values for the Stack Overflow network.

Block	Normal(μ)	Normal(σ)	Description
0	0.084 (0.049, 0.118)	1.396 (1.318, 1.453)	
1	13.482 (11.314, 15.748)	13.124 (12.051, 14.199)	.NET
2	26.153 (21.407, 30.642)	3.282 (2.226, 4.355)	Tests
3	4.841 (3.713, 6.115)	12.844 (12.063, 13.771)	Web apps
4	21.153 (16.703, 25.326)	5.469 (4.581, 6.541)	Angular
5	8.005 (6.040, 9.649)	12.795 (11.855, 13.746)	Developer tools
6	34.780 (29.414, 40.067)	5.212 (3.927, 6.557)	Excel
7	9.157 (7.071, 11.244)	13.289 (12.245, 14.371)	Operating system
8	20.752 (16.720, 25.611)	2.775 (1.818, 3.827)	Regular expressions
9	22.308 (18.223, 26.780)	7.729 (6.583, 8.816)	Big data
10	14.631 (9.254, 19.359)	8.893 (7.045, 11.588)	Scientific

Table 2.8.: Model block structure for the Stack Overflow network.

Block	Modal block members	Description
1	azure sql-server asp.net entity-framework wpf linq wcf c# asp.net-web-api .net sql mvc vb.net xamarin unity3d visual-studio	.NET
2	testing selenium	Tests
3	tdd codeigniter jquery mysql css php javascript json angularjs ionic-framework reactjs mongodb sass twitter-bootstrap express node.js html5 nginx c++ embedded qt c laravel ajax wordpress photoshop html bootstrap less postgresql redis redux twitter-bootstrap-3 xml vue.js apache elasticsearch react-native ruby-on-rails ruby agile drupal	Web apps
4	typescript angular2 angular	Angular
5	cloud devops docker android-studio java android rest amazon-web-services jenkins go maven jsp spring-boot web-services spring-mvc java-ee spring hibernate eclipse api	Developer tools
6	vba excel-vba excel	Excel
7	ios linux shell git github bash swift osx objective-c iphone xcode unix ubuntu windows plsql oracle powershell	Operating system
8	regex perl	Regular expressions
9	scala hadoop apache-spark haskell	Big data
10	python flask django R machine-learning matlab	Scientific

model with an unknown number of blocks (Mørup and Schmidt, 2012, 2013; McDaid et al., 2013), the proposed algorithms handle edge-state distributions without

conjugate priors. This allows for more flexible modelling of network data. For example, the Enron email network in Section 2.5 consists of edge-states representing count data. A negative binomial model (with both parameters unknown) was fit to the edge-states, allowing a higher variance than a Poisson model. In comparison to the Poisson model when applied to the Enron data set, the negative binomial explored the parameter space better by visiting posterior states with more blocks than the Poisson edge-state model.

In Section 2.5.1 both algorithms were applied to example networks. As discussed in Section 2.1, Gibbs samplers can get stuck in local modes of the posterior. This is also true of the DP sampler, which is outperformed by the split-merge algorithm in terms of identifying the true number of blocks in the examples.

The algorithms presented here are quite general and can applied to the stochastic block model with any edge-state distributions from which samples can be taken and densities evaluated. This can easily include covariate information in either the edge-state distribution, G, or the block membership distribution, F.

The model as presented here assumes that all edge-states are observed. This could be relaxed by including a sparsity parameter as in Matias and Miele (2017) which could be inferred within the SBM framework (since it treats edges as a mixture of a density and a Dirac mass at zero). Alternatively, for truly missing data, a data augmentation scheme could be applied within the proposed samplers to infer the missing edge-states together with the block memberships and model parameters.

3. Autoregressive stochastic block model with changes in block membership

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.

3.1. 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: Barabási-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(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); Bardwell et al. (2016), 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 (2017)).

Within a network context, changing behaviour can arise in many different scenarios. This article focuses on movement of individuals from one community to another 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 3.6, 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 3.6 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, 2017). 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 3.2 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 3.3 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 3.4. In Sections 3.5 and 3.6 the RJMCMC sampling scheme is demonstrated on simulated data sets and a data set involving monitoring social behaviour in mice (Lopes et al., 2016b), respectively. Finally, in Section 3.7 we make some concluding remarks concerning directions for future research in this area.

3.2. The autoregressive stochastic block model

3.2.1. 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, 1998), 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 Exp(λ) 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, $M_i \sim \text{Pois}(\lambda)$ with the times of the changes $\boldsymbol{\tau}_i = \left(\tau_i^1, \ldots, \tau_i^{M_i}\right)$ being ordered and uniformly distributed on $[t_0, t_T]$. The new community level, $C_i\left(\tau_i^d\right)$, is drawn uniformly at random from

the set $\{k \neq c_i(\tau_i^{d-1}) : k = 1, \ldots, K\}$, 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 $\boldsymbol{c}_i(\boldsymbol{t})$.

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 (3.1).

$$C_{ij}(t) = \begin{cases} C_i(t) & \text{if } C_i(t) = C_j(t)\,, \\ 0 & \text{if } C_i(t) \neq C_j(t)\,, \end{cases} \qquad (3.1)$$
$$C_{ij}(t) \in \{0, 1, \ldots, K\}.$$

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

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

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.

3.2.2. 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 $\boldsymbol{t} = (t_0, t_1, \ldots, t_T)$ 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 $\boldsymbol{e}(\boldsymbol{t}) = \left\{e_{ij}^s | 1 \leq i < j \leq N, s = 0, 1, \ldots, T\right\}$ denote the set of all network snapshot data. Let $\boldsymbol{c}_i(\boldsymbol{t}) = \{c_i^s | s = 0, 1, \ldots, T\}$, the community membership of node i at every observation time, with $\boldsymbol{c}(\boldsymbol{t}) = \{\boldsymbol{c}_i(\boldsymbol{t}) \,| i = 1, 2, \ldots, N\}$,

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 (3.2).

$$\begin{aligned}\pi(\boldsymbol{\theta}, \boldsymbol{c}(\boldsymbol{t}) \,|\boldsymbol{e}(\boldsymbol{t})) &\propto \pi(\boldsymbol{e}(\boldsymbol{t}) \,, \boldsymbol{c}(\boldsymbol{t}) \,|\boldsymbol{\theta}) \, \pi(\boldsymbol{\theta}) \\ &= \pi(\boldsymbol{e}(\boldsymbol{t}) \,|\boldsymbol{c}(\boldsymbol{t}) \,, \boldsymbol{\theta}) \, \pi(\boldsymbol{c}(\boldsymbol{t}) \,|\boldsymbol{\theta}, \boldsymbol{c}(t_0)) \, \pi(\boldsymbol{\theta}, \boldsymbol{c}(t_0)) \,,\end{aligned} \tag{3.2}$$

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

We now provide equations for each term in Equation (3.2). Firstly, in Equation (3.3), the likelihood of the observed edge sequence, given the latent community memberships and model parameters, is computed.

$$\pi(\boldsymbol{e}(\boldsymbol{t}) \,|\boldsymbol{c}(\boldsymbol{t}) \,, \boldsymbol{\theta}) = \prod_{\substack{s,i \\ j \neq i}} \pi\big(e_{ij}^s | e_{ij}^{s-1}, c_i^{s-1}, c_i^s, c_j^{s-1}, c_j^s\big) \tag{3.3}$$

The computation of each factor in Equation (3.3) 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, $\boldsymbol{\tau}_i = \big(\tau_i^1, \ldots, \tau_i^{M_i}\big)$, and the community membership of the nodes at the changepoints, $\boldsymbol{c}_i(\boldsymbol{\tau}_i)$ with $\boldsymbol{c}(\boldsymbol{\tau}) = (\boldsymbol{c}_1(\boldsymbol{\tau}_1) \,, \ldots, \boldsymbol{c}_N(\boldsymbol{\tau}_N))$. Note that $\boldsymbol{c}_{ij}(\boldsymbol{\tau}_{ij})$ is a deterministic function of $\boldsymbol{c}_i(\boldsymbol{\tau}_i)$ and $\boldsymbol{c}_j(\boldsymbol{\tau}_j)$, where $\boldsymbol{\tau}_{ij} = \left(\tau_{ij}^1, \ldots, \tau_{ij}^{M_{ij}}\right)$ 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

$$\mathbb{P}\big[e^s = 1 | e^{s-1}, c(t)\big] = \pi_k + (e^{s-1} - \pi_k) \exp(-\rho_k \Delta_s) \,, \tag{3.4}$$

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 $\mathbb{P}\big[e_{ij}^s = 1 | e_{ij}^{s-1}, c_{ij}(t)\big]$ 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

$$\begin{aligned}
&\mathbb{P}\big[e^s = 1 | e^{s-1}, c(t)\big] \\
&= \sum_{l=0}^{1} \mathbb{P}[e^s = 1 | e(\tau) = l] \, \mathbb{P}\big[e(\tau) = l | e^{s-1}\big] \\
&= \sum_{l=0}^{1} \{\pi_k + (l - \pi_k) \exp(-\rho_k (t_s - \tau))\} \, \mathbb{P}\big[e(\tau) = l | e^{s-1}\big] ,
\end{aligned} \tag{3.5}$$

where again, we drop the subscript ij for brevity.

Whilst it is possible to compute $\pi(\boldsymbol{e}(\boldsymbol{t}) | \boldsymbol{c}(\boldsymbol{t}), \boldsymbol{c}(\boldsymbol{\tau}), \boldsymbol{\theta})$ from (3.5), it is far simpler to augment the data with $\boldsymbol{e}(\boldsymbol{\tau}) = \big\{e_{ij}\big(\tau_{ij}^d\big); 1 \leq i, j \leq N, d = 1, 2, \ldots, M_{ij}\big\}$. 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, $\pi(\boldsymbol{e}(\boldsymbol{\sigma}) | \boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}, \boldsymbol{\theta})$ becomes

$$\prod_{i \neq j} \prod_{d=0}^{T_{ij}-1} \mathbb{P}\big[e_{ij}\big(\sigma_{ij}^{d+1}\big) | e_{ij}\big(\sigma_{ij}^d\big), c_{ij}\big(\sigma_{ij}^d\big)\big] \tag{3.6}$$

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

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

The computation of $\pi(\boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{\tau} | \boldsymbol{\theta}, c(t_0))$ is straightforward. Firstly, $\boldsymbol{\sigma}_i$ is determin-

istic given $\boldsymbol{\tau}_i$, so

$$\begin{aligned}
& \pi(\boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}|\boldsymbol{\theta}, c(t_0)) \\
= & \prod_{i=1}^{N} \pi(\boldsymbol{c}_i(\boldsymbol{\tau}_i)|\boldsymbol{\tau}_i, \boldsymbol{c}(t_0))\, \pi(\boldsymbol{\tau}_i|\lambda) \\
= & \prod_{i=1}^{N} \pi(\boldsymbol{c}_i(\boldsymbol{\tau}_i)|\boldsymbol{\tau}_i, \boldsymbol{c}(t_0))\, \pi(\boldsymbol{\tau}_i|M_i)\, \pi(M_i|\lambda) \\
= & \prod_{i=1}^{N} \left(\frac{1}{k-1}\right)^{M_i} \times \frac{M_i!}{(t_T - t_0)^{M_i}} \\
& \times \frac{\{\lambda(t_T - t_0)\}^{M_i}}{M_i!} \exp(-\lambda(t_T - t_0)) \\
= & \left(\frac{1}{k-1}\right)^{M} \lambda^{M} \exp(-\lambda N(t_T - t_0)), \qquad (3.7)
\end{aligned}$$

where $M = \sum_{i=1}^{N} M_i$ is the total number of changepoints. The three components on the right-hand side of (3.7) 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 (3.6) and (3.7), 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

$$\begin{aligned}
\pi(\boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{e}(\boldsymbol{\tau}), \boldsymbol{\tau}, \boldsymbol{\theta}|\boldsymbol{e}(\boldsymbol{t})) \propto \pi(\boldsymbol{e}(\boldsymbol{\sigma})|\boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}, \boldsymbol{\theta}) \\
\times \pi(\boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{\tau}|\boldsymbol{c}(t_0), \boldsymbol{\theta}) \times \pi(\boldsymbol{c}(t_0), \boldsymbol{\theta}).
\end{aligned} \qquad (3.8)$$

3.2.3. 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, 2017). 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 $\boldsymbol{E}(t) = \boldsymbol{E}(s)$ 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 (3.4), 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

$$\rho_* = \frac{1}{K^2}\sum_{k=1}^{K}\rho_k + \frac{K-1}{K}\rho_0, \tag{3.9}$$

and

$$\pi_* = \frac{1}{\rho_*}\left\{\frac{1}{K^2}\sum_{k=1}^{K}\rho_k\pi_k + \frac{K-1}{K}\rho_0\pi_0\right\}. \tag{3.10}$$

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 (2017) for the discrete time SBM models of Xu and Hero (2014) and Matias and Miele (2017) 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 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.

3.3. Reversible jump MCMC

3.3.1. 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 (3.8) and data augmentation of the values $(\boldsymbol{\tau}, \boldsymbol{c}(\boldsymbol{\tau}), \boldsymbol{e}(\boldsymbol{\tau}))$. 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 (3.6) and (3.7). 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 3.5. 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 (3.7) that, $\lambda|\boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\tau}, \boldsymbol{c}(\boldsymbol{\sigma}), \boldsymbol{e}(\boldsymbol{\sigma})$ 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 $\log(\rho_k^*) \sim \text{Normal}\big(\log(\rho_k), \sigma_\rho^2\big)$. 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

$$q(\pi_k^*|\pi_k) = \frac{\phi(\text{logit}(\pi_k^*)\,|\text{logit}(\pi_k), \sigma_\pi^2)}{\pi_k^*(1-\pi_k^*)}, \tag{3.11}$$

Algorithm 3.5 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 $\rho_k^{(h)}$.
end for
if There are no changes in the current sampler state **then**
Propose inserting a change
elseDraw 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.
elseX=2
Propose deleting a change from the current state, removing and adding affected augmented edge states as required.
end if
end if
Given that $M > 0$, propose moving each changepoint into an adjacent observation interval and using a Gaussian random walk proposal.
Resample the augmented edges.
end for
Discard samples $1, \ldots,$burn-in.

and

$$q(\rho_k^*|\rho_k) = \frac{\phi(\log(\rho_k^*)\,|\log(\rho_k)\,, \sigma_\rho^2)}{\rho_k}, \tag{3.12}$$

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.

3.3.2. Updating change points and augmented edge states

The trans-dimensional sampler for updating $\boldsymbol{\tau}$, and consequently, $(\boldsymbol{c}(\boldsymbol{\tau})\,, \boldsymbol{e}(\boldsymbol{\tau}))$ 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 3.1 for an example of possible insert moves.

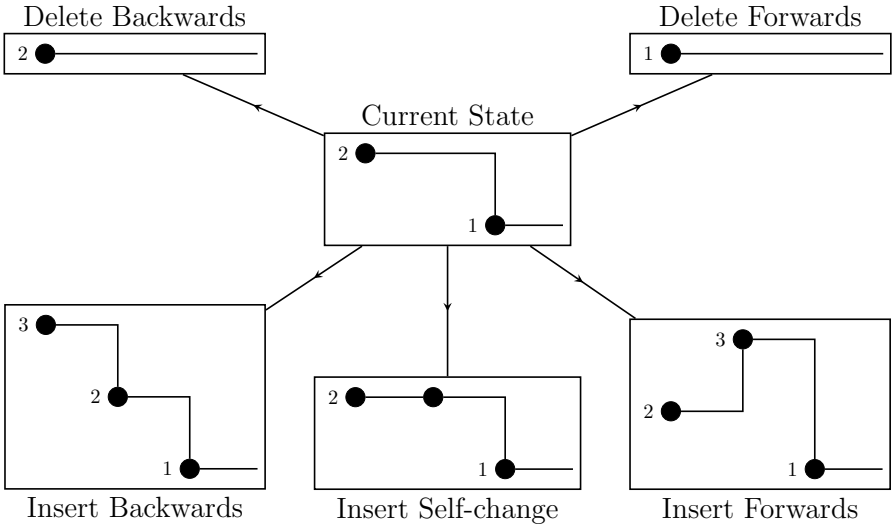


Figure 3.1.: Possible moves to insert or delete a changepoint for a node which currently has one change. After choosing to insert or delete, a model is proposed proportional to the likelihood.

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 $\boldsymbol{E}_i(\tau^*) = (E_{i1}(\tau^*), E_{i2}(\tau^*), \ldots, E_{iN}(\tau^*))$. 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,

$$\mathbb{P}[e_{ij}(\tau^*) = 1 | e_{ij}(\sigma^* \wedge t^*), c_{ij}(\sigma^*) = \kappa, \pi_\kappa, \rho_\kappa], \tag{3.13}$$

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

$$\frac{\mathbb{P}\left[\mathcal{A}^{k_1,k_2} | \boldsymbol{\theta}, \tau^*\right]}{\sum_{l_1,l_2} \mathbb{P}[\mathcal{A}^{l_1,l_2} | \boldsymbol{\theta}, \tau^*]}. \tag{3.14}$$

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

$$\frac{\mathbb{P}[M+1|M]}{N(t_T - t_0)} \cdot \mathbb{P}[\mathcal{A}^* | \boldsymbol{\theta}, \tau^*] \cdot \frac{\mathbb{P}\left[\mathcal{A}^{k_1,k_2} | \boldsymbol{\theta}, \tau^*\right]}{\sum_{l_1,l_2} \mathbb{P}[\mathcal{A}^{l_1,l_2} | \boldsymbol{\theta}, \tau^*]}. \tag{3.15}$$

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 (3.13) 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

$$\frac{\mathbb{P}\left[\mathcal{B}^{k_l}|\boldsymbol{\theta}\right]}{\mathbb{P}[\mathcal{B}^{k_1}|\boldsymbol{\theta}] + \mathbb{P}[\mathcal{B}^{k_2}|\boldsymbol{\theta}]}. \tag{3.16}$$

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

$$\frac{\mathbb{P}[M-1|M]}{M} \cdot \mathbb{P}[\mathcal{B}^*|\boldsymbol{\theta}] \cdot \frac{\mathbb{P}\left[\mathcal{B}^{k_l}|\boldsymbol{\theta}\right]}{\mathbb{P}[\mathcal{B}^{k_1}|\boldsymbol{\theta}] + \mathbb{P}[\mathcal{B}^{k_2}|\boldsymbol{\theta}]}. \tag{3.17}$$

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

$$\begin{aligned}
&\frac{\pi(\boldsymbol{e}(\boldsymbol{\sigma'}), \boldsymbol{c}(\boldsymbol{\tau'}), \boldsymbol{\tau'}, \boldsymbol{\theta}|\boldsymbol{e}(\boldsymbol{t}))}{\pi(\boldsymbol{e}(\boldsymbol{\sigma}), \boldsymbol{c}(\boldsymbol{\tau}), \boldsymbol{\tau}, \boldsymbol{\theta}|\boldsymbol{e}(\boldsymbol{t}))} \\
&\times \frac{\mathbb{P}[M|M+1]\, N(t_T - t_0)}{\mathbb{P}[M+1|M]\,(M+1)} \times \frac{\mathbb{P}[\mathcal{B}^*|\boldsymbol{\theta}]}{\mathbb{P}[\mathcal{A}^*|\boldsymbol{\theta}, \tau^*]} \\
&\times \frac{\mathbb{P}\left[\mathcal{B}^{k_l}|\boldsymbol{\theta}\right] \sum_{l_1,l_2} \mathbb{P}\left[\mathcal{A}^{l_1,l_2}|\boldsymbol{\theta}, \tau^*\right]}{\mathbb{P}[\mathcal{A}^{k_1,k_2}|\boldsymbol{\theta}, \tau^*]\, \mathbb{P}[\mathcal{B}^{k_1}|\boldsymbol{\theta}] + \mathbb{P}[\mathcal{B}^{k_2}|\boldsymbol{\theta}]}
\end{aligned} \tag{3.18}$$

where $\boldsymbol{\sigma'} = \boldsymbol{\sigma} \cup \tau^*$ and $\tau' = \tau \cup \tau^*$. The acceptance probability for deleting a changepoint is the reciprocal of (3.18).

The two procedures for moving a changepoint are straightforward. Firstly, each change point 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 state $A \in \mathcal{A}^*$ is resampled proportional to the relative likelihood using Equation (3.3) in the proposal distribution,

$$\mathbb{P}[A = 1] = \frac{\pi(A = 1, \boldsymbol{e}(\boldsymbol{\sigma}) \,|\theta)}{\pi(A = 0, \boldsymbol{e}(\boldsymbol{\sigma}) \,|\theta) + \pi(A = 1, \boldsymbol{e}(\boldsymbol{\sigma}) \,|\theta)}.$$

In the case that a change point 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.

3.4. 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 $\boldsymbol{e}(\boldsymbol{t})$ 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 fit to each M^d to yield an initial assignment of nodes to communities labelled 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. A comparison between the different initialisation procedures can be found in Appendix B.

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 (3.19).

$$d(i, j) = \gamma d_1(i, j) + (1 - \gamma) d_2(i, j) \tag{3.19}$$

3. Autoregressive stochastic block model with changes in block membership

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 (3.19) 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 (3.20).

$$\begin{aligned} d_{11}(i,j) &= \gamma_{11} d_1(i,j) + (1-\gamma_{11}) d_2(i,j) \\ d_{10}(i,j) &= \gamma_{10} d_1(i,j) + (1-\gamma_{10})(1-d_2(i,j)) \\ d_{01}(i,j) &= \gamma_{01}(1-d_1(i,j)) + (1-\gamma_{01}) d_2(i,j) \\ d_{00}(i,j) &= \gamma_{00}(1-d_1(i,j)) + (1-\gamma_{00})(1-d_2(i,j)) \end{aligned} \tag{3.20}$$

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 (3.20). 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.

3.5. 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 3.1. For equal

community sizes, N/C nodes were placed in each community. The sizes of communities for other simulations are given in Table 3.2. 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.

Table 3.1.: 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 3.2.: 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

The RJMCMC described in Section 3.3 was applied to each simulated data set for $H = 20,000$ steps. The prior 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 (3.21)) are computed for each of the parameters $\boldsymbol{\pi}$, $\boldsymbol{\rho}$, λ and $\boldsymbol{\tau}$. The

MAPE of an estimate E from true value T is given by:

$$\text{MAPE}(E,T) = \sum_{i=1}^{n} \frac{|E_i - T_i|}{|T_i|} \tag{3.21}$$

Additionally, to assess the estimation of community assignments $\underline{c}(t)$, 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.

3.6. 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 (3.20). 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 3.2 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.

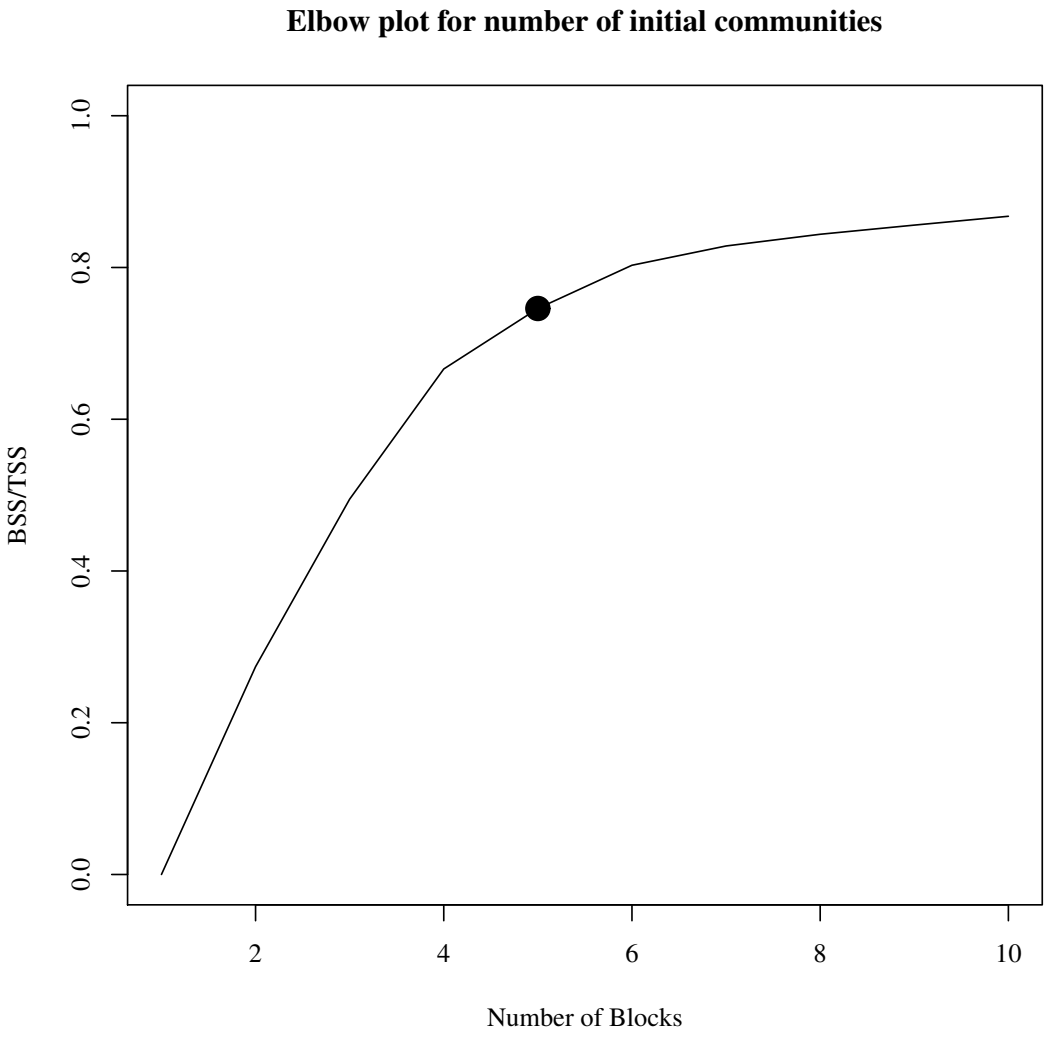


Figure 3.2.: Elbow plot for determining the number of communities with which to initialise the sampler.

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 3.3 with around 50 changes in community membership. Trace-plots are available in Appendix B. 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 3.3 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 (white) in Figure 3.3. 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 Appendix B.

For comparison, the dynamic SBM (dynSBM) of Matias and Miele (2017) 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 3.3 and 3.4. The mean parameter estimates for the dynSBM are given in Table 3.4. For communities $k = 3, 4, 5$, the mean estimates of the parameters β_k and π_k, the probabilities of an edge between two mice in community k in the dynSBM and ARSBM, respectively, are similar. For communities 1 and 2, the parameters differ significantly, 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 3.3.: 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

Table 3.4.: 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

3.7. Concluding remarks

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

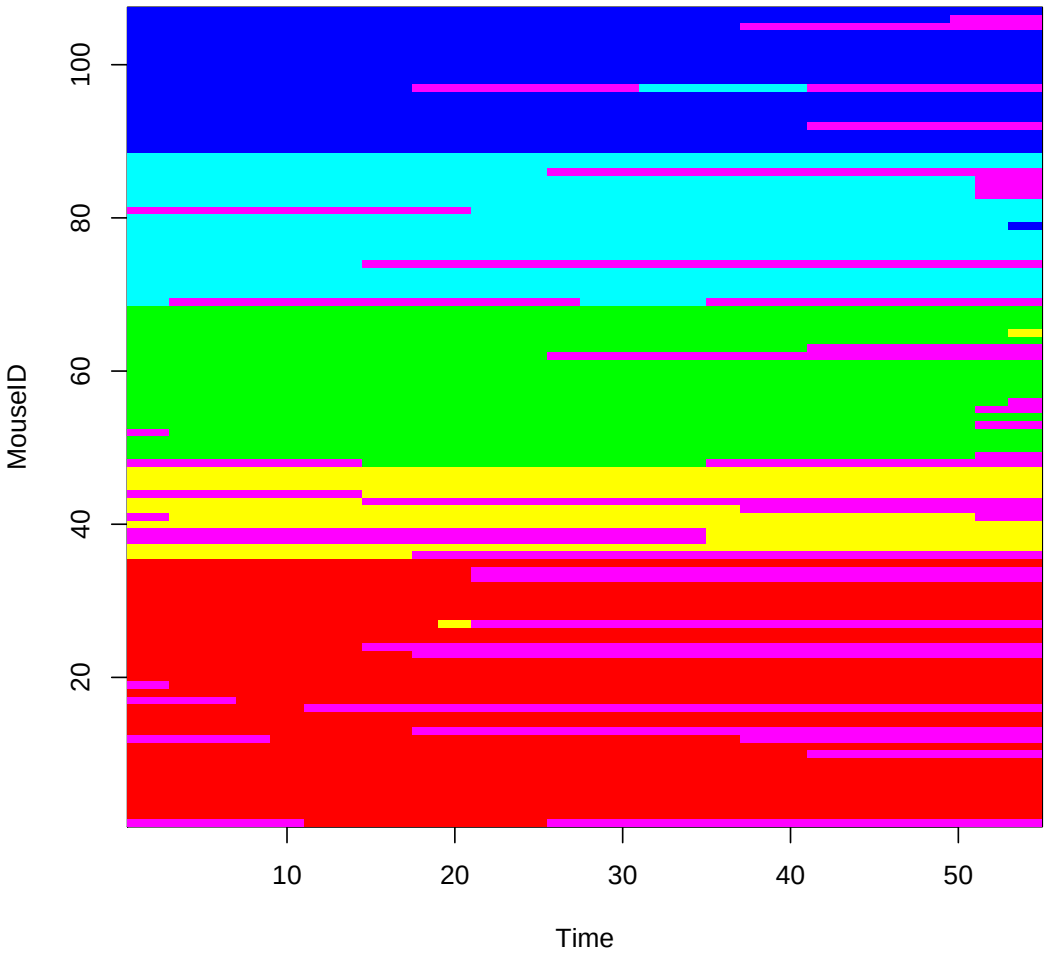


Figure 3.3.: ARSBM: Maximum a posteriori community membership of each mouse through time. Community labels: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple.

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

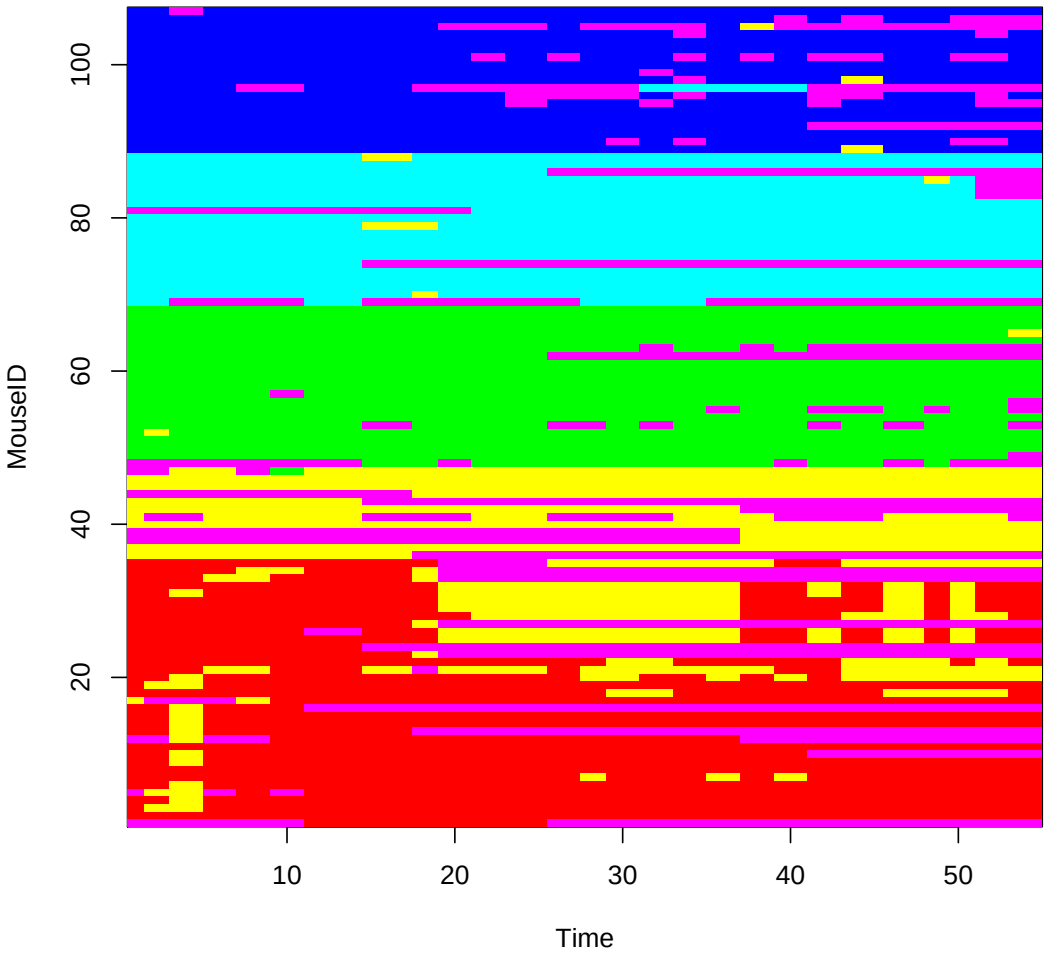


Figure 3.4.: dynSBM: Maximum a posteriori community membership of each mouse through time. Community labels: 1 - red, 2 - yellow, 3 - green, 4 - sky blue, 5 - dark blue, 6 - purple.

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.

4. Online monitoring of block membership in the autoregressive stochastic block model

Abstract

In this paper we consider the problem of online monitoring of community structures in a network. Specifically, we aim to detect when nodes change their community membership. For this, we extend the autoretrogressive stochastic block model (ARSBM) for network evolution first introduced in Ludkin et al. (2017), which describes a process for nodes changing community together with a mechanism for the creation and deletion of edges between nodes. The inference procedure discussed in this paper is based on the Sequential Monte Carlo algorithm of Fearnhead (2002) and Storvik (2002). This allows inference to be made not only on the times when nodes change community membership (changepoints), but also on the static model parameters governing the evolution of edge-states. The algorithm is described in Section 4.3. For efficient implementation of the proposed algorithm, we provide some approximations in Section 4.3.2 together with justification, which greatly improve the computational complexity, making online monitoring a practical possibility. We compare the SMC procedure with the RJMCMC of Ludkin et al. (2017) on both simulated and real data in Section 4.4 and 4.5 to show the computational/accuracy trade off.

4.1. Introduction

In this paper we provide a sequential algorithm for the online monitoring of community structures in a network. A network consists of a set of nodes $\mathcal{V}$ which can interact. We consider binary interactions, which can be modelled as the presence or absence of an edge between a pair of nodes. Furthermore, we consider a dynamic network, where edges can change state over time, signifying the evolution of interactions. These edges are a set of stochastic processes denoted $\mathcal{E}(t)$. For example, in a social network between people, an interaction such as "being in conversation with" can be modelled as an edge. The amount of time the edge remains "on" signifies the length of a given conversation. Denoting $i, j \in \mathcal{V}$ as a pair of distinct nodes, the edge process $E_{ij}(t) \in \mathcal{E}(t)$ models the interaction process between the nodes i and j. We assume a data collection mechanism whereby the state of edges are observed at some given time points, say $t_1, t_2, \ldots, t_T$ called observation times. At each observation time t_s, a *network snapshot* is taken, such that the states of all edges at time t_s are recorded as $E(t_s)$. In practice multiple snapshots will be available and a series of observations of edge states for each pair of nodes is recorded. Each snapshot of the network is available as an *adjacency matrix* $\boldsymbol{E}^s$, with the corresponding series of adjacency matrices denoted by $\boldsymbol{E}^{1:T} = \{E_{ij}(t_s) : s = 1, \ldots, T\}$. In the following, the observations are indexed as $E^s_{ij} = E_{ij}(t_s)$ for $s = 1, \ldots, T$ and $i, j \in \mathcal{V}$.

Various statistical models have been developed for networks where a single snapshot forms the data set. These are referred to as static models (i.e. not dynamic). These include the ERGM (Wasserman and Pattison, 1996; Anderson et al., 1999), latent position model (Hoff et al., 2002; Handcock et al., 2007; Hoff, 2008a,b; Krivitsky et al., 2009), and the stochastic block model (SBM) (Holland et al., 1983; Snijders and Nowicki, 1997). In this paper we consider a dynamic extension to the SBM. The SBM partitions the set of nodes $\mathcal{V}$ into groups or *blocks* such that nodes in the same block have similar interaction behaviours. In the classic SBM, the behaviour used to partition nodes is simply the appearance of edges. In this

way, a block consists of nodes who share many edges, with fewer edges between nodes in different blocks (or vice versa). This is an *assortative* (*disassortative*) block structure.

We apply the stochastic block model to a dynamic model for interactions, hence a partition of the node set is desired such that interaction behaviour is similar for nodes in the same block compared to nodes in different blocks. Specifically, this model treats the appearance of an edge as a binary random variable, but allows autoregressive behaviour. This is a natural assumption in sequential data, allowing the past to be indicative of the future.

Given that the state of edges may change through time, it is natural to consider nodes to change through time as well. This is achieved by allowing nodes to change block membership as time progresses. In this way, a partition of the node set is sought for each time point. Together with the autoregressive edge state model, this forms the autoregressive stochastic block model (ARSBM).

Other authors have considered dynamic extensions to the SBM with changes in block membership. Yang et al. (2011), Xu and Hero (2014) and Matias and Miele (2017) all extend the SBM to a discrete time dynamic version, which allow dynamics of the edge process to depend on the group membership of the nodes. In all of these models, the edge states at each time are assumed independent of past states, given the block memberships. In this way, the dynamics are all modelled via the latent block memberships. The autoregressive stochastic block model relaxes this assumption of independence to allow previous edge states to influence the present edge state via Markovian dynamics.

In this paper we consider an online inference procedure to track the block membership of nodes along time, as well as providing estimates for the posterior of the model parameters. This consists of tracking some sufficient statistics for the parameters, consisting of summaries on the edges and block memberships of nodes. These statistics can be used in conjunction with a data augmentation scheme to provide samples from the posterior for the parameters at any point in time, using

the past data. Since this algorithm only depends on sufficient statistics, the storage requirements for the algorithm do not grow in time without a loss in accuracy.

The remainder of this paper is organised as follows: The autoregressive stochastic block model (ARSBM) is defined in Section 4.2. The SMC algorithm to track the block memberships and provide an estimate of the posterior for the model parameters is described in Section 4.3. In Section 4.4 the algorithm is demonstrated on simulated data to show the effectiveness at tracking block memberships. This includes a comparison to previous inference procedures first considered in Ludkin et al. (2017). The SMC algorithm is applied to a data set of mice in Section 4.5. The paper ends with a discussion and considerations for future work in Section 4.6.

4.2. Model

In this section we introduce the ARSBM and some notation. The model is set in continuous time, allowing easy application in missing data or irregularly spaced sampling regimes. The model splits into a simple hierarchy: firstly the nodes are assigned to blocks together with the process governing their changes in block membership. Secondly, the edge state distribution depends on the block membership of the end-nodes. Note that we assume, as in the SBM, that the edges are independent given the block memberships.

First we describe the block membership model. Each node belongs to one of κ blocks at any given time. In this paper we consider κ is known and fixed. It is assumed that each node starts as a member of block k with some probability ω_k, such that $\sum_{k=1}^{\kappa} \omega_k = 1$. Since we allow the block memberships to change through time, a model is also required to describe this behaviour. To allow block membership to evolve in time, we assume that the block membership of a node follows a continuous time Markov chain (CTMC). Specifically, we assume that the block membership of a node is reassigned at the points in a homogeneous Poisson process with rate λ. When a block membership is reassigned, it is chosen as block k with probability ω_k independently of the current block membership.

Note that this does not exclude the case of reassigning a node back to the same block. Another view of this model is as follows: if a node is currently in block k, it remains in block k for an $\text{Exp}(\lambda(1-\omega_k))$ distributed amount of time. Then, when the node leaves block k, it joins a block $l \neq k$ with probability $\omega_l/(1-\omega_k)$. In this second interpretation, every event is a change in block membership; however the first interpretation allows trivial simulation. Letting $\underline{1}$ be a vector of κ ones and $\mathbb{I}$ be the $\kappa \times \kappa$ identity matrix it is straightforward to show this CTMC has generator matrix:

$$Q = \lambda(\underline{1}\boldsymbol{\omega}^T - \mathbb{I}). \tag{4.1}$$

Let $Z_i(t)$ be the block membership process for node i. At time t_1 the block memberships are drawn from some distribution. In this paper we assume that the initial block memberships are drawn from a Multinomial($\boldsymbol{\omega}$) distribution for simplicity. Then, for all times $t > t_1$, $Z_i(t)$ follows a CTMC(Q).

Given the form of Q, the probability of observing node i in block k at time t, given that $Z_i(s) = l$, is:

$$\mathbb{P}[Z_i(t) = k | Z_i(s) = l, \lambda, \boldsymbol{\omega}] = \omega_k(1 - \exp(-\lambda(t-s))) + \exp(-\lambda(t-s))\,\mathbb{I}[k = l]\,.$$

For brevity we denote $z_{ik}^s = 1$ if $Z_i(t_s) = k$ for $s = 1, \ldots, T$. As such, $\boldsymbol{z}_i^s$ is a one-of-κ indicator vector, with $z_{ik}^s = 1$ if node i is in block k at observation time t_s. Also, let $\boldsymbol{z}^{1:T} = \{\boldsymbol{z}_i^s : i \in \mathcal{V}, s = 1, \ldots, T\}$ represent the complete paths of block memberships at the observation times.

Next the model for edge states is described. It will be convenient to define an indicator for each edge, based on the block memberships of the end nodes. Recall that the process governing an edge depends on the block memberships of the end nodes, such that if two nodes i and j are in block k, then the k^{th} process governs edge ij, and if i and j are in different blocks, then the 0^{th} process is used. Let $Z_{ij}(t)$ be an indicator for which process governs the edge ij at some time t; this is a deterministic function of both $Z_i(t)$ and $Z_j(t)$. Simply, $Z_{ij}(t) = k$ if

$Z_i(t) = Z_j(t) = k$ and $Z_{ij}(t) = 0$ otherwise. We define the corresponding indicator variable at the observation times, $\boldsymbol{z}_{ij}^s$, as a one-of-$(\kappa+1)$ indicator vector (indexed from 0 to κ) as:

$$\boldsymbol{z}_{ij}^s = \begin{cases} (0, \boldsymbol{z}_i^s), \text{ if } \boldsymbol{z}_i^s = \boldsymbol{z}_j^s, \\ (1, 0, \ldots, 0), \text{ if } \boldsymbol{z}_i^s \neq \boldsymbol{z}_j^s. \end{cases}$$

As such $z_{ij0}^s = 1$ if nodes i and j are in different blocks at time t_s and $z_{ijk}^s = 1$ if nodes i and j are both in block k at time t_s.

At time t_1, and given the block memberships of the nodes $\boldsymbol{z}^1$, we assume that the edge states are drawn independently from a Bernoulli distribution with parameter ϕ_k for $z_{ijk} = 1$. Here, ϕ_0 is the probability of an edge appearing between nodes in different blocks at time t_1, whereas ϕ_k is the probability of an edge appearing between nodes in block k at time t_1, for $k = 1, \ldots, \kappa$. Given that $\boldsymbol{z}_{ij}$ is an indicator, the dot product $\boldsymbol{z}_{ij}'\boldsymbol{\phi}$ selects the correct parameter for this Bernoulli distribution. To allow edge states to depend on previous edge states, we let $\boldsymbol{E}(t)$ be a stochastic process with autoregressive components. Given the block membership of the nodes i and j, the edge process $E_{ij1}(t)$ will follow different distributions (similar to the model for time t_1 above). Therefore we require a process for each block and a process for between blocks. Since binary edge states are considered we set each of the $\kappa+1$ edge processes as a CTMC with two states: 0 and 1. The k^{th} CTMC has generator matrix G_k with elements consisting of two rates: α_k and δ_k. The rates $\boldsymbol{\alpha}$ are called the *appearance* rates, the rates which edges transition from state 0 to 1, whilst $\boldsymbol{\delta}$ is the vector of *deletion* rates for transitions in the opposite direction. For comparison to the static SBM and to the edge model for time t_1, we reparameterise with $\phi_k = \frac{\alpha_k}{\alpha_k+\delta_k}$ and $\rho_k = \alpha_k + \delta_k$. The interpretation of this new parameterisation is as follows: ϕ_k is the stationary distribution of an edge in process k, whilst ρ_k is the strength of the autoregressive behaviour. A smaller value for ρ_k signifies more autoregressive behaviour.

Given the nodes i and j, the state of edge ij at times $r < s$ and the parameters $\boldsymbol{\phi}, \boldsymbol{\rho}$, we can write the probability distribution for E_{ij}, if nodes i and j do not

change blocks, as:

$$\begin{aligned}\mathbb{P}[E_{ij}(s) = 1 | E_{ij}(r) = x, Z_{ij}(r) = k, \boldsymbol{\phi}, \boldsymbol{\rho}] &= \phi_k + (x - \phi_k)e^{-\rho_k(s-r)} \\ \mathbb{P}[E_{ij}(s) = 0 | E_{ij}(r), Z_{ij}(r), \boldsymbol{\phi}, \boldsymbol{\rho}] &= 1 - \mathbb{P}[E_{ij}(s) = 1 | E_{ij}(r), \boldsymbol{Z}, \boldsymbol{\phi}, \boldsymbol{\rho}]\end{aligned} \tag{4.2}$$

If node i changes block membership at some time $\tau \in (r, s)$, then we can write the probability of the observed sequence as:

$$\begin{aligned}&\mathbb{P}[E_{ij}(s), E_{ij}(\tau) | E_{ij}(r), Z_{ij}(r), Z_{ij}(\tau), Z_{ij}(s), \boldsymbol{\phi}, \boldsymbol{\rho}] = \\ &\quad \mathbb{P}[E_{ij}(s) | E_{ij}(\tau), Z_{ij}(\tau), \boldsymbol{\phi}, \boldsymbol{\rho}] \, \mathbb{P}[E_{ij}(\tau) | E_{ij}(r), Z_{ij}(r), \boldsymbol{\phi}, \boldsymbol{\rho}] \, .\end{aligned}$$

However, the changepoint τ is unknown, hence we must integrate over this changepoint location and the edge state at this time to obtain the probability of the observed sequence only at observation times as:

$$\begin{aligned}&\mathbb{P}[E_{ij}(s) | E_{ij}(r), Z_{ij}(s) \neq Z_{ij}(r), \boldsymbol{\phi}, \boldsymbol{\rho}] = \\ &\quad \int_\tau \sum_{x=0}^{1} \mathbb{P}[E_{ij}(s) | E_{ij}(\tau) = x, Z_{ij}(\tau), \boldsymbol{\phi}, \boldsymbol{\rho}] \, \mathbb{P}[E_{ij}(\tau) = x | E_{ij}(r), Z_{ij}(r), \boldsymbol{\phi}, \boldsymbol{\rho}] \, p(\tau) \, d\tau.\end{aligned}$$

In this case $p(\tau)$ is the density for the changepoint being at time τ. This is not a simple integral to calculate.

In this paper we assume that the rate of block membership change is slow compared to the observation process (a standard assumption in any hidden Markov model). As such, the probability of a given node changing block membership twice in an observation interval is negligible and thus ignored. Furthermore, for most consecutive observations, the block memberships of a given node i will remain the same, and Equation (4.2) holds. However, if a node does change at time τ, then we can approximate the above integral by using only the block membership at time s:

$$\mathbb{P}[E_{ij}(s) | E_{ij}(r), \boldsymbol{Z}, \boldsymbol{\phi}, \boldsymbol{\rho}] \approx \mathbb{P}[E_{ij}(s) | E_{ij}(r), \boldsymbol{Z}(\tau) = \boldsymbol{Z}(s), \boldsymbol{\phi}, \boldsymbol{\rho}] \, .$$

This approximation assumes that the changepoint $\tau = r$ so that the edge ij is

governed by $\boldsymbol{Z}(s)$ for the entire interval (r, s).

We now consider the likelihood of an observed edge series up to the t^{th} time point, conditional on the block membership and parameters. We define $\Delta_s = t_s - t_{s-1}$ as the time difference between observations s and $s-1$ and let $\boldsymbol{\theta} = (\lambda, \boldsymbol{\omega}, \boldsymbol{\phi}, \boldsymbol{\rho})$. We may write the joint-likelihood of the observed edge states, given the block memberships and parameters, as $\mathbb{P}[\boldsymbol{E}^{1:t}, \boldsymbol{z}^{1:t}|\boldsymbol{\theta}] = \mathbb{P}[\boldsymbol{E}^{1:t}|\boldsymbol{z}^{1:t}\boldsymbol{\theta}]\,\mathbb{P}[\boldsymbol{z}^{1:t}|\boldsymbol{\theta}]$. This comprises of the data likelihood:

$$
\begin{aligned}
\mathbb{P}\big[\boldsymbol{E}^{1:t}|\boldsymbol{z}^{1:t}\boldsymbol{\theta}\big] &= \prod_{i=\!\!\!\!\!\!=j} \mathbb{P}\big[E^1_{ij}|z^1_{ij}, \boldsymbol{\phi}\big] \prod_{s=2}^{t} \mathbb{P}\big[E^s_{ij}|E^{s-1}_{ij}, z^s_{ij}, \boldsymbol{\phi}, \boldsymbol{\rho}\big] \\
&= \prod_{i=\!\!\!\!\!\!=j} \prod_{k=0}^{\kappa} \phi_k^{E^1_{ij} z^1_{ijk}} (1-\phi_k)^{(1-E^1_{ij}) z^1_{ijk}} \\
&\times \prod_{s=2}^{t} \prod_{i=\!\!\!\!\!\!=j} \prod_{k=0}^{\kappa} \left(\phi_k + (1-\phi_k)\exp(-\rho_k \Delta_s)\right)^{z^s_{ijk} E^s_{ijs} E^{s-1}_{ij}} \\
&\times \prod_{s=2}^{t} \prod_{i=\!\!\!\!\!\!=j} \prod_{k=0}^{\kappa} \left(1-\phi_k + \phi_k\exp(-\rho_k \Delta_s)\right)^{z^s_{ijk}(1-E^s_{ij})(1-E^{s-1}_{ij})} \\
&\times \prod_{s=2}^{t} \prod_{i=\!\!\!\!\!\!=j} \prod_{k=0}^{\kappa} \left(\phi_k - \phi_k\exp(-\rho_k \Delta_s)\right)^{z^s_{ijk} E^s_{ij}(1-E^{s-1}_{ij})} \\
&\times \prod_{s=2}^{t} \prod_{i=\!\!\!\!\!\!=j} \prod_{k=0}^{\kappa} \left(1-\phi_k - (1-\phi_k)\exp(-\rho_k \Delta_s)\right)^{z^s_{ijk}(1-E^s_{ij}) E^{s-1}_{ij}}
\end{aligned}
\tag{4.3}
$$

and the block membership likelihood:

$$
\begin{aligned}
\mathbb{P}\big[\boldsymbol{z}^{1:t}|\omega, \lambda\big] &= \prod_{i=1}^{N} \prod_{k=1}^{\kappa} \omega_k^{z^1_{ik}} \\
&\times \prod_{s=2}^{t} \prod_{i=1}^{N} \prod_{k=1}^{\kappa} \left(\omega_k(1-\exp(-\lambda\Delta_s)) + \exp(-\lambda\Delta_s)\right)^{z^s_{ik} z^{s-1}_{ik}} \\
&\times \prod_{s=2}^{t} \prod_{i=1}^{N} \prod_{k=1}^{\kappa} \left(\omega_k(1-\exp(-\lambda\Delta_s))\right)^{z^s_{ik}(1-z^{s-1}_{ik})} .
\end{aligned}
\tag{4.4}
$$

Interest lies in the posterior distribution for the block memberships z^t and parameters $\boldsymbol{\theta}$ at each time point. An algorithm for monitoring a network using the ARSBM is described in Section 4.3. This is a sequential Monte Carlo algorithm, and as such, the posterior estimates are updated each time that new data arrives.

The algorithm presented uses sufficient statistics to allow posterior samples of the parameters to be drawn at any time. The space requirements for the sufficient statistics are independent of the length of the observation series. Therefore, if interest only lies in the current block memberships (or a finite recent history), then the following algorithm can be deployed in an online fashion for an arbitrary length of series. The model simplifies when the observations are taken at regular time intervals. In this case, the difference $\Delta_s = \Delta$ for all s.

4.3. SMC

In this section we provide a sequential Monte Carlo (SMC) algorithm to estimate the block memberships of nodes through time. Due to the sequential nature of the algorithm, the posterior distribution for block memberships can be updated every time new network snapshots arrive. We treat the static parameters $\boldsymbol{\theta} = (\boldsymbol{\omega}, \lambda, \boldsymbol{\phi}, \boldsymbol{\rho})$ as unknown. As such, we require a posterior distribution for both $\boldsymbol{\theta}$ and $\boldsymbol{z}^s$ for $s = 1, \ldots, T$.

SMC algorithms that can handle static parameters include the SMC^2 method (Chopin et al., 2012). This sampler combines two particle filters to estimate the latent states and fixed parameters. However, each time a parameter is resampled, the full path of latent states must be updated from time 1. This could occur at any future time, hence the complexity of the algorithm depends on t, thus is not a truly online procedure. Alternatively, the sampler of Liu and West (Liu and West, 2001) combines ideas from kernel shrinkage with particle filters to develop a particle filter that can estimate fixed parameters. This does not eradicate the issue of particle degeneracy, but can delay it. Therefore, in time, the Monte Carlo error will dominate the filter such that inference can no longer be drawn.

The aforementioned sampling schemes either require storage growing with the number of observations (prohibiting an online approach), or do not abate the problem of particle degeneracy. Moreover, they are black box approaches: they do not exploit any structure in the data model. In Section 4.3.1 we derive sufficient

statistics under a data augmentation scheme that allows sampling of the parameters from the posterior at any time point. Therefore, we can employ the "MCMC within particle filter" methods of Storvik (2002); Fearnhead (2002). In this SMC algorithm, the parameters are estimated at each time based on low dimensional sufficient statistics for the observations and block memberships. As such, particle impoverishment is abated, unlike in the approaches mentioned above. This approach is more specialised than the black-box algorithms above, and can only be used if suitable sufficient statistics exist.

In the remainder of this section, we derive sufficient statistics and integrate them into a sequential importance sampling algorithm for the ARSBM.

4.3.1. Sufficient statistics

In this section we derive sufficient statistics for the parameters $\boldsymbol{\theta}$ based on the observed edge states and block memberships up to time t. To calculate sufficient statistics, we consider the posterior for $\boldsymbol{\theta}$ up to time t:

$$\pi\big(\boldsymbol{\theta}|\boldsymbol{E}^{1:t}, \boldsymbol{z}^{1:t}\big) \propto \pi_0(\boldsymbol{\theta})\, \mathbb{P}\big[\boldsymbol{E}^{1:t}, \boldsymbol{z}^{1:t}|\boldsymbol{\theta}\big] \tag{4.5}$$

In the case that observations are equally spaced in time, we let $\Delta = \Delta_s$ and reparameterise with $\mu = \exp(-\lambda\Delta)$ and $\nu_k = \exp(-\rho_k\Delta)$. By substituting Equations (4.3) and (4.4) into Equation (4.5), the posterior then becomes:

$$\begin{aligned}\pi\big(\boldsymbol{\theta}|\boldsymbol{E}^{1:t}, \boldsymbol{z}^{1:t}\big) \propto \pi_0(\boldsymbol{\theta}) \prod_{k=0}^{\kappa} &\phi_k^{N_{1,k}+N_{01,k}^t}(1-\phi_k)^{N_{0,k}+N_{10,k}^t}\,(1-\nu_k)^{N_{10,k}^t+N_{01,k}^t} \\ &\times (\phi_k + (1-\phi_k)\nu_k)^{N_{11,k}^t}\,(1-\phi_k+\phi_k\nu_k)^{N_{00,k}^t} \\ &\times \prod_{k=1}^{\kappa} \omega_k^{L_k+M_k^{t,+}-1}\,(1-\mu)^{M_k^{t,+}}\,(\omega_k(1-\mu)+\mu)^{M_k^t},\end{aligned} \tag{4.6}$$

where the statistics $S_t = (\boldsymbol{L}, \boldsymbol{M}^{t,+}, \boldsymbol{M}^t, \boldsymbol{N}_1, \boldsymbol{N}_0, \boldsymbol{N}_{11}^t, \boldsymbol{N}_{01}^t, \boldsymbol{N}_{10}^t, \boldsymbol{N}_{00}^t)$ are defined and summarised in Table 4.1.

To implement the sampling algorithm of Storvik, a sampling distribution for

Table 4.1.: Summary statistics for the ARSBM

Statistic	Interpretation	Formula
$L_{1,k}$	The number of nodes assigned to block k at time 1.	$\sum_{i=1}^{N} z_i^1$
$M_k^{t,+}$	The number of nodes which change block membership and join block k up to time t.	$\sum_{i=1}^{N}\sum_{s=2}^{t} z_{ik}^s(1-z_{ik}^{s-1})$
M_k^t	The number of nodes remaining in block k at consecutive observation times.	$\sum_{i=1}^{N}\sum_{s=2}^{t} z_{ik}^s z_{ik}^{s-1}$
$N_{1,k}$	The number of edges governed by process k at time one in state 1.	$\sum_{i=j} z_{ijk}^1 E_{ij}^1$
$N_{0,k}$	The number of edges governed by process k at time one in state 0.	$\sum_{i=j} z_{ijk}^1(1-E_{ij}^1)$
$N_{11,k}^t$	The number of edges which remain in state 1 at consecutive observation times while governed by process k up to observation time t.	$\sum_{i=j}\sum_{s=2}^{t} z_{ijk}^s E_{ij}^s E_{ij}^{s-1}$
$N_{10,k}^t$	The number of edges which transition from state 1 to 0 at consecutive observation times while governed by process k up to observation time t.	$\sum_{i=j}\sum_{s=2}^{t} z_{ijk}^s(1-E_{ij}^s)E_{ij}^{s-1}$
$N_{01,k}^t$	The number of edges which transition from state 0 to 1 at consecutive observation times while governed by process k up to observation time t.	$\sum_{i=j}\sum_{s=2}^{t} z_{ijk}^s E_{ij}^s(1-E_{ij}^{s-1})$
$N_{00,k}^t$	The number of edges which remain in state 0 at consecutive observation times while governed by process k up to observation time t.	$\sum_{i=j}\sum_{s=2}^{t} z_{ijk}^s(1-E_{ij}^s)(1-E_{ij}^{s-1})$

$\boldsymbol{\theta}|S_{t-1}$ is required to draw parameters to use for inference of the block memberships at time t. Note that in Equation (4.6), the parameters separate into a product of joint distributions for (ϕ_k, ν_k) and $(\mu, \boldsymbol{\omega})$. By introducing an auxiliary variable for

each of these pairs, a separable form for the posterior can be found. We introduce the auxiliary variable for $(\mu, \boldsymbol{\omega})$ first, then discuss the pairs (ϕ_k, ν_k) for $k = 0, \ldots, \kappa$.

Consider the parameters $(\mu, \boldsymbol{\omega})$ governing the process which governs changes in block membership for the nodes. This process is modelled as a CTMC with generator Q as in Equation (4.1). Under this model, the probability that node i remains in block k at consecutive times (t_{s-1}, t_s) can be decomposed as: (0) the probability that i does not move, plus (1) the probability i moves at least once but returns to block k by time t_s. Suppose that we could identify which of (0) or (1) occurred for each node. As such, let $c_i^s = c$ if node i stayed in block k in time interval (t_{s-1}, t_s) via (c) for c=0,1. With this additional information $\boldsymbol{c}^{1:t} = \{c_i^s : i \in \mathcal{V}, s = 0, \ldots, t\}$, we can write the likelihood for $\boldsymbol{z}^{1:t}, \boldsymbol{c}^{1:t}$ given ω, λ as:

$$\begin{aligned}
\pi\big(\boldsymbol{z}^{1:t} | \boldsymbol{c}^{1:t}, \boldsymbol{\theta}\big) &\propto \prod_{i=1}^{N}\prod_{k=1}^{\kappa} \omega_k^{z_{ik}^1} \\
&\quad \times \prod_{s=2}^{t}\prod_{i=1}^{N}\prod_{k=1}^{\kappa} \left(\omega_k(1-\mu)\right)^{c_i^s z_{ik}^s z_{ik}^{s-1}} \mu^{(1-c_i^s) z_{ik}^s z_{ik}^{s-1}} \left(\omega_k(1-\mu)\right)^{z_{ik}^s (1 - z_{ik}^{s-1})} \\
&= \prod_{k=1}^{\kappa} \omega_k^{L_k + M_k^{t,+} + M_k^{t,1}} \mu^{M_k^{t,0}} (1-\mu)^{M_k^{t,+} + M_k^{t,1}}.
\end{aligned}$$

We have introduced the additional statistics $M_k^{t,c}$ which count the number of times a node remains in block k with $c_i^s = c$ for $c = 0, 1$. Note that $M_k^{t,0} + M_k^{t,1} = M_K^t$ by construction. Knowledge of $M_k^{t,c}$ makes the posterior distribution for $\mu, \boldsymbol{\omega}$ separable. We can write the likelihood of c_i^s given the data and parameters as:

$$\mathbb{P}[c_i^s = 1 | z_{ik}^s = 1, \boldsymbol{\omega}, \lambda] = \frac{\omega_k(1-\mu)}{\omega_k(1-\mu) + \mu}.$$

Therefore, given that $M_k^{t,1} = \sum_{i=1}^{N} \sum_{s=2}^{t} c_i^s z_{ik}^s z_{ik}^{s-1}$, we find:

$$M_k^{t,1} \sim \text{Bin}\left(M_k^t, \frac{\omega_k(1-\mu)}{\omega_k(1-\mu) + \mu}\right)$$

By assigning a Dirichlet(γ) prior to $\boldsymbol{\omega}$, and a Beta(α, β) prior to μ then the

above posterior yields the following set of distributions:

$$\begin{aligned}
\mu|\boldsymbol{M}^t &\sim \text{Beta}\left(\alpha + \sum_k M_k^{t,0}, \beta + \sum_k M_k^{t,+} + M_k^{t,1}\right), \\
\boldsymbol{\omega}|\boldsymbol{M}^t &\sim \text{Dirichlet}\big(\boldsymbol{L} + \boldsymbol{M}^{t,+} + \boldsymbol{M}^{t,1} + \gamma\big), \\
M_k^{t,1}|\lambda, \boldsymbol{\omega} &\sim \text{Bin}\left(M_k^t, \frac{\omega_k(1-\mu)}{\omega_k(1-\mu)+\mu}\right).
\end{aligned} \tag{4.7}$$

This form naturally leads to a Gibbs sampling approach which we will discuss in Section 4.3.2.

Similarly, an augmentation approach can be applied to the pairs (ϕ_k, ρ_k). Let d_{ij}^s be an auxiliary variable such that $d_{ij}^s = 0$ if edge ij remains in the same state without changing between times t_{s-1} and t_s, and $d_{ij}^s = 1$ if edge ij is in the same state at consecutive time points and changes at least once (therefore the last change in state before time t_s returned ij to the state at time t_{s-1}).

Thus, the terms of the likelihood for both $\boldsymbol{E}^{1:T}, \boldsymbol{d}^{1:T}$ involving terms in $\boldsymbol{\phi}, \boldsymbol{\rho}$ can be rewritten as:

$$\begin{aligned}
\pi\big(\boldsymbol{E}^{1:t}|\boldsymbol{d}^{1:t}, \boldsymbol{z}^{1:t}, \boldsymbol{\theta}\big) \propto & \prod_{i=j}\prod_{k=0}^{\kappa} \phi_k^{z_{ijk}^1 E_{ij}^1}(1-\phi_k)^{z_{ijk}^1(1-E_{ij}^1)} \\
& \times \prod_{s=2}^{t}\prod_{i=j}\prod_{k=0}^{\kappa} \big(\phi_k(1-\nu_k)\big)^{d_{ij}^s z_{ijk}^s E_{ij}^s E_{ij}^{s-1}} \\
& \times \prod_{s=2}^{t}\prod_{i=j}\prod_{k=0}^{\kappa} \nu_k^{(1-d_{ij}^s)z_{ijk}^s(E_{ij}^s E_{ij}^{s-1} + (1-E_{ij}^s)(1-E_{ij}^{s-1}))} \\
& \times \prod_{s=2}^{t}\prod_{i=j}\prod_{k=0}^{\kappa} \big((1-\phi_k)(1-\nu_k)\big)^{d_{ij}^s z_{ijk}^s (1-E_{ij}^s)(1-E_{ij}^{s-1})} \\
& \times \prod_{s=2}^{t}\prod_{i=j}\prod_{k=0}^{\kappa} (1-\phi_k)^{z_{ijk}^s(1-E_{ij}^s)E_{ij}^{s-1}} \phi_k^{z_{ijk}^s E_{ij}^s (1-E_{ij}^{s-1})} \\
& \times \prod_{s=2}^{t}\prod_{i=j}\prod_{k=0}^{\kappa} (1-\nu_k)^{z_{ijk}^s(E_{ij}^s(1-E_{ij}^{s-1}) + (1-E_{ij}^s)E_{ij}^{s-1})} \\
= & \prod_{k=0}^{\kappa} \phi_k^{N_1 + N_{01,k}^t + N_{11,k}^{t,1}}(1-\phi_k)^{N_{0,k} + N_{10,k}^t + N_{00,k}^{t,1}} \\
& \times \prod_{k=0}^{\kappa} \nu_k^{N_{00,k}^{t,0} + N_{11,k}^{t,0}} (1-\nu_k)^{N_{11,k}^{t,1} + N_{00,k}^{t,1} + N_{01}^t + N_{10}^t}
\end{aligned}$$

where $N^{t,d}_{aa,k}$ counts the number of times an edge in block k stays in state a, for $d = 0$, or changes state back to a, for $d = 1$, up to time t.

Again, we may write the distribution of $\boldsymbol{d}|\boldsymbol{\phi}, \boldsymbol{\nu}, \boldsymbol{E}, \boldsymbol{z}$ as:

$$\mathbb{P}\big[d^s_{ij} = 1 | z^s_{ijk} = 1, E^s_{ij} = 0, E^{s-1}_{ij} = 0, \boldsymbol{\phi}, \boldsymbol{\rho}\big] = \frac{(1-\phi_k)(1-\nu_k)}{(1-\phi_k)(1-\nu_k)+\nu_k},$$
$$\mathbb{P}\big[d^s_{ij} = 1 | z^s_{ijk} = 1, E^s_{ij} = 1, E^{s-1}_{ij} = 1, \boldsymbol{\phi}, \boldsymbol{\rho}\big] = \frac{\phi_k(1-\nu_k)}{\phi_k(1-\nu_k)+\nu_k}.$$

Thus, the distributions for $N^{t,1}_{11,k}$ and $N^{t,1}_{00,k}$ are Binomial distributed with the above probabilities and sizes $N^t_{11,k}, N^t_{00,k}$ respectively. Therefore, assigning a Beta prior to ϕ_k with parameters a_k, b_k and a Beta(c_k, d_k) to ν_k, we obtain the following posterior distributions in Equation (4.8):

$$\begin{aligned}
\phi_k|\boldsymbol{N}^t &\sim \text{Beta}\big(N_{1,k} + N^t_{01,k} + N^{t,1}_{11,k} + a_k, N_{0,k} + N^t_{10,k} + N^{t,1}_{00,k} + b_k\big), \\
\nu_k|\boldsymbol{N}^t &\sim \text{Beta}\big(N^{t,0}_{11,k} + N^{t,0}_{00,k} + c_k, N^t_{01,k} + N^t_{10,k} + N^{t,1}_{00,k} + N^{t,1}_{11,k} + d_k\big), \\
N^{t,1}_{00,k}|\phi_k, \nu_k &\sim \text{Bin}\left(N^t_{00,k}, \frac{(1-\phi_k)(1-\nu_k)}{(1-\phi_k)(1-\nu_k)+\nu_k}\right), \\
N^{t,1}_{11,k}|\phi_k, \nu_k &\sim \text{Bin}\left(N^t_{11,k}, \frac{\phi_k(1-\nu_k)}{\phi_k(1-\nu_k)+\nu_k}\right).
\end{aligned} \tag{4.8}$$

Using the above distributions, independent Gibbs samplers can be applied to the variables $\lambda, \boldsymbol{\omega}, M^{t,1}$ and $\phi_k, \rho_k, N^{t,1}_{11,k}, N^{t,1}_{00,k}$ for each $k = 0, \ldots, \kappa$. Therefore, given the sufficient statistics at time $t-1$, a Gibbs sampler can be applied with the augmented statistics $M^{t,1}, N^{t,1}_{11,k}, N^{t,1}_{00,k}$ to provide draws from the posterior for the parameters $(\lambda, \boldsymbol{\omega}, \boldsymbol{\phi}, \rho)$. The details of this algorithm are presented in the next section.

4.3.2. SMC algorithm

The sequential importance sampling algorithms of Storvik (2002) and Fearnhead (2002) can be applied to the ARSBM with the sufficient statistics defined in Section 4.3.1. The aim of this algorithm is to update the posterior distribution each time a new network snapshot arrives. The algorithm achieves this via a set of

particles which each store a realisation of the block structure and the sufficient statistics at time t. Each particle has a corresponding weight, such that the particles and weights form a discrete approximation of the posterior distribution. At each time t, parameter values $\tilde{\boldsymbol{\theta}}$ are sampled from the currently available posterior. This is achieved by performing a Gibbs sampler on the parameters and augmented statistics as discussed in the previous section. Specifically, draws are taken from the distributions in Equations (4.7) and (4.8) for some pre-specified number of steps.

Given $\tilde{\boldsymbol{\theta}}$, a proposed block structure for time t is generated, denoted as $\tilde{\boldsymbol{z}}^t$. The weights are then updated, taking into account the proposed block structure and parameters ability to describe the new snapshot $\boldsymbol{E}^t$. Next, the sufficient statistics are updated for the new snapshot and block structure. Finally, the particles are resampled according to their weights. This removes particles with little support in the posterior. Each of the steps in the algorithm are now provided in detail.

Inference for time 1

An initial set of particles must be generated at time 1 to start the algorithm. In the case of the ARSBM, a parameter value $\tilde{\theta}^p$ is drawn from the prior for each particle p. Given $\tilde{\boldsymbol{\theta}}$ the initial block structures $\tilde{\boldsymbol{z}}^{1,p}$ are drawn from a proposal distribution q_1. This particle is then weighted as in Equation 4.9. There is scope to include information on $\boldsymbol{E}_1$ in the proposal distribution q_1. In the case of the ARSBM, since the initial block structure lies in a large space (all possible assignments of N nodes to κ blocks, which has N^κ elements), simply setting q to the prior is unlikely to be successful. In this paper we draw $\tilde{\boldsymbol{z}}^1$ by clustering the nodes dependent on some initial set of snapshots via a similarity matrix $\boldsymbol{D}$. For example, in assortative networks, counting the number of edges between a pair of nodes with $D_{ij} = \sum_{s=1}^{5} E_{ijs}$ leads to a similarity matrix where nodes in the same block score higher in similarity. This can be combined with the concept of spectral clustering (Von Luxburg, 2007) to generate a similarity matrix which is better able

to separate blocks in the SBM (Lei et al., 2015). Given a similarity matrix $\boldsymbol{D}$, we perform clustering by fitting a Gaussian mixture model with κ components to the rows of $\boldsymbol{D}$. Under this proposal, node i is assigned to block k with probability proportional to the density row i receives in component k of the mixture model.

Given $\tilde{\boldsymbol{z}}$ and initial edge states $\boldsymbol{E}^1$, the parameters $\boldsymbol{\omega}$ and $\boldsymbol{\phi}$ are drawn from the time 1 posterior as in Equation (4.7). The weight is then obtained via Equation (4.9):

$$w_1^p = \frac{f(\tilde{\boldsymbol{z}}^{1,p}|\boldsymbol{\omega})\, g(\boldsymbol{E}^1|\tilde{\boldsymbol{z}}^{1,p}|\boldsymbol{\phi})}{q_1(\tilde{\boldsymbol{z}}^{1,p}|\boldsymbol{\omega}, \boldsymbol{E}^1)} \tag{4.9}$$

Next, the sufficient statistics are initialised for particle p, with only $\boldsymbol{L}, \boldsymbol{N}_0, \boldsymbol{N}_1$ affected. Finally, the particles are resampled with replacement proportional to their weights.

Sequential inference

For all time points t after the first, the set of particles from time $t-1$ is updated to yield a discrete approximation to the posterior on $\boldsymbol{z}^{1:t}, \boldsymbol{\theta}$. For each particle p, a parameter $\tilde{\boldsymbol{\theta}}$ is drawn from the posterior of time $t-1$ using the Gibbs sampling procedure in Equations (4.7) and (4.8) based on the sufficient statistics S_{t-1}^p available from particle p. Given $\tilde{\theta}$, the block structure $\tilde{\boldsymbol{z}}^{tp}$ is drawn from a proposal distribution $q_t(\boldsymbol{z}_t|\boldsymbol{z}^{t-1,p}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1})$ (described in the following). Next, the weight of the proposed particle is updated via Equation (4.10):

$$w_t^p = w_{t-1}^p \frac{f(\tilde{\boldsymbol{z}}^{t,p}|\boldsymbol{z}^{t-1,p}, \boldsymbol{\omega}, \lambda)\, g(\boldsymbol{E}^t|\boldsymbol{E}^{t-1}\tilde{\boldsymbol{z}}^{t,p}, \boldsymbol{\phi}, \boldsymbol{\rho})}{q_t(\tilde{\boldsymbol{z}}^{t,p}|\boldsymbol{\omega}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1})} \tag{4.10}$$

Then, the sufficient statistics are updated to include the information in the most recent snapshot and proposed block structure. Once all particles have been updated, the set is resampled proportional to weight to obtain a set of equally weighted particles.

The proposal distribution for $\boldsymbol{z}^t$ used at time t makes use of the time t edge

state data:

$$q\big(z_{ik}^t = 1|\boldsymbol{z}_i^{t-1}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1}, \boldsymbol{\theta}\big) = \frac{\prod_{j\neq i} g\big(E_{ij}^t|E_{ij}^{t-1}, z_{ik}^t = 1, \boldsymbol{z}_j^t = \boldsymbol{z}_j^{t-1}, \boldsymbol{\theta}\big)\, f\big(z_{ik}^t = 1|\boldsymbol{z}_i^{t-1}, \boldsymbol{\theta}\big)}{\sum_{l=1}^{\kappa}\prod_{j\neq i} g\big(E_{ij}^t|E_{ij}^{t-1}, z_{il}^t = 1, \boldsymbol{z}_j^t = \boldsymbol{z}_j^{t-1}, \boldsymbol{\theta}\big)\, f\big(z_{il}^t = 1|\boldsymbol{z}_i^{t-1}, \boldsymbol{\theta}\big)}. \tag{4.11}$$

This allows block structures with high posterior mass to be proposed. Notice that, although we have a closed form expression for the posterior of $\boldsymbol{z}^t|\boldsymbol{z}^{t-1}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1}, \boldsymbol{\theta}$ up to a normalising constant:

$$\pi\big(\boldsymbol{z}^t|\boldsymbol{z}^{t-1}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1}, \boldsymbol{\theta}\big) \propto \prod_{i=j} g\big(E_{ij}^t|E_{ij}^{t-1}, z_{ij}^t, \boldsymbol{\theta}\big) \prod_{i=1}^{N} f\big(z_i^t|z_i^{t-1}, \boldsymbol{\theta}\big)\,,$$

this involves the likelihood of the edges at time t given the unknown block memberships $\boldsymbol{z}^t$ for all nodes, of which there are κ^N possibilities. The proposal distribution introduced in Equation (4.12) proposes the block memberships for each node in turn, using only the current information on the other nodes, for which there are $N\kappa$ possibilities. This greatly reduces the computation required by assuming that the posterior distribution separates into a product of independent distributions, one for each node. Recall that the nodes do move blocks independently, and only a few nodes will change block at any given time interval. Therefore, when proposing the new block membership for a given node, we assume all other nodes do not move. As such, the proposal distribution has the following form:

$$q\big(z_{ik}^t = 1|\boldsymbol{z}_i^{t-1}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1}, \boldsymbol{\theta}\big) \propto \prod_{j\neq i} g\big(E_{ij}^t|E_{ij}^{t-1}, z_{ik}^t = 1, \boldsymbol{z}_j^t = \boldsymbol{z}_j^{t-1}, \boldsymbol{\theta}\big)\, f\big(z_{ik}^t = 1|\boldsymbol{z}_i^{t-1}, \boldsymbol{\theta}\big)\,. \tag{4.12}$$

This yields a discrete distribution over the κ blocks and as such, the probabilities are normalised by their sum. The proposal at time t, for a set of P particles $\{\boldsymbol{z}^{t,p}, S^{t,p} : p = 1, \ldots, P\}$ is thus:

- For each particle p:
  - Draw $\tilde{\boldsymbol{\theta}}^p$ using the Gibbs sampling moves in Equation (4.7) and (4.8)

conditional on $S^{t-1,p}$.

- For each node $i = 1, \ldots, N$, draw $\tilde{z}_i^{t,p}$ from a Multinomial distribution with probability given by Equation (4.12), depending on $\tilde{\boldsymbol{\theta}}^p$ $\boldsymbol{E}_{i\cdot}^t, \boldsymbol{E}_{i\cdot}^{t-1}$ and z_j^{t-1} for all nodes $j \neq i$.
- Compute the weight of the particle via Equation (4.10).
- Update the sufficient statistics: $S^{t,p} = S(S^{t-1,p}, \boldsymbol{E}^t, \boldsymbol{E}^{t-1}, \tilde{\boldsymbol{z}}^{t,p})$.

- Resample the particles proportional to their weights, setting the new weights to $1/P$.

Algorithm implementations

There is choice in the way the above algorithms is implemented in the Gibbs sampling step:

1. Store the block memberships and sufficient statistics for each particle.
2. Store the block memberships, sufficient statistics and parameter values for each particle.
3. Store the block memberships, sufficient statistics and augmented statistics for each particle.

In case (1), the Gibbs sampler can be initiated from the mean of the parameters from time $t-1$. For case (2), each particle retains the parameter value from time $t-1$. This is used to initiate the Gibbs sampler. Intuitively, if the particles are not very varied, then any particle will have parameters close to the mean so these approaches will be similar. Case (3) stores the augmented statistics instead of the parameter values. Intuitively, case (3) should perform equally well to case (2) since they both store the same number of unknowns from the joint posterior. We try each of these three approaches in Section 4.4 on simulated data.

4.4. Simulated data

We applied the SMC algorithm described in Section 4.3.2 with the three implementations discussed in Section 4.3.2 to a set of simulated networks on 120 nodes. For each simulation, the nodes were split into three blocks with $\boldsymbol{\omega} = (30/120, 40/120, 50/120)$ then simulated for 100 time points with a fixed difference in observation times $\Delta = 1$. In each network, the parameters $(\boldsymbol{\phi}, \boldsymbol{\rho})$ were set to generate different block structures including assortative and disassortative behaviours in the proportion of edges between nodes $(\boldsymbol{\phi})$ and the strength of the autoregressive component $(\boldsymbol{\rho})$. Furthermore, for each setting of $\boldsymbol{\phi}$ and $\boldsymbol{\rho}$, we increased λ between 1/120 and 9/120. Since the expected number of changes in a given time period Δ is $\lambda\Delta(1 - \boldsymbol{\omega}'\boldsymbol{\omega})$, this yields an expected number of changes in block memberships between 0.653 and 6 per unit time. The parameter settings lead to 16 data sets shown in Table 4.2.

In all cases, we used $P = 250$ particles. We used $G = 10$ Gibbs step for implementation (1) and $G = 1$ for implementations (2) and (3) to sample parameters from the posterior distribution of $\boldsymbol{\theta}$. Case (1) uses more computation but less storage than cases (2) and (3).

The initialisation procedure used a Gaussian mixture model to assign nodes to blocks at time one. This Gaussian mixture model was fit to a function of the first five snapshots. This function aims to capture the group structure. Specifically, we first map the initial snapshots to a similarity matrix W with $W_{ij} = \sum_{s=1}^{5} E_{ijs}$. Secondly, the spectral decomposition of W is found as $D = \text{Spec}(W)$. This maps the matrix W into a space where similar nodes are closer together (Von Luxburg, 2007). The first κ columns of D are treated as "features" of the nodes before finally fitting the Gaussian mixture model with the rows of D as the data for each node as in Section 4.3.2.

To evaluate the performance of the samplers, we compute the v-measure (Rosenberg and Hirschberg, 2007) at each time point. This is an information theoretic measure which scores a clustering between 0 and 1. If the clustering is perfect (i.e.

Table 4.2.: Parameter settings for simulated data sets.

Index	λ	ϕ_0	ϕ_1	ϕ_2	ϕ_3	ρ_0	ρ_1	ρ_2	ρ_3
1	1.0	0.05	0.6	0.7	0.8	0.2	0.4	0.45	0.5
2	1.0	0.05	0.6	0.7	0.8	0.8	0.4	0.45	0.5
3	1.0	0.6	0.1	0.075	0.05	0.2	0.4	0.45	0.5
4	1.0	0.6	0.1	0.075	0.05	0.8	0.4	0.45	0.5
5	3.0	0.05	0.6	0.7	0.8	0.2	0.4	0.45	0.5
6	3.0	0.05	0.6	0.7	0.8	0.8	0.4	0.45	0.5
7	3.0	0.6	0.1	0.075	0.05	0.2	0.4	0.45	0.5
8	3.0	0.6	0.1	0.075	0.05	0.8	0.4	0.45	0.5
9	6.0	0.05	0.6	0.7	0.8	0.2	0.4	0.45	0.5
10	6.0	0.05	0.6	0.7	0.8	0.8	0.4	0.45	0.5
11	6.0	0.6	0.1	0.075	0.05	0.2	0.4	0.45	0.5
12	6.0	0.6	0.1	0.075	0.05	0.8	0.4	0.45	0.5
13	9.0	0.05	0.6	0.7	0.8	0.2	0.4	0.45	0.5
14	9.0	0.05	0.6	0.7	0.8	0.8	0.4	0.45	0.5
15	9.0	0.6	0.1	0.075	0.05	0.2	0.4	0.45	0.5
16	9.0	0.6	0.1	0.075	0.05	0.8	0.4	0.45	0.5

assigns all nodes to the right blocks) then the v-measure is 1. A v-measure of 0.8 is considered a good score in terms of cluster evaluation (Rosenberg and Hirschberg, 2007). To evaluate the performance of the parameter estimation we measure the mean bias over time and the mean absolute deviation (MAD).

The mean v-measure over a data set is computed as:

$$\mathrm{vm}\big(\boldsymbol{z}^{1:T}\big) = \frac{\sum_{s=1}^{T}\sum_{i=1}^{N}\sum_{p=1}^{P} w_s^p \mathrm{vm}(z_{im}^p, Z_{im})}{N\sum_{i=1}^{N}\sum_{p=1}^{P} w_s^p},$$

where $\mathrm{vm}(\boldsymbol{x}, \boldsymbol{y})$ is given by Rosenberg and Hirschberg (2007).

The MAD of an estimate $\hat{\boldsymbol{\theta}}$ to the true value $\boldsymbol{\theta}$ is:

$$\mathrm{MAD}\Big(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}\Big) = \frac{\sum_{s=1}^{T} |\hat{\boldsymbol{\theta}}_s - \boldsymbol{\theta}|}{T}.$$

and similarly the bias is:

$$\text{bias}\left(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}\right) = \frac{\sum_{s=1}^{T} \hat{\boldsymbol{\theta}}_s - \boldsymbol{\theta}}{T}.$$

Figures 4.1, 4.2 and 4.3 show the performance of the methods via v-measure, MAD and bias for the posterior mean respectively. Notice that method 1 and 2 perform similarly in all measures and that method 3 performs comparably in v-measure but leads to more varied bias and MAD. For these reasons, we propose that method 2 is preferred since it is more accurate than method 3 and requires less computation than method 1.

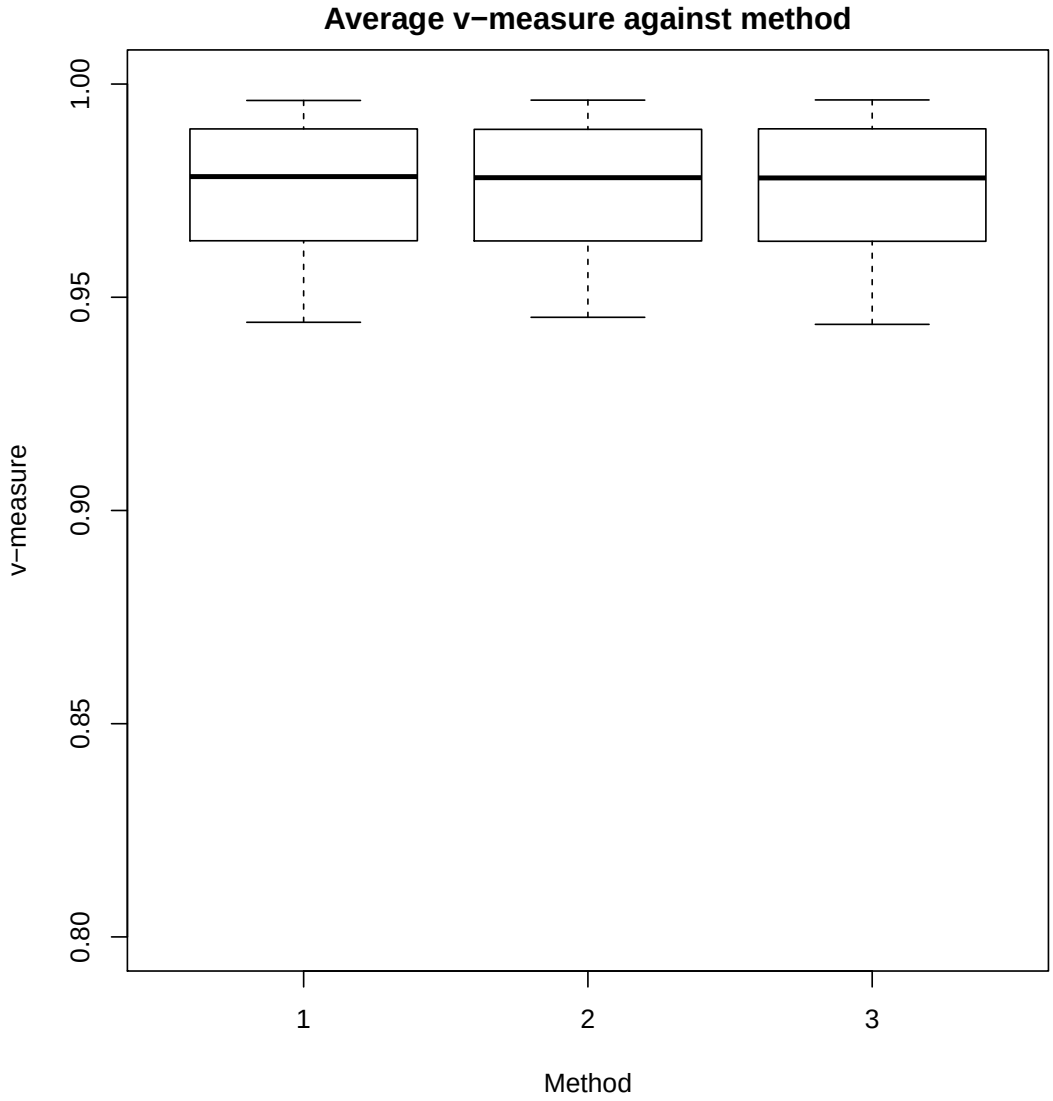


Figure 4.1.: v-measure for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states.

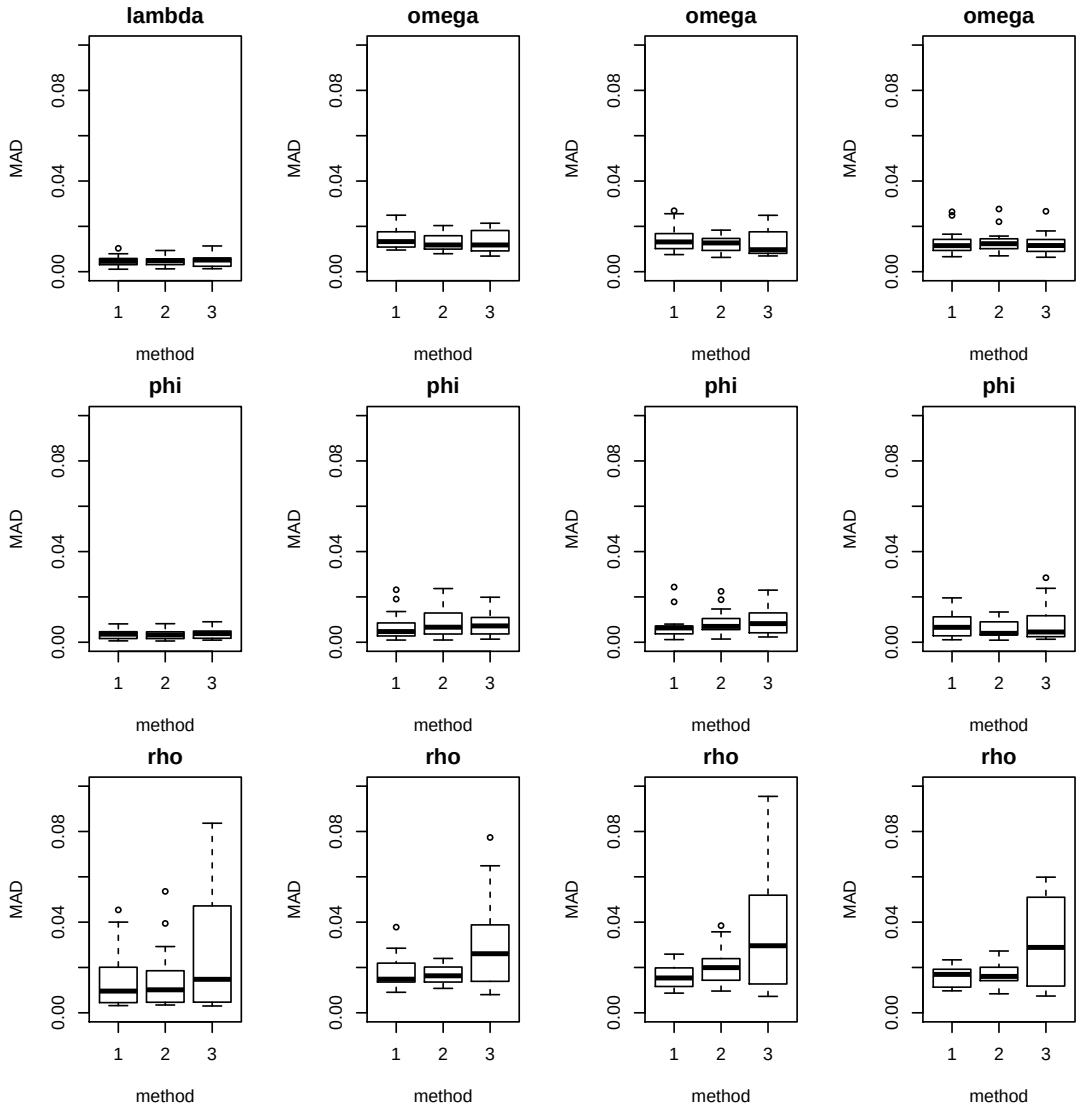


Figure 4.2.: Mean absolute deviation for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states.

The results for method 2 are now compared across the individual parameters. Figure 4.4 compares the v-measure against the true value for λ. A higher value of λ leads to more changes in block membership making the identification of blocks more difficult since the length of data which can identify if a node belongs to a particular block is shorter. Therefore, as expected, the v-measure decreases with increasing λ. Note however that for the highest value of λ, the block structure is still identified with an average v-measure of 0.95, representing strikingly good performance.

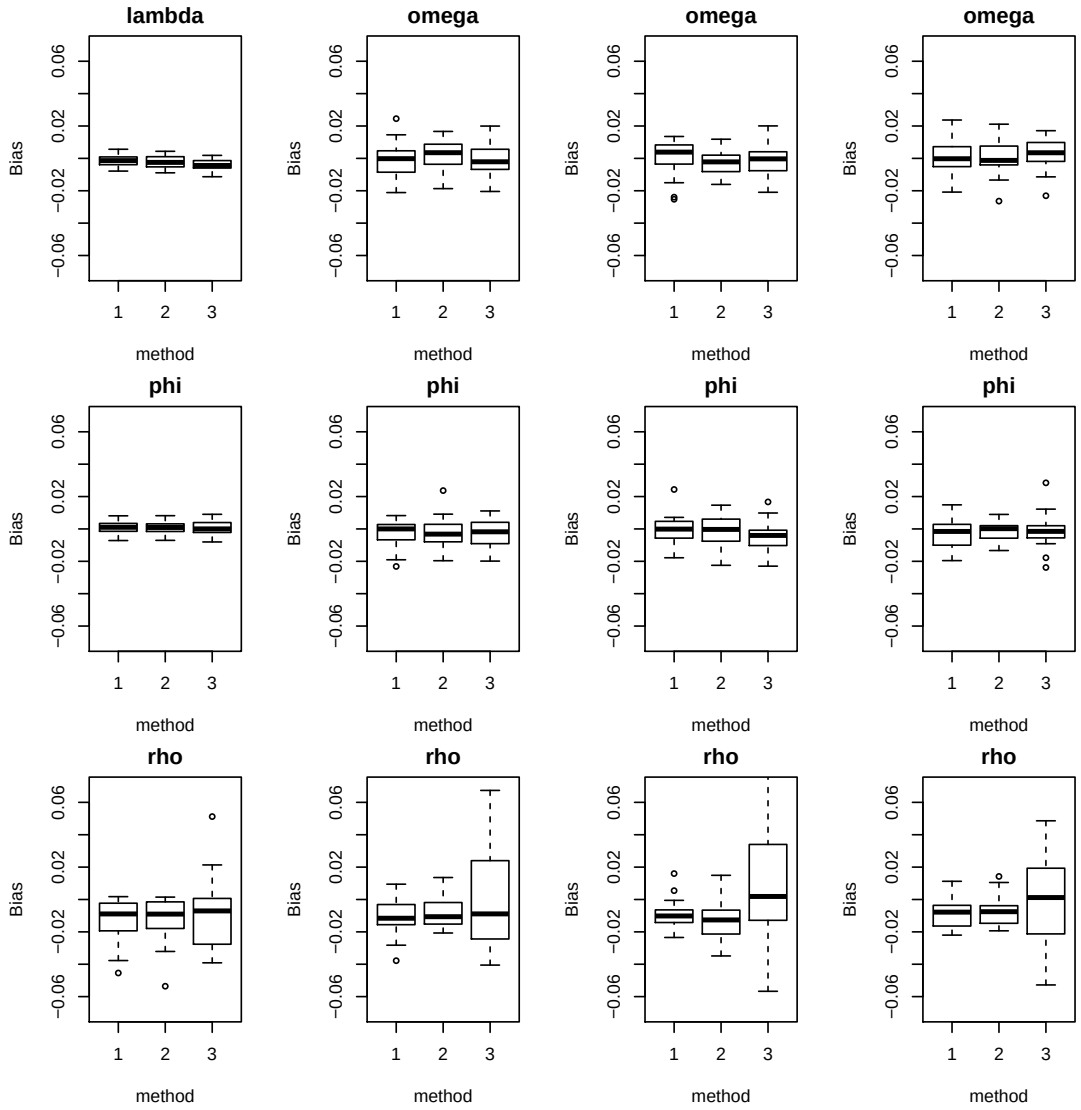


Figure 4.3.: Bias for simulated networks against method. 1 - Gibbs from mean, 2 - Gibbs from previous particle, 3 - store augmented states.

4.4.1. Comparison to offline methods

Previous methods have considered the problem of detecting changes in block membership under a dynamic stochastic block model. Most of these extensions consider the snapshots as independent draws from a static SBM, with all dynamics modelled via the latent block memberships. Matias and Miele (2017) develop a variational approximation to the dynamic SBM which can detect changes in block membership. This is an offline method, requiring the entire series of snapshots. Various authors (Yang et al., 2011; Xu and Hero, 2014) have developed online methods for the detection of changes in block structure in the case of independent snapshots.

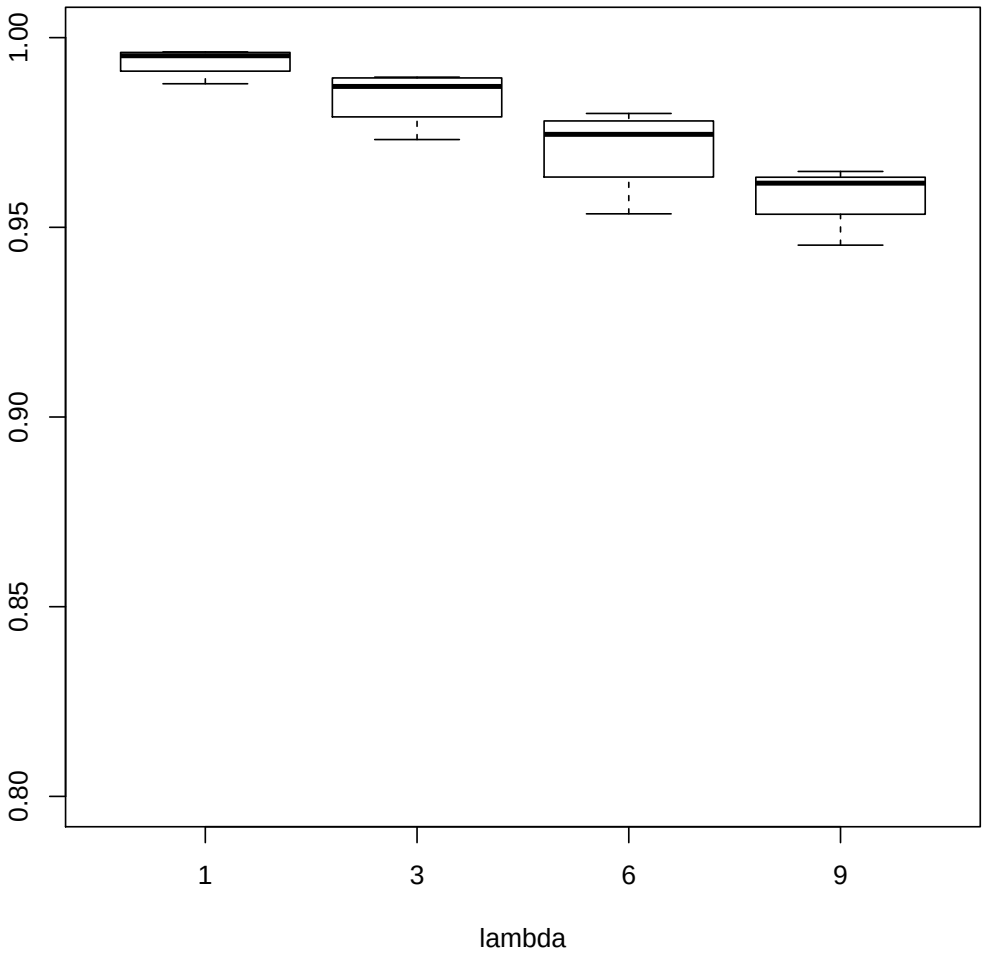


Figure 4.4.: v-measure against true λ values with method 2.

On the other hand, in Ludkin et al. (2017), a reversible jump Markov chain Monte Carlo (RJMCMC) algorithm was proposed to infer the changing block membership of nodes in the autoregressive stochastic block model. This also requires the full series of snapshots.

The method in this paper performs online inference and allows autoregressive components in the edge state distribution. We now compare the RJMCMC output to the SMC algorithm in Section 4.3 by analysing four simulated networks first described in Ludkin et al. (2017). These consist of either 72 or 120 nodes split into 3. In all cases the between block parameters were set as $(\phi_0, \rho_0) = (0.1, 0.2)$ while the within block parameters were $(\phi_k, \rho_k) = (0.5, 1.2)$. The rate of change

in block membership is set such that either $\lambda = 0.01$ or 0.03 leading to either 0.75 or 1.25 changes per time point (in the 72 node networks) or either 2.5 or 4.15 changes per time point (in the 120 nodes networks). Snapshots of the network were taken at unit time intervals for 30 snapshots in total. Since the RJMCMC algorithm assumes a fixed $\boldsymbol{\omega}$, with $\omega_k = 1/\kappa$ in all cases, we fix $\boldsymbol{\omega}$ in the SMC implementation for a fair comparison. For the SMC we use the first 5 snapshots to estimate the initial block structure, 250 particles and 20 Gibbs steps. We evaluate the performance of the two approaches using the v-measure for block structure and bias for parameters. These were defined in Section 4.4.

We find that the SMC is consistently better at determining the simulated block structure as shown by the v-measure of close to 1 obtained in Figure 4.5. This is due to the proposal distribution introduced in Section 4.3. When a change in block membership occurs at time t, this proposal is very good at determining the correct block structure at time t, given that the current block structure at time $t-1$ is a good fit to the data. Contrast this with the RJMCMC implementation which starts with no changes in block structure and proposes changes at random times in a random node. This is less efficient at searching the space of changes compared to the SMC algorithm requiring a long burn-in period to identify changes.

On the other hand, the RJMCMC can "correct" changes which are not a good fit. Due to the sequential nature of the SMC implementation, block structure can only be inferred once at any given time t. Therefore, if errors enter the block structure in every particle, they either need to be corrected at a later time (meaning the estimation of λ and $\boldsymbol{\omega}$ will be biased) or the sampler diverges from the true block structure, with no hope of recovery. This is an unlikely event, since the assumption that snapshots are recorded faster than the rate of node migration implies that only a few nodes will change block at a given time. As such, the problem of divergence is only a minor concern if the assumptions of the model are upheld.

When considering the ability to estimate parameters, the performance of the

samplers are reversed, with the RJMCMC providing better estimates of the parameters and their posterior distributions. The estimates are shown in Table 4.3. Notice that the bias in the RJMCMC inference contains 0 in all cases in the right of Figure 4.6. This is to be expected, since the RJMCMC sampler has access to all snapshots and hence can compute the complete likelihood. Due to the dynamic nature of the networks considered, the time t posterior may have low density at the true parameter values. For example, in the ARSBM, since nodes move between blocks, a small block k can easily contain half the expected number of nodes. This leads to an estimate of ω_k which is smaller than expected. Over time this estimate should improve, but at the start of the series, the parameters can be very biased due to small fluctuations like the above. Therefore, in these simulated data (with only 30 observation times) the SMC does start with large biases but soon settles on the true values. In both cases the time $t = 30$ posterior distributions are used for the SMC. Notice that it is the rate in edge change $\boldsymbol{\rho}$ that is most biased.

Although the parameter estimation is less accurate, the SMC is a much faster algorithm and can be run in a sequential manner, making online network monitoring a possibility. Notice that the complexity is fixed with respect to time since only the sufficient statistics are stored, whose dimensionality does not depend on time.

4.5. Application to dynamic contact network

In this section we apply the SMC algorithm from Section 4.3 to a network of mice.

4.5.1. Mice network

The data set comprises of 257 mice whose sleeping habits were analysed during a period of 54 days. During this time, researchers recorded which nest-box the mice slept in. A subset of 107 mice (who did not interact with the other 150) are analysed. Previous analysis in Ludkin et al. (2017) shows that six blocks are formed

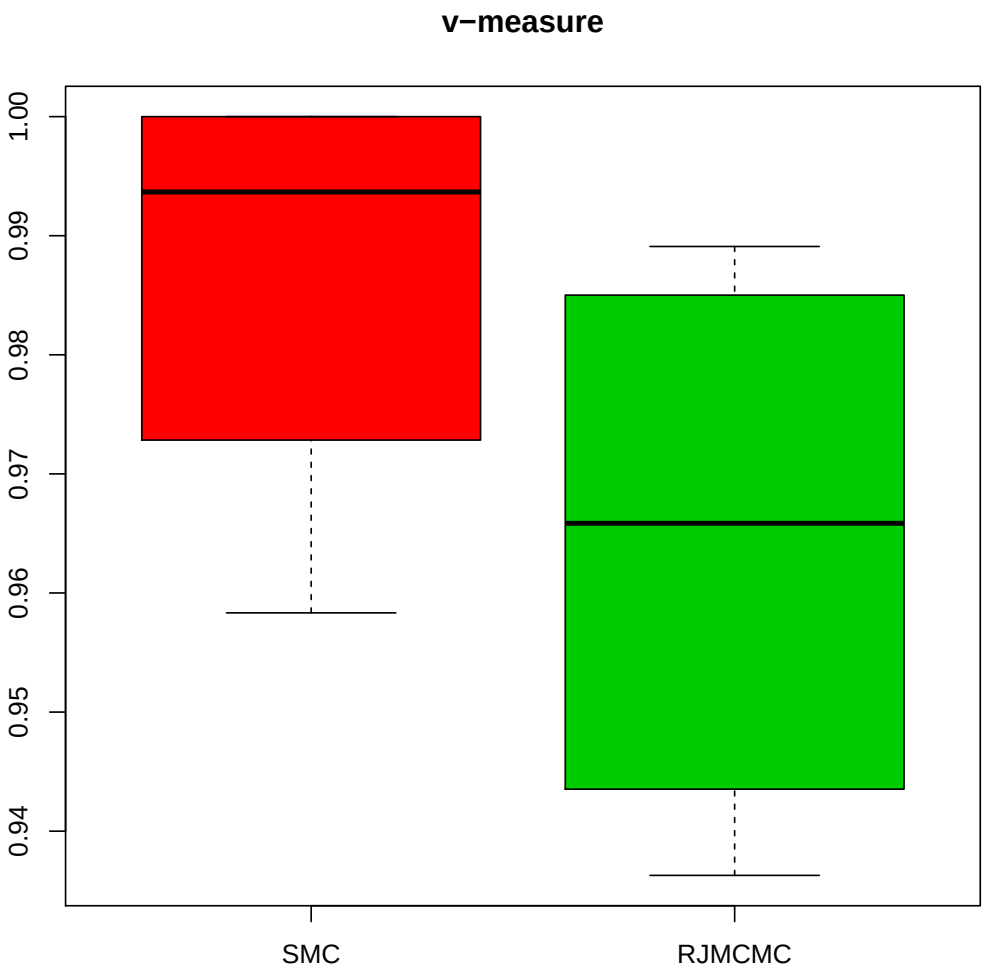


Figure 4.5.: Comparison of v-measure for simulated networks under SMC and RJMCMC algorithms.

in this network. It is also known that some mice drop out of the network and may return at a later time. To model this, we set the parameters $\phi_6 = 0, \rho_6 = \infty$. Furthermore, if a mouse has no contacts at time t, it is automatically assigned to the sixth block. All other block migrations are inferred via the algorithm in Section 4.3.2. The series contains two time points where the number of interactions drops significantly within the network. This is due to the researchers only partially recording the data on some days, therefore, we drop these time points from the analysis. We used 1000 particles and 50 Gibbs steps with five initial snapshots to determine the block structure at time t_1. The maximum *a posteriori* block

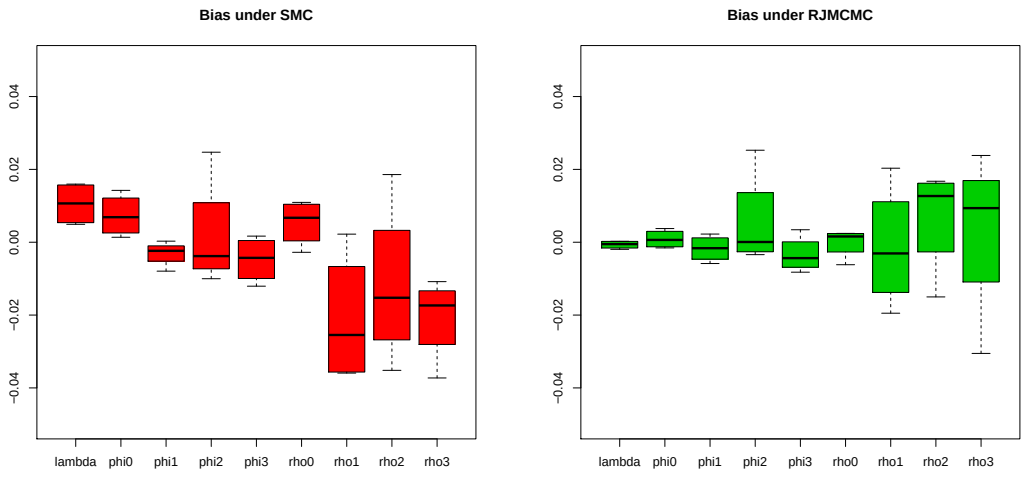


Figure 4.6.: Comparison of bias in parameters under SMC (left) and RJMCMC (right) for example networks.

Table 4.3.: Posterior means for SMC and RJMCMC algorithms in simulated networks.

Parameter	Truth	SMC	RJMCMC	Truth	SMC	RJMCMC
λ	0.0105	0.0164	0.0108	0.0345	0.0504	0.0332
ϕ_0	0.1	0.1014	0.099	0.1	0.1142	0.1022
ϕ_1	0.5	0.4977	0.4942	0.5	0.5003	0.5022
ϕ_2	0.5	0.5247	0.5253	0.5	0.4900	0.4982
ϕ_3	0.5	0.4922	0.4968	0.5	0.4879	0.4918
ρ_0	0.2	0.1972	0.1938	0.2	0.2099	0.2023
ρ_1	1.2	1.2022	1.2203	1.2	1.1646	1.1919
ρ_2	1.2	1.2186	1.2167	1.2	1.1816	1.2157
ρ_3	1.2	1.1892	1.2238	1.2	1.1627	1.2100
Parameter	Truth	SMC	RJMCMC	Truth	SM	RJMCMC
λ	0.0105	0.0153	0.0106	0.0345	0.0500	0.0325
ϕ_0	0.1	0.1037	0.0984	0.1	0.1100	0.1038
ϕ_1	0.5	0.4921	0.5002	0.5	0.4975	0.4965
ϕ_2	0.5	0.4970	0.4966	0.5	0.4954	0.5020
ϕ_3	0.5	0.4993	0.4945	0.5	0.5017	0.5034
ρ_0	0.2	0.2035	0.2008	0.2	0.2109	0.2024
ρ_1	1.2	1.1641	1.2019	1.2	1.1844	1.1805
ρ_2	1.2	1.1648	1.1850	1.2	1.1879	1.2097
ρ_3	1.2	1.1811	1.1695	1.2	1.1842	1.2087

memberships are shown in Figure 4.7 with time along the x-axis, mouse identity along the y-axis and the colour of pixel (x, y) representing which block mouse y belongs to at time x. In this case, the pink block is the block for mice absent from the network. The block membership between the blue and yellow blocks becomes blurred near the end of the series. The block membership for each mouse is plotted in Appendix C.2.

The final time posterior means of the parameters are given in Table 4.4 along with the mean estimates for the corresponding blocks in the RJMCMC procedure. Notice that unlike the RJMCMC algorithm, we treat $\boldsymbol{\omega}$ as unknown, hence the blocks are not identical. When considering the trace plots for the parameters in Appendix C.1, they do not seem stable, hence there is a case that the assumption of fixed parameters is not valid.

Table 4.4.: Final time posterior mean and variance for mice network under SMC and the corresponding estimates for the RJMCMC algorithm.

Parameter	λ	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6
RJMCMC Mean	0.0019	-	-	-	-	-	-
SMC Mean	0.0431	0.2145	0.2115	0.1960	0.1111	0.0979	0.1689
SMC s.d.	0.0006	0.0355	0.0292	0.0028	0.0091	0.0087	0.0076
Parameter	ϕ_0	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6
RJMCMC Mean	0.0004	0.6994	0.6660	0.4349	0.4473	0.6821	0
SMC Mean	0.0006	0.5039	0.6922	0.5220	0.8289	0.7658	0
SMC s.d.	0.0000	0.0007	0.0131	0.0053	0.0060	0.0027	0
Parameter	ρ_0	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6
RMCMC Mean	1.3164	0.5138	0.4104	0.5650	0.4077	0.6851	0
SMC Mean	1.0466	0.2752	0.5294	0.1073	0.4417	0.3933	0
SMC s.d.	0.0180	0.0020	0.0138	0.0100	0.0061	0.0055	0

4.6. Closing remarks

In this paper we have introduced a particle filter algorithm with MCMC moves to infer the block membership and parameters for the autoregressive stochastic block model. This makes use of a data augmentation scheme to allow samples of the

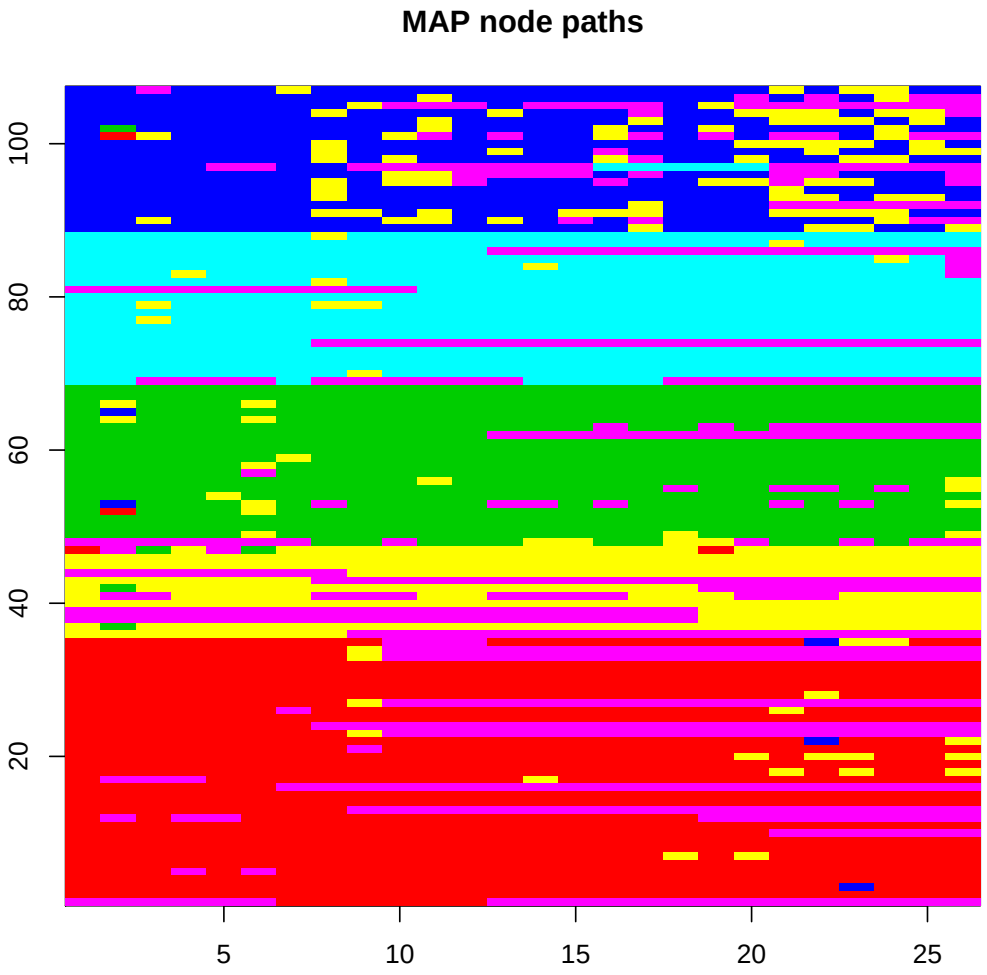


Figure 4.7.: Maximum *a posteriori* block memberships of mice.

static parameters to be drawn from the posterior at any time point. Furthermore, the implementation depends on sufficient statistics which are updated sequentially at each observation time. The size of these statistics is independent of time, allowing a fully online algorithm to be developed. The efficacy of the approach is demonstrated on both simulated data in Section 4.4 and real data in Section 4.5. Block memberships are tracked well through time, as demonstrated on simulated data in Section 4.4. Compared to offline approaches, the block membership detection is comparable, if not better, whereas parameter estimation can be more varied. This is apparent in both the simulated and real data examples.

Interesting extensions to the ARSBM include extending to arbitrary edge-states

(i.e. weighted edges). This would extend the applicability of the model to richer network data sets. In some cases the number of blocks in the network may change, via two blocks merging into one or a block splitting in two. Methods to capture these large scale changes would complement the methods in this paper for detecting more fine-grain changes in block structure.

5. Perspectives and future directions

In this chapter some extensions and future directions of the research presented in this thesis are discussed.

Considering the sampling procedures in Chapter 2, further research is needed for the handling of missing edge-state data. The algorithms proposed in Chapter 2 assume that all edge-states are included in a given network data set. One approach to relax this assumption is to model the missing data. If data is missing completely at random, that is data are missing independent of the observation process, this could be modelled via a specific edge-state distribution. For example, if all edge-states follow a distribution G, but some edge-states are unobserved or missing completely at random with probability η, then the distribution:

$$\bar{G}(E_{ij}) = \eta\delta_0 + (1-\eta)G(E_{ij})$$

is a mixture model for such missing data scenarios. The components are the edge-state distribution, G, if data is available, which occurs with probability $1-\eta$. The other component is a Dirac mass at 0, which encodes the missingness of data, which has a probability η of occurring. This covers the case where data appear as 0 but are missing at random.

On the other hand, a possible network structure could be enforced, where some edge-states are impossible. As an example, consider a road network where traffic flows are recorded between nodes placed at the intersection points of roads. Many edge-states will be 0 since there is no road between many nodes. However, this could be recorded as NA, it is impossible to get a value, rather than stating it

is 0. The mixture model above does not apply in this case, hence additional methodologies need to be developed to perform inference in this case.

The inclusion of covariate information into the inference procedures in Chapter 2 should be straightforward: such information can be included in the edge-state distribution G. On the other hand, information on the nodes could be considered informative of the block membership distribution F. For example, knowledge of peoples hobbies in a social network may be indicative of their block membership: blocks may divide the network into hobby groups. On the other hand, by including a parameter in the edge-state distribution such that people who share the same hobby are more likely to interact, then the block structure in the base rate of interaction may disappear. To make this concrete consider the two models in Equations (5.1) and (5.2).

$$\begin{aligned} \boldsymbol{z} &\sim \text{Multinomial}(\text{logit}(\boldsymbol{\omega} + \boldsymbol{\alpha}' X)) \\ \boldsymbol{E}|\boldsymbol{z} &\sim G(\boldsymbol{\theta}) \end{aligned} \tag{5.1}$$

In Equation (5.1), the parameter $\boldsymbol{\alpha}$ dictates the strength of covariates for informing block membership. In the example above, X may encode sport team membership where $X_{ik} = 1$ shows person i is a members of sport team k. Thus, if α_k is positive, then members of sport team k are more likely to be members of the same block k. If sport team membership does partition the social network in the manner of an assortative SBM, then parameters $\boldsymbol{\theta}_k > \boldsymbol{\theta}_0$. This would be an interpretable result: $\alpha_k > 0$ implies membership of sport team k makes it more likely that a member of sport team k is a member of the social block k. On top of this, since an assortative structure is assumed in this toy example, membership of social block k makes a member more likely to interact with members of social block k than other people in the network. A reasonable conclusion is that members of sports teams are more likely to interact with each other than members of other sports teams.

$$\begin{aligned} \boldsymbol{z} &\sim \text{Multinomial}(\boldsymbol{\omega}) \\ \boldsymbol{E}|\boldsymbol{z} &\sim G(\boldsymbol{\theta} + \boldsymbol{\beta}' X) \end{aligned} \tag{5.2}$$

On the other hand, under the model in Equation (5.2), implies that group membership is random, but the interactions are informed by the covariate information. Thus, in the example, members of the same sport team k are more likely to interact if β_k is positive. Under such a model, and the assortative behaviour assumed above, all interactions between members of a sports team could be explained via β. This could lead to no block structure in $\boldsymbol{z}$. Of course the final conclusion would be the same: sports team membership makes it more likely that two people interact.

The above example highlights the care required in the choice of inclusion of covariate information for nodes. Even though the potential conclusions could be the same from an application perspective, for modelling such phenomenon the model in Equation (5.1) seems more appealing

Covariate information could be available on the edges, not just the nodes. In such cases, the model in Equation (5.2) is a natural choice.

Future research on the ARSBM could allow for non-binary edge-states. Note that in Chapter 3 and 4, the edge-states are considered as binary random variables. This is exploited to allow fast computation of the likelihood since, at consecutive time points, there are only four possible observations: either the edge-state remained in state 0, remained in state 1, switched from 0 to 1, or switched from 1 to 0. Furthermore, for the RJMCMC sampler in Chapter 3, binary edge-states were useful for the augmentation scheme. When augmenting the state of an edge at some intermediate time, there are only two possible states leading to a small discrete distribution.

Hence, to directly extend the methodology of Chapter 3, an efficient augmentation scheme is required. If states are augmented without care, then the proposed insertion of a change will be a poor fit in the augmented space, leading to small acceptance probabilities and thus poor mixing of the MCMC chain. For non-binary

edge-states, an alternative augmentation scheme is required to augment a state at a changepoint between two observations. This would lead to values that are likely to have occurred in the observation interval.

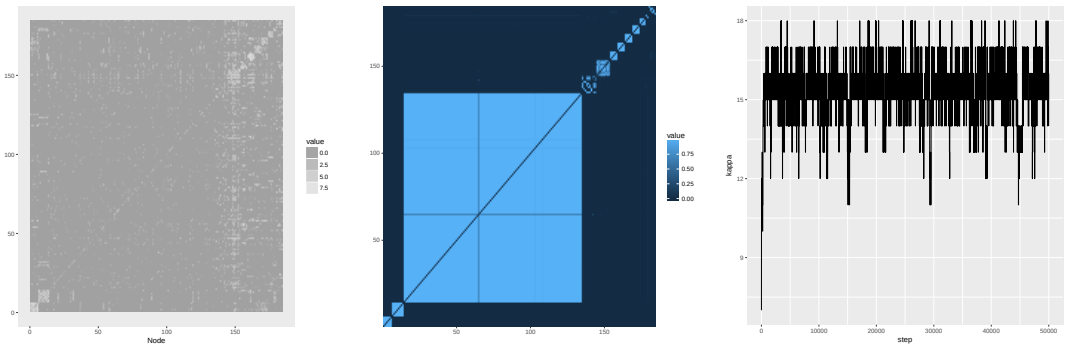
Recent research for the static stochastic block model has considered scalable inference algorithms. Recent work by El-Helw et al. (2016); Li et al. (2016) has produced a stochastic gradient Langevin dynamics algorithm to perform Bayesian inference for the static SBM in a distributed computing environment. Extending such an algorithm to the dynamic ARSBM would be non-trivial but would offer faster inference. For example, by sub-sampling nodes in the network in a principled way, the authors obtain unbiased estimates for the parameters of the SBM to include in their MCMC algorithm. Finding an analogue in the ARSBM must include sub-sampling in time to speed up computations. However, by sub-sampling in the time domain, changes in block structure may be missed, leading to longer MCMC chains to perform the same inference for non-sub-sampled MCMC algorithms. If such sub-sampling procedures can be produced then inference times for the ARSBM with changes in block membership could be drastically reduced.

A. Appendix for arbitrary edge-states and unknown number of blocks in the stochastic block model

A.1. Enron

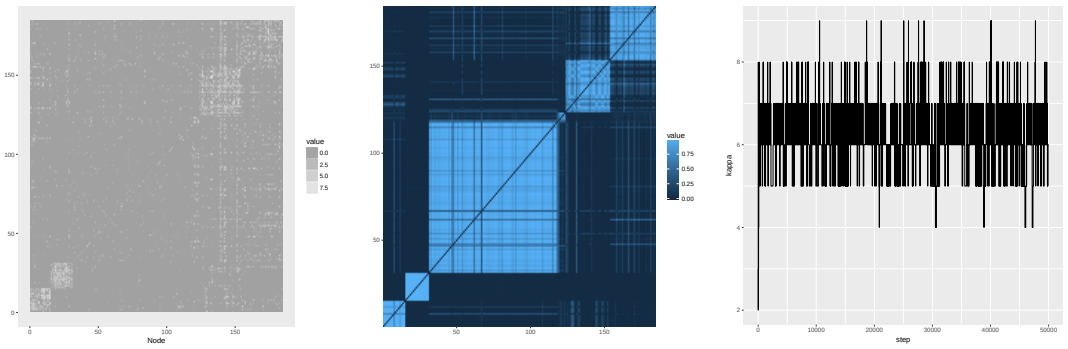
The priors use in the analysis of the Enron data in Section 2.6.2 used reference and vague priors. To test robustness under stronger prior assumptions, the split-merge algorithm was also run for 50,000 steps with a stronger prior in the both the Poisson and negative binomial edge models. Specifically, for the between block parameter, the parameters r_0, p_0 were assigned a Gamma(1/5, 10) and a Beta(1, 1000) respectively. Therefore, the prior mean (variance) for r_0 is 1/50 (1/500) and for p_0 is approximately 1/1000 (1×10^{-7}). These prior means give a mean edge-state of 20 between nodes in different blocks. As for the between block parameters, the parameters r_k, p_k were assigned a Gamma(1/3, 10) and a Beta(1, 1000) respectively. In this case, the prior mean (variance) for r_k is 1/30 (1/300) and for p_k is approximately 1/1000 (1×10^{-7}). At the expected values, the mean edge-state is 33. Therefore, these priors enforce a prior assortative structure, where the between-block edge-states have smaller expected value than within-block edge-states. The respective block structures found under both vague and strong prior

specifications are very similar as shown by comparing Figures A.2 to Figures 2.10. Furthermore, the parameter values for all models are given in Table A.1. Notice these are similar for the first six blocks under the negative binomial model and first 10 for the Poisson model (i.e. the modal number of blocks in each case).



(a) Edge states (log scale) – permuted (b) SM joint block membership posterior (c) SM trace plot for κ

Figure A.1.: Posterior summaries for block membership in Enron network with Poisson edge-state model and strong prior.



(a) Edge states (log scale) – permuted (b) SM joint block membership posterior (c) SM trace plot for κ

Figure A.2.: Posterior summaries for block membership in Enron network with negative binomial edge-state model and strong prior.

Table A.1.: Mean posterior parameter values and 95% credible interval for Enron network under Poisson and negative binomial models with vague and strong priors.

Vague	Poisson(λ)	NBin(r)	NBin(p)
0	2.948 (2.832, 3.011)	0.016 (0.014, 0.018)	0.012 (0.009, 0.014)
1	348.494 (343.838, 356.793)	0.189 (0.116, 0.519)	0.002 (0.001, 0.009)
2	106.433 (103.874, 109.264)	0.313 (0.268, 0.374)	0.007 (0.005, 0.010)
3	0.368 (0.354, 0.384)	0.009 (0.007, 0.012)	0.035 (0.022, 0.052)
4	116.907 (0.562, 120.995)	0.059 (0, 0.267)	0.006 (0, 0.010)
5	202.968 (70.603, 265.529)	0.142 (0.076, 0.229)	0.003 (0, 0.003)
6	141.434 (94.038, 157.022)	0.077 (0.056, 0.158)	0.004 (0, 0.007)
7	671.401 (661.435, 682.603)	0.130 (0, 3737.000)	0.006 (0, 0.978)
8	88.199 (78.024, 182.127)	0.177 (0, 19001.158)	0.004 (0, 0.996)
9	86.587 (0.002, 638.540)		
10	86.948 (0.015, 725.435)		
11	83.801 (0.003, 453.429)		
12	84.072 (0.299, 221.739)		
13	85.986 (0.644, 232.241)		
14	85.503 (0.012, 279.680)		
Strong	Poisson(λ)	NBin(r)	NBin(p)
0	2.901 (2.829, 3.025)	0.016 (0.015, 0.018)	0.012 (0.010, 0.013)
1	348.022 (344.359, 356.772)	0.074 (0.056, 0.093)	0.001 (0, 0.001)
2	106.010 (103.385, 108.543)	0.109 (0.091, 0.124)	0.002 (0.001, 0.003)
3	0.366 (0.354, 0.381)	0.007 (0.006, 0.009)	0.016 (0.011, 0.025)
4	120.722 (0, 121.046)	0.012 (0, 0.014)	0 (0, 0.001)
5	200.602 (71.444, 268.602)	0.105 (0, 0.113)	0.002 (0, 0.003)
6	140.993 (97.647, 147.590)	0.066 (0.026, 0.080)	0.005 (0, 0.005)
7	671.162 (660.071, 681.965)	0 (0, 0.005)	0 (0, 0.002)
8	84.350 (0.012, 356.456)	0.001 (0, 0.004)	0 (0, 0.002)
9	84.835 (0, 211.653)	0.001 (0, 0.003)	0.001 (0, 0.001)
10	85.377 (0.044, 220.774)		
11	87.336 (4.120, 219.729)		
12	87.363 (0.129, 218.385)		
13	87.545 (0.060, 219.549)		
14	87.779 (0.170, 224.636)		

B. Appendix for autoregressive stochastic block model with changes in block membership

In this section 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. For most mice the community to which they belong is clearly identified. Some changepoints are clearly identified whereas others exhibit greater uncertainty. Interesting behaviour to note is that mice 27, 33 and 34 have very similar behaviour starting in community 4, which they leave around time 10 to join community 6, possibly via community 2. Mouse 97 moves between communities 1 and 3 and the "absentee" group, community 6.

Figure B.1.: Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1

Figure B.2.: Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1

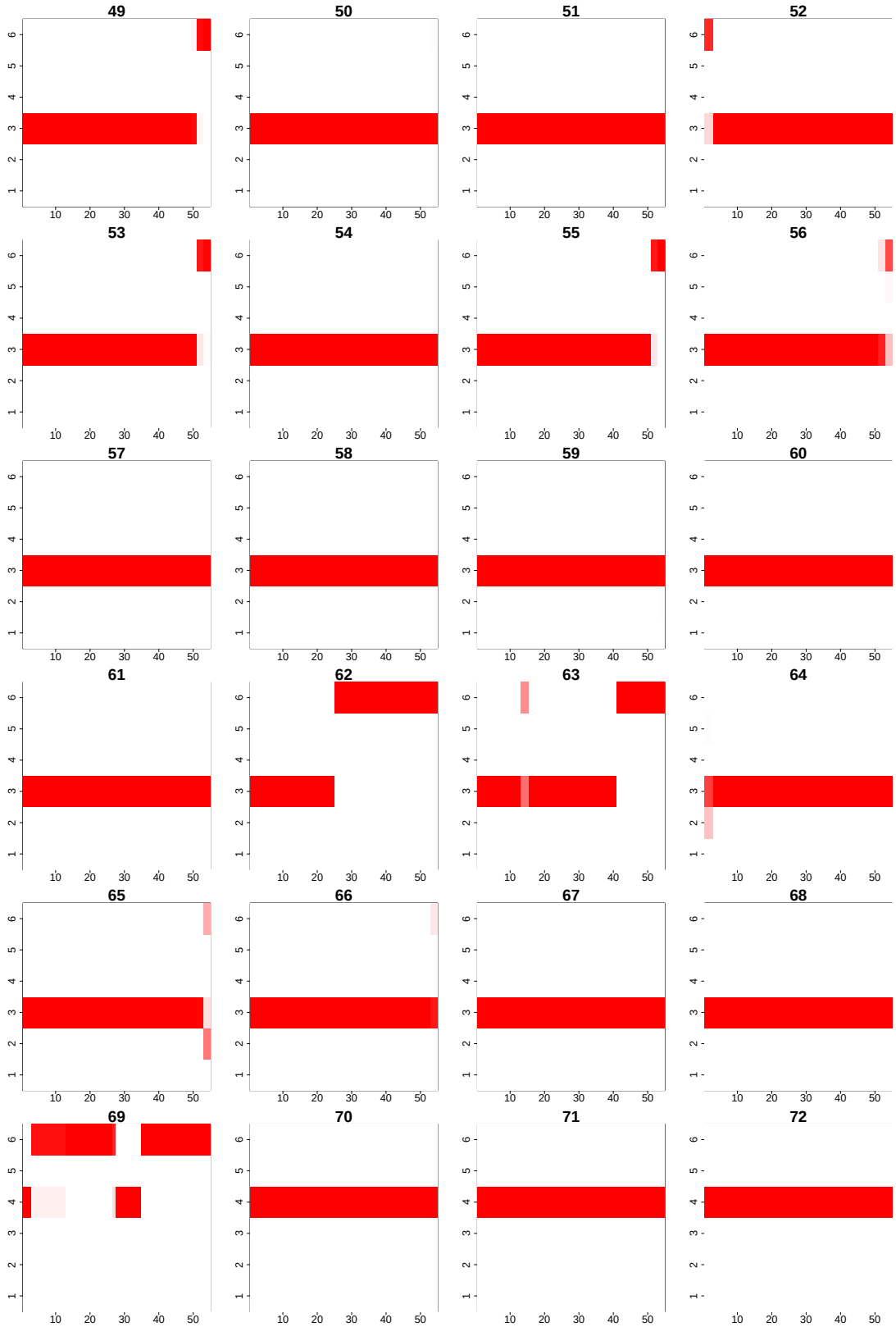


Figure B.3.: Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1

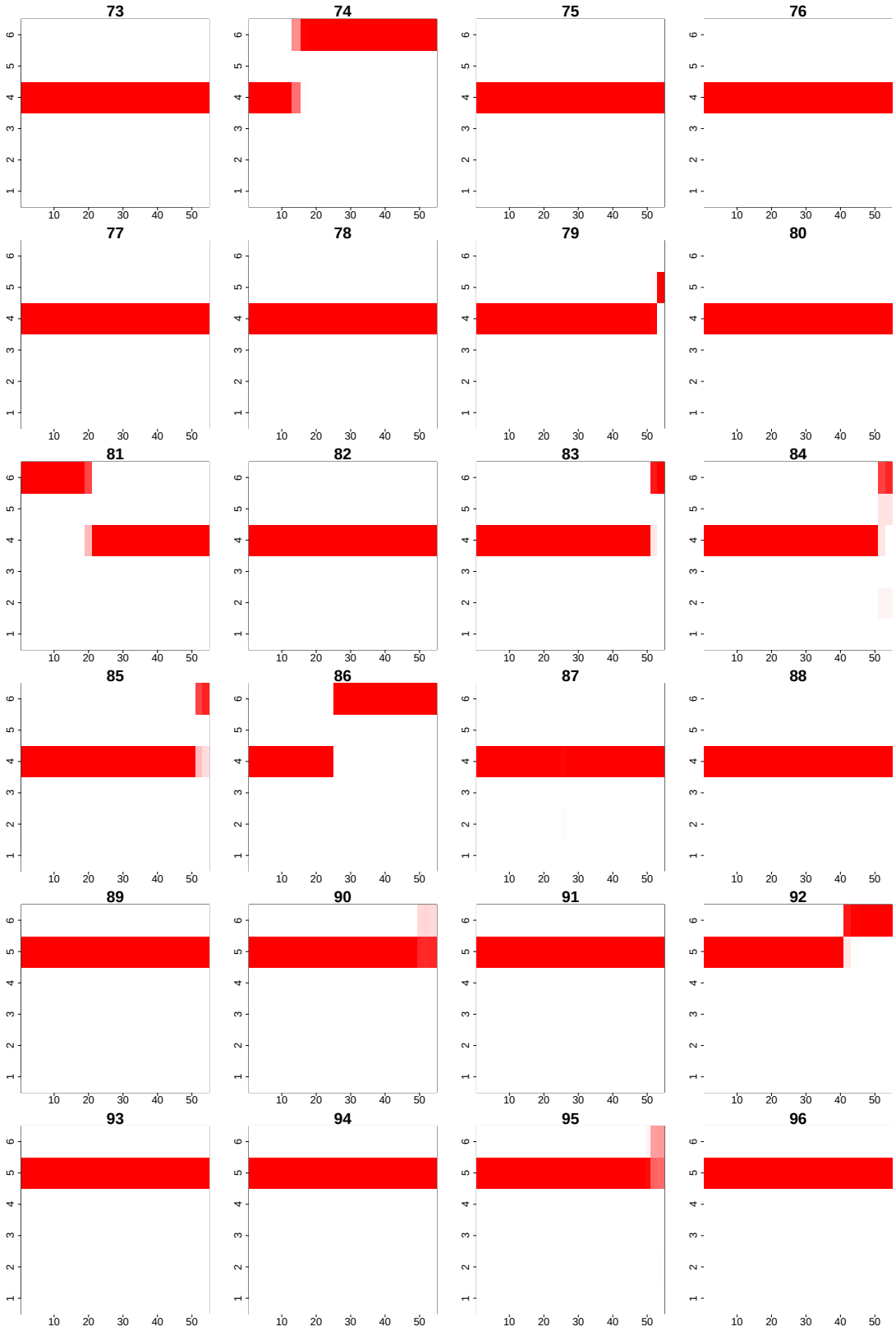


Figure B.4.: Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1

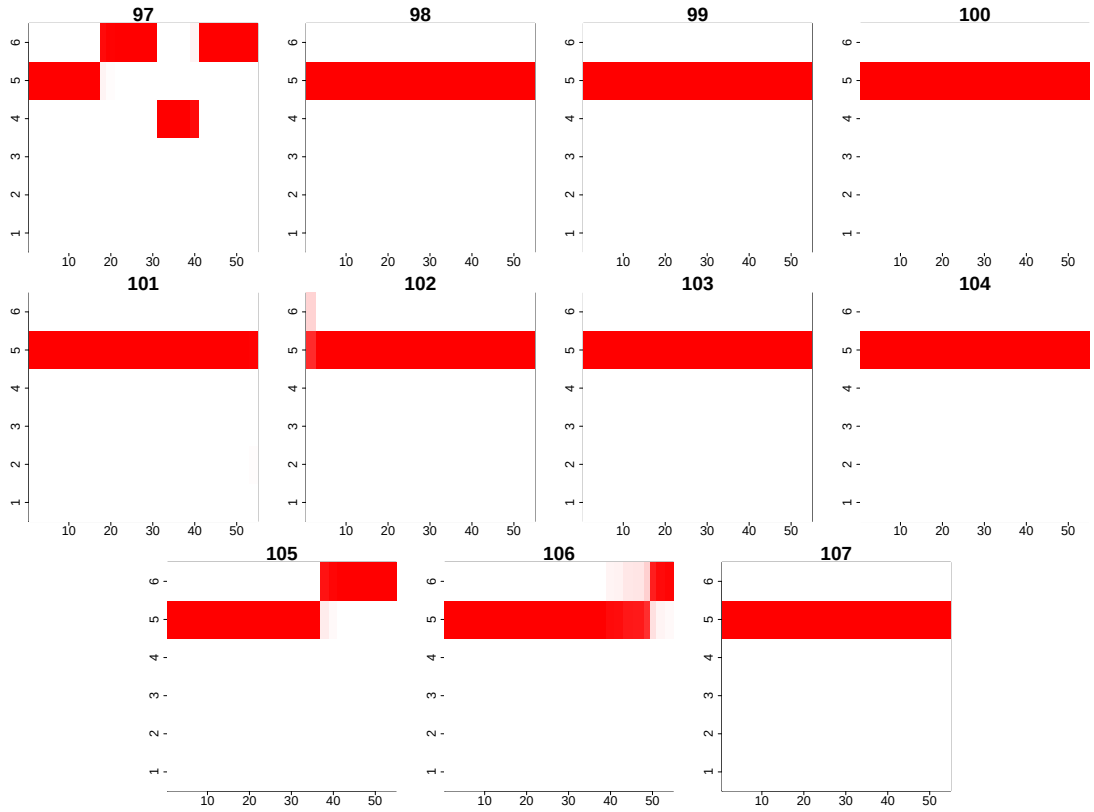


Figure B.5.: Posterior density for each mouse's community membership against time. Shading implies levels of probability: White=0, Red=1

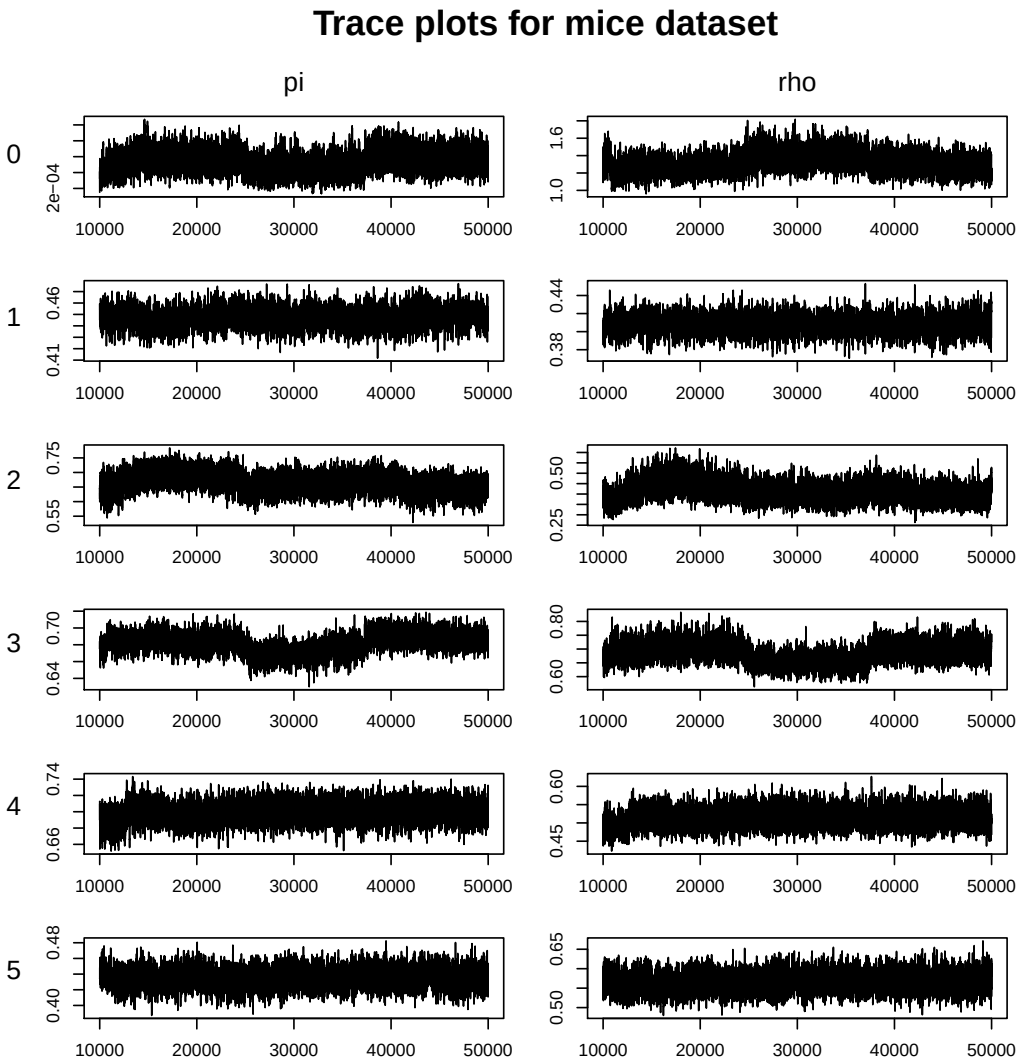


Figure B.6.: Trace plots for $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$ parameters for mice data set.

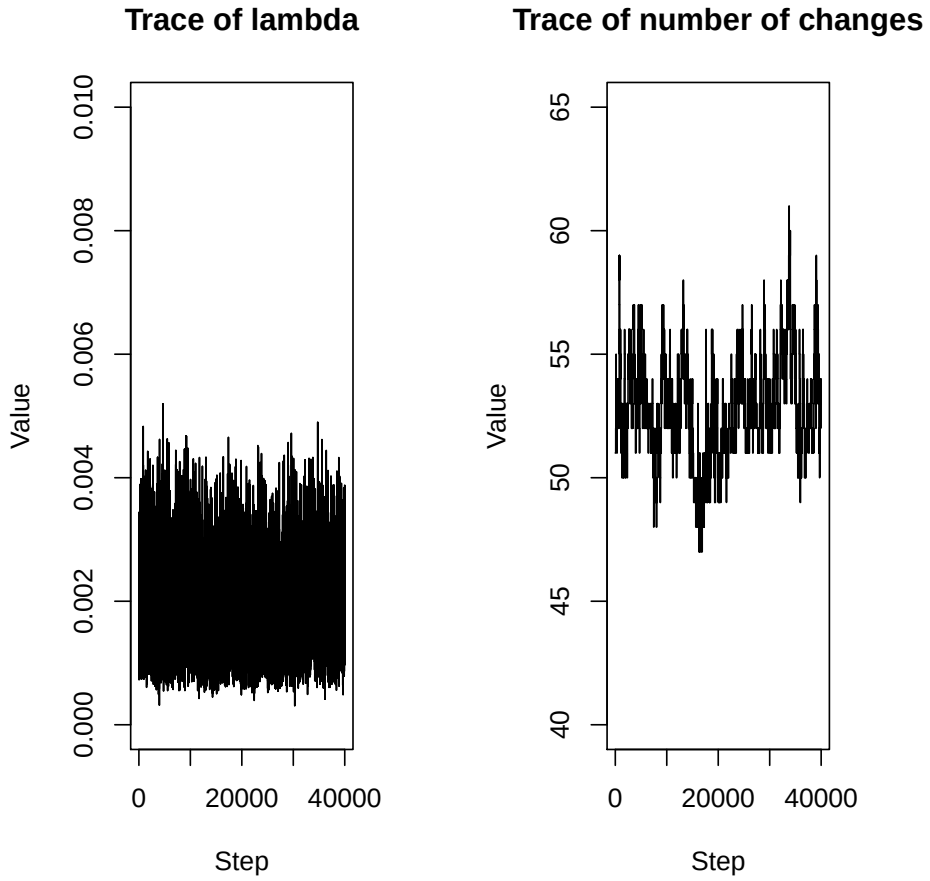


Figure B.7.: Trace plots for λ and number of change-points for mice data set.

C. Appendix for monitoring block membership in the autoregressive stochastic block model

Trace plots for the parameters in the mice data set are shown in the following figures. Note that the estimates a not stationary, calling into question the assumption that the parameters are fixed through time.

C.1. Posterior plots

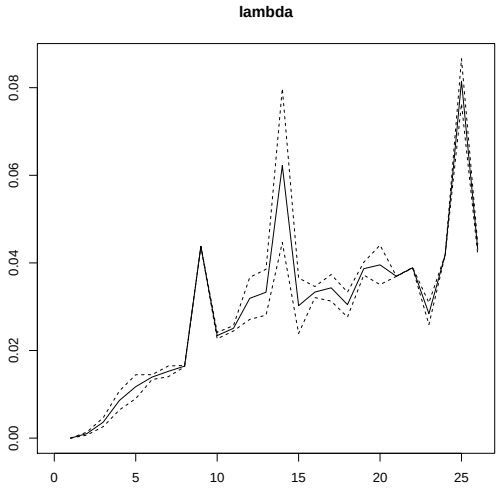


Figure C.1.: Posterior mean for λ at each time point in the mice network.

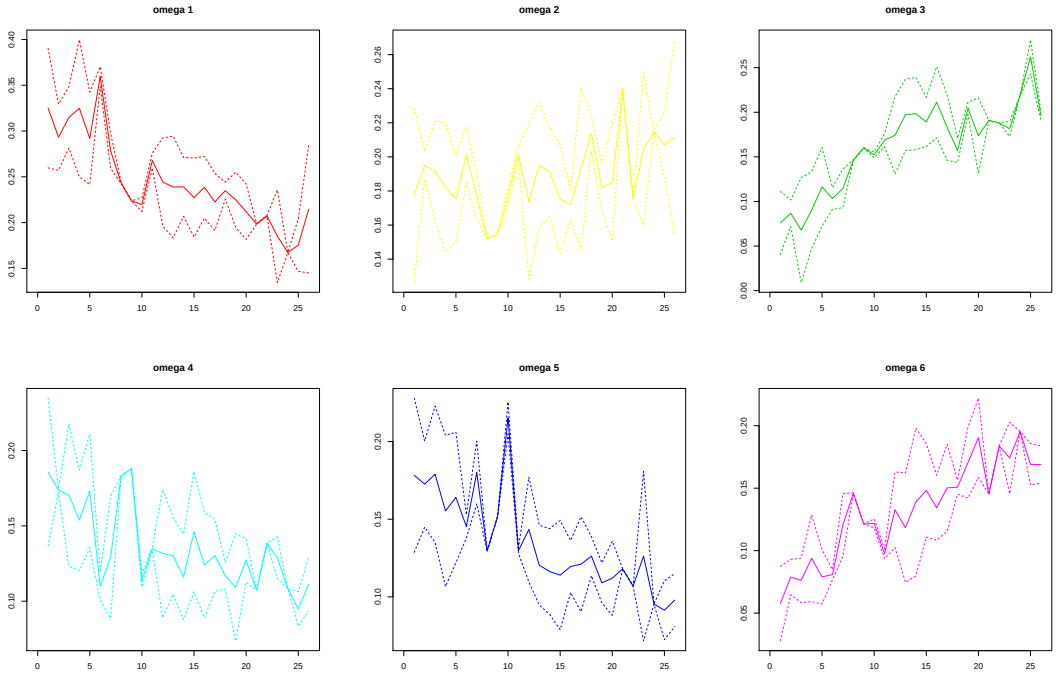


Figure C.2.: Posterior mean for $\boldsymbol{\omega}$ at each time point in the mice network.

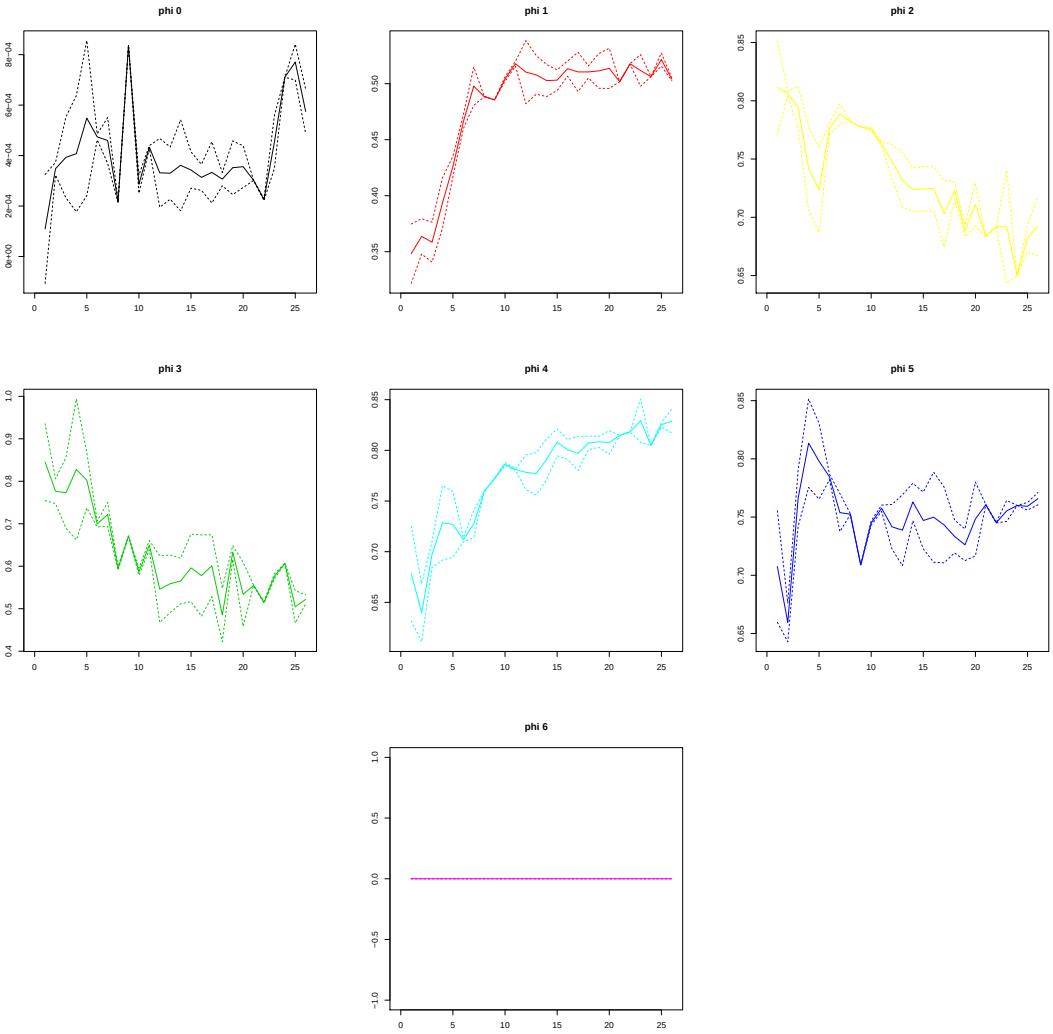


Figure C.3.: Posterior mean for $\boldsymbol{\phi}$ at each time point in the mice network.

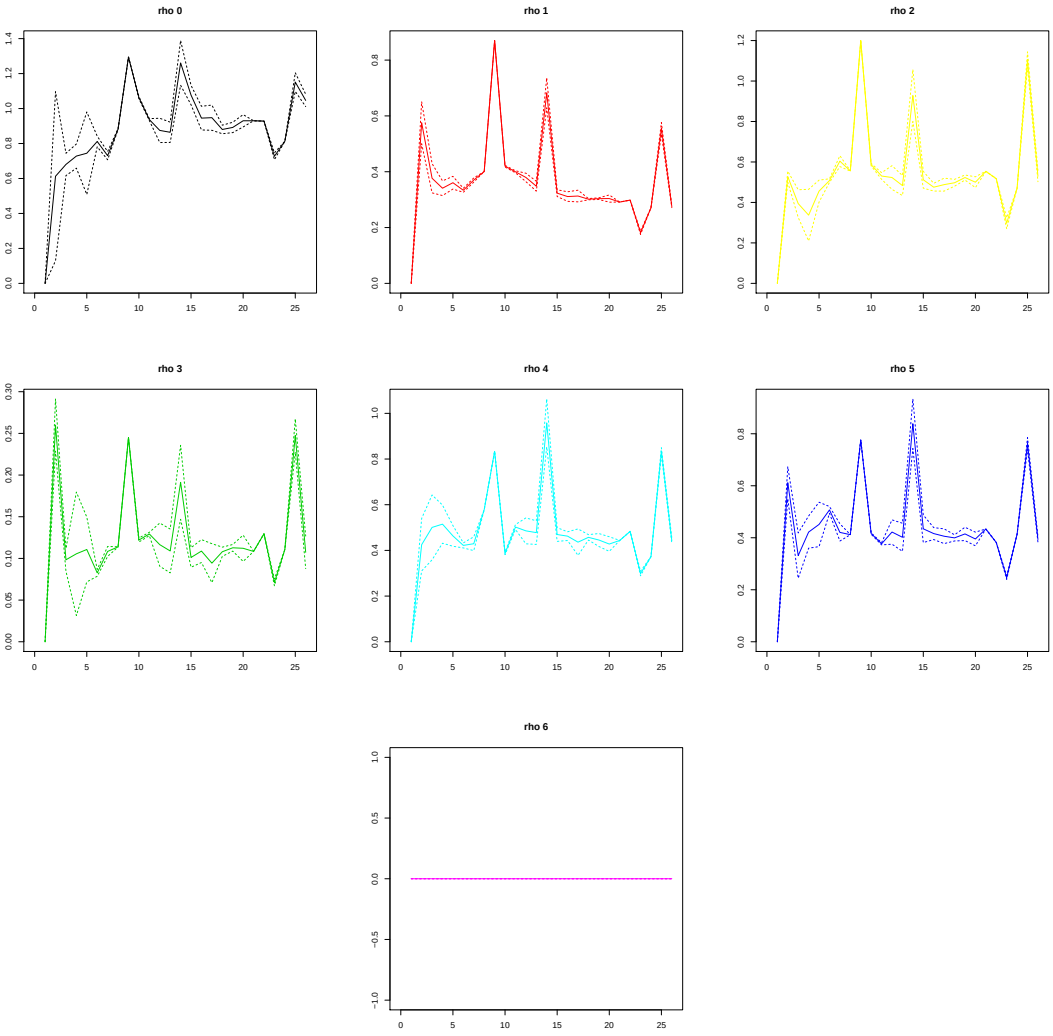


Figure C.4.: Posterior mean for $\boldsymbol{\rho}$ at each time point in the mice network.

C.2. Trace of block membership per mouse

This section contains traces of the posterior block membership for each mouse through time. This shows the uncertainty in the block membership estimates. Compared to the RJMCMC implementation in Ludkin et al. (2017), the SMC algorithm is underestimating uncertainty.

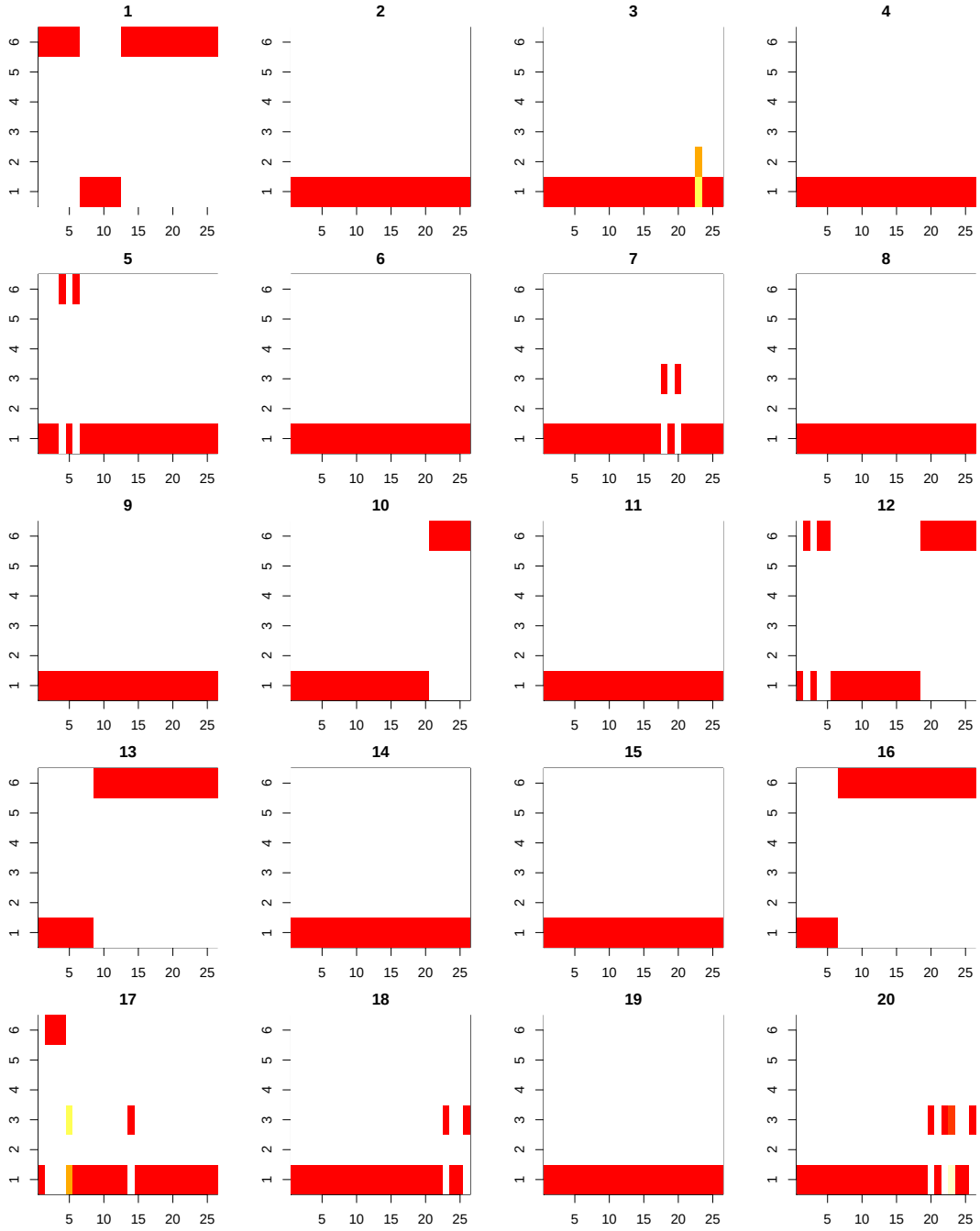


Figure C.5.: Traces of block membership in mouse network.

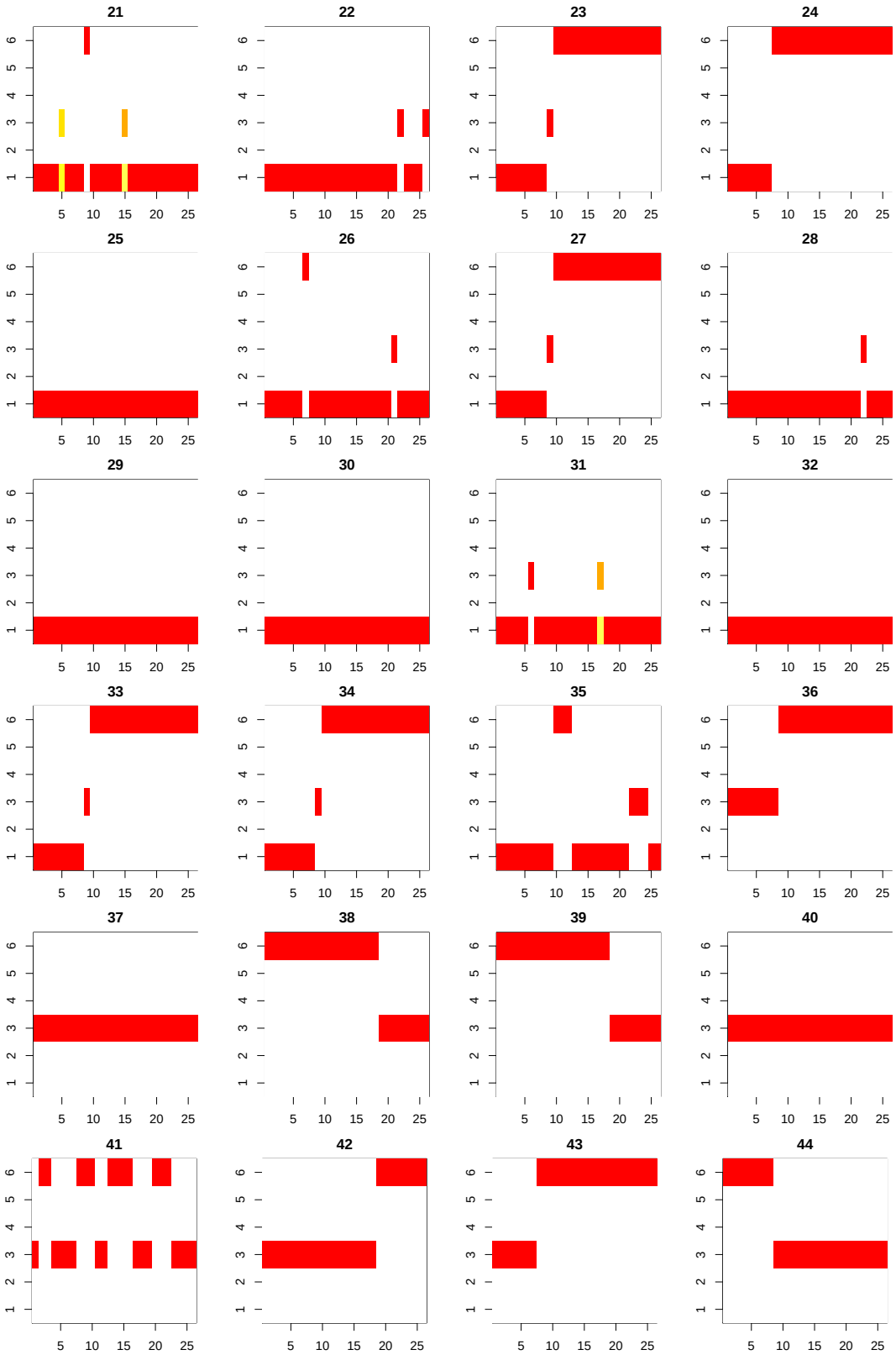


Figure C.6.: Traces of block membership in mouse network.

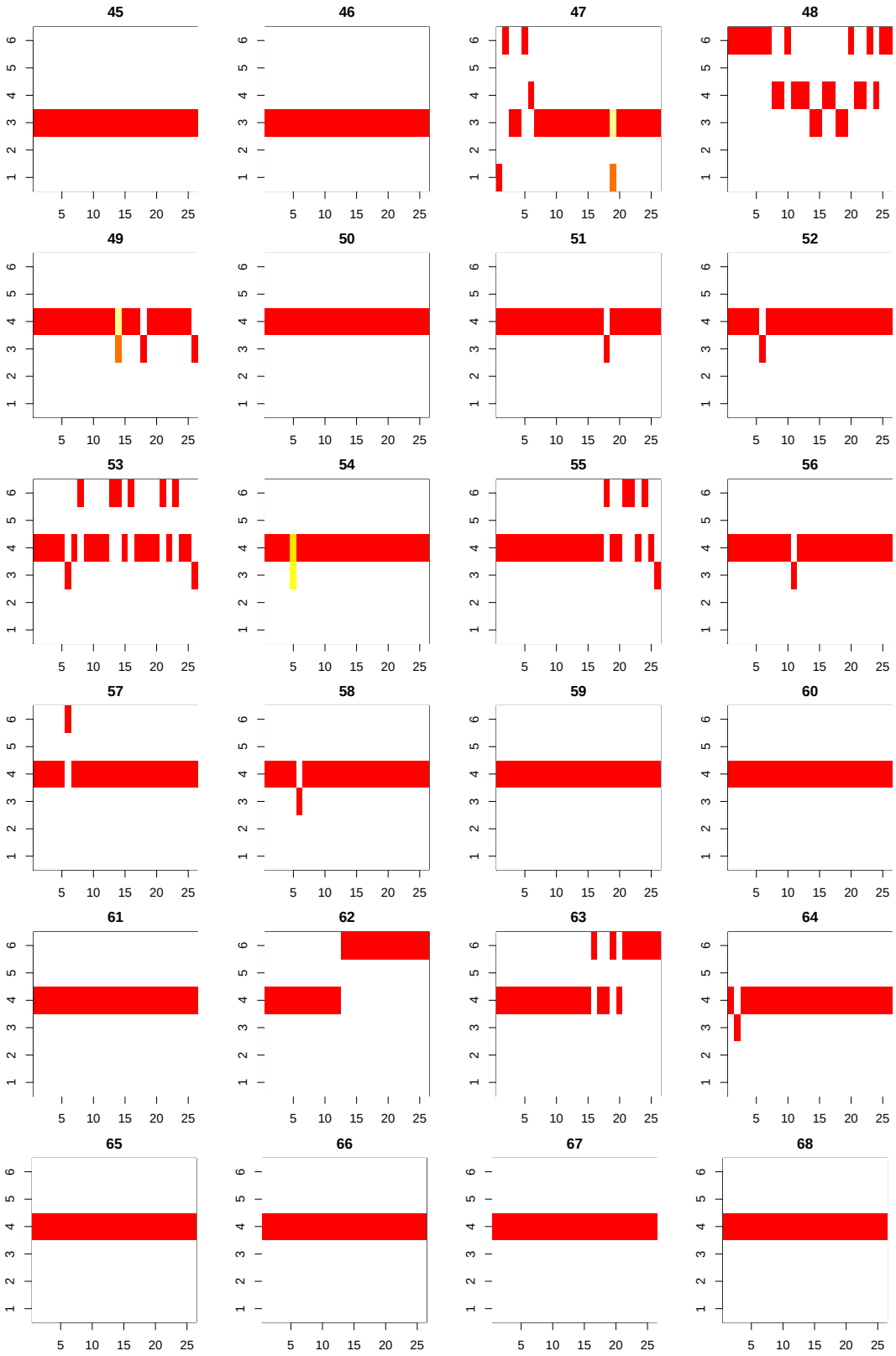


Figure C.7.: Traces of block membership in mouse network.

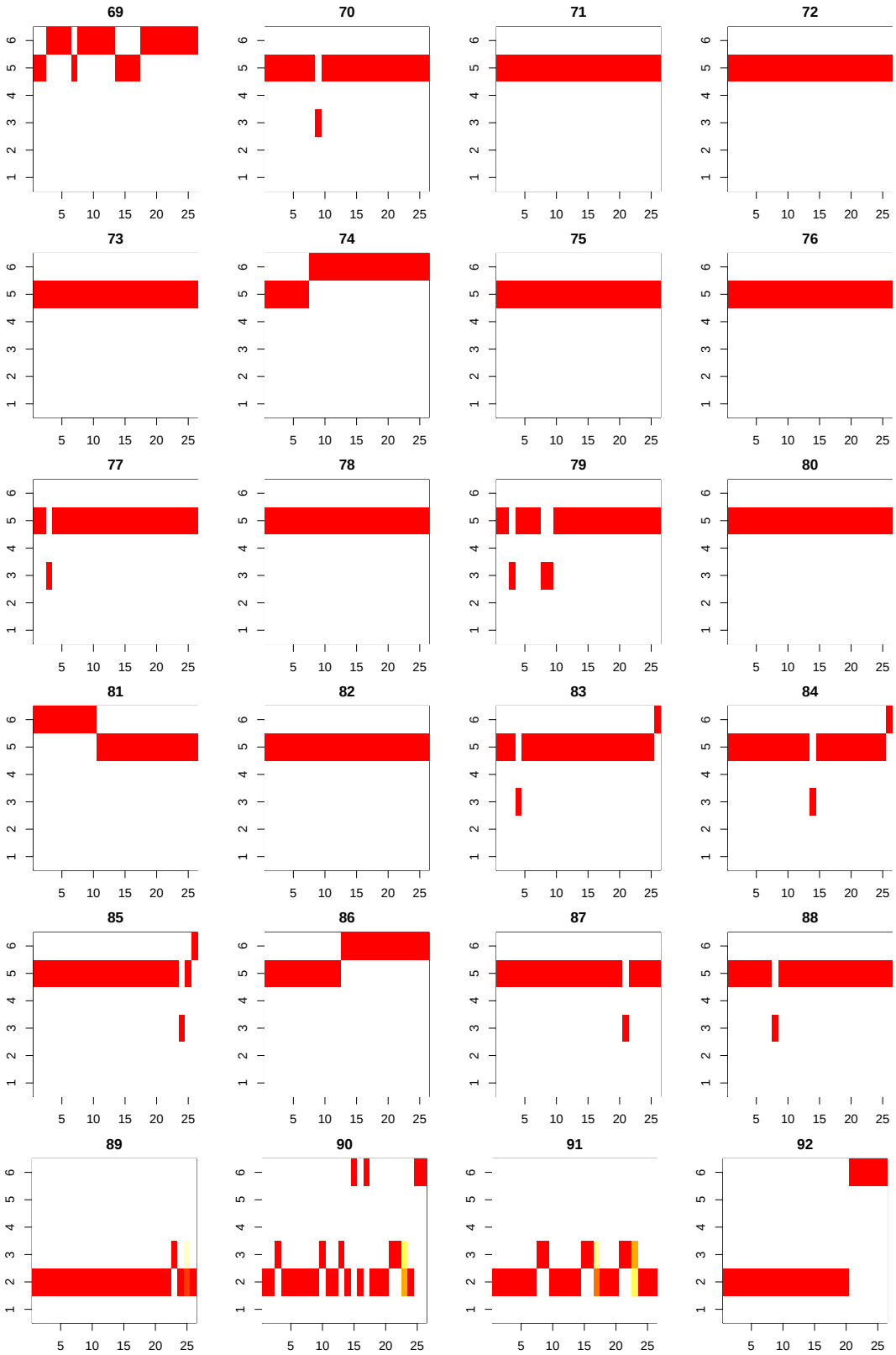


Figure C.8.: Traces of block membership in mouse network.

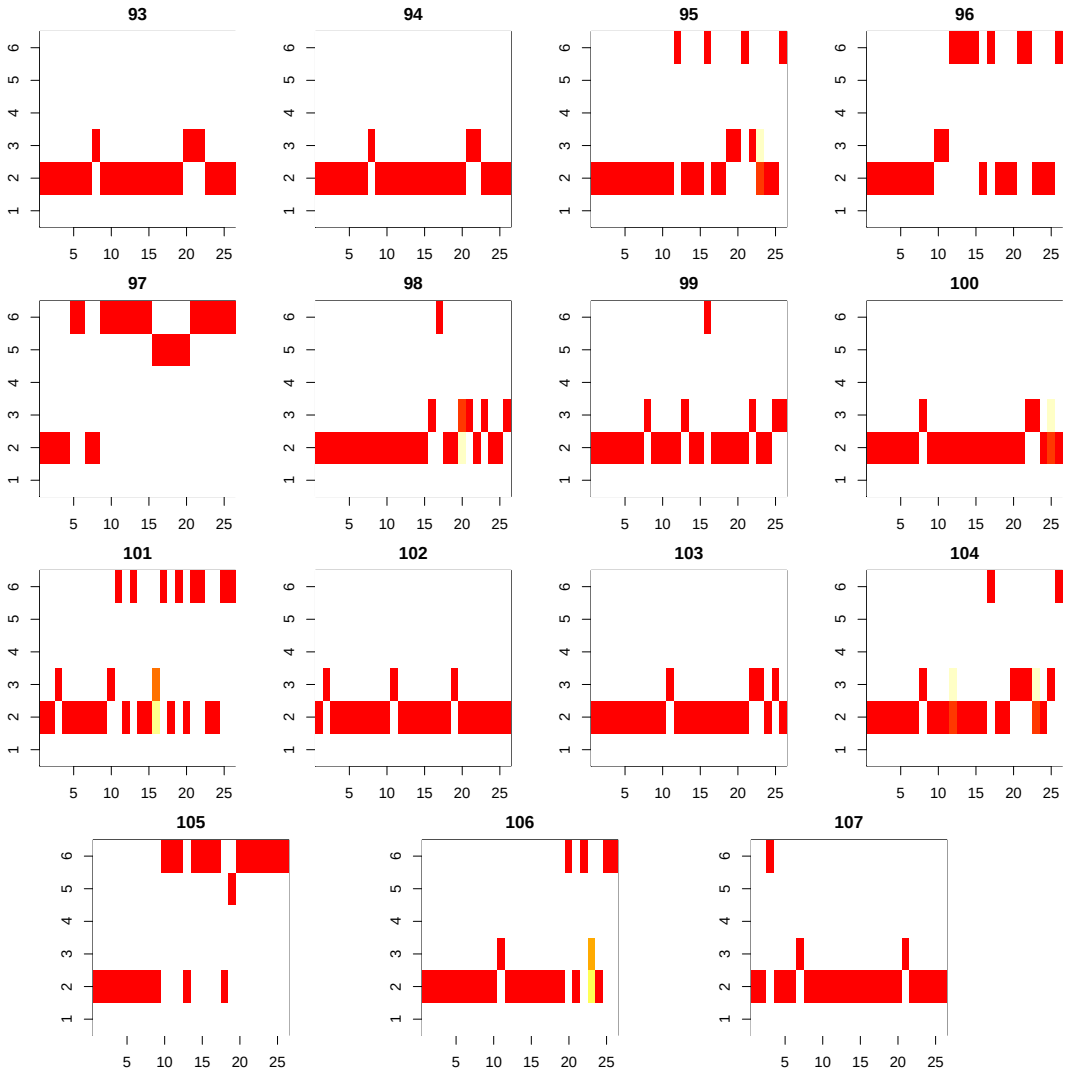


Figure C.9.: Traces of block membership in mouse network.

Bibliography

Airoldi, E., Blei, D., Xing, E., and Fienberg, S. (2005). A latent mixed membership model for relational data. In *Proceedings of the 3rd international workshop on Link discovery*, pages 82–89. ACM.

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2008). Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014.

Albert, R. and Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47.

Aldous, D. J. (1985). Exchangeability and related topics. In *Lecture Notes in Mathematics*, pages 1–198. Springer Berlin Heidelberg.

Allman, E., Matias, C., and 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, E. M., Cocchi, D., and Illian, J. B. (2015). A changepoint analysis of spatio-temporal point processes. *Spatial Statistics*, 14:197–207.

Ambroise, C. and Matias, C. (2012). New consistent and asymptotically normal parameter estimates for random-graph mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(1):3–35.

Anderson, C. J., Wasserman, S., and Crouch, B. (1999). A p^* primer: logit models for social networks. *Social Networks*, 21(1):37–66.

Anderson, J. R. (1991). The adaptive nature of human categorization. *Psychological Review*, 98(3):409.

Antoniak, C. E. (1974). Mixtures of dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics*, pages 1152–1174.

Bardwell, L., Eckley, I., Fearnhead, P., Smith, S., and Spott, M. (2016). Most recent changepoint detection in panel data. *ArXiv e-prints*.

Bhamidi, S., Bresler, G., and Sly, A. (2011). Mixing time of exponential random graphs. *The Annals of Applied Probability*, 21(6):2146–2170.

Bickel, P. J. and Chen, A. (2009). A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Blundell, C., Beck, J., and Heller, K. A. (2012). Modelling reciprocating relationships with hawkes processes. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 2600–2608. Curran Associates, Inc.

Bollobás, B. (1998). Random graphs. In *Modern Graph Theory*, pages 215–252. Springer New York.

Broderick, T., Jordan, M. I., Pitman, J., et al. (2013). Cluster and feature modeling from combinatorial stochastic processes. *Statistical Science*, 28(3):289–312.

Butts, C. T. (2008). A relational event framework for social action. *Sociological Methodology*, 38(1):155–200.

Cha, Y. and Cho, J. (2012). Social-network analysis using topic models. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 565–574. ACM.

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

Chen, K. and Lei, J. (2016). Network cross-validation for determining the number of communities in network data. *Journal of the American Statistical Association*, pages 1–11.

Cho, Y.-S., Galstyan, A., Brantingham, P. J., and Tita, G. (2013). Latent self-exciting point process model for spatial-temporal networks. *arXiv preprint arXiv:1302.2671*.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2012). SMC2: an efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):397–426.

Clauset, A. (2005). Finding local community structure in networks. *Physical review E*, 72(2):026132.

Copic, J., Jackson, M. O., and Kirman, A. (2009). Identifying community structures from network data via maximum likelihood methods. *The BE Journal of Theoretical Economics*, 9(1).

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

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

Davis, R. A., Lee, T. C. M., and Rodriguez-Yam, G. A. (2006). Structural break

estimation for nonstationary time series models. *Journal of the American Statistical Association*, 101(473):223–239.

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

El-Helw, I., Hofman, R., Li, W., Ahn, S., Welling, M., and Bal, H. (2016). Scalable overlapping community detection. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, pages 1463–1472. IEEE.

Erdos, P. and Rényi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60.

Escobar, M. D. and West, M. (1995). Bayesian density estimation and inference using mixtures. *Journal of the american statistical association*, 90(430):577–588.

Fearnhead, P. (2002). Markov chain monte carlo, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862.

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

Fienberg, S. E., Meyer, M. M., and Wasserman, S. S. (1985). Statistical analysis of multiple sociometric relations. *Journal of the american Statistical association*, 80(389):51–67.

Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3):75–174.

Fox, E. W., Short, M. B., Schoenberg, F. P., Coronges, K. D., and Bertozzi, A. L. (2016). Modeling e-mail networks and inferring leadership using self-exciting

point processes. *Journal of the American Statistical Association*, 111(514):564–584.

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

Frank, O. and 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., and Xing, E. P. (2009). Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th annual international conference on machine learning*, pages 329–336. ACM.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statist. Sci.*, 7(4):457–472.

Gershman, S. J. and Blei, D. M. (2012). A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1 – 12.

Ghahramani, Z. and Griffiths, T. L. (2006). Infinite latent feature models and the indian buffet process. In *Advances in neural information processing systems*, pages 475–482.

Gilbert, E. N. (1959). Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144.

Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826.

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

Green, P. J. and Richardson, S. (2001). Modelling heterogeneity with and without the dirichlet process. *Scandinavian Journal of Statistics*, 28(2):355–375.

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

Handcock, M. S., Raftery, A. E., and Tantrum, J. M. (2007). Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354.

Handcock, M. S., Robins, G., Snijders, T. A., Moody, J., and Besag, J. (2003). Assessing degeneracy in statistical models of social networks. Technical report, Citeseer.

Hastings, M. B. (2006). Community detection as an inference problem. *Physical Review E*, 74(3):035102.

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

Hoff, P. (2008a). Modeling homophily and stochastic equivalence in symmetric relational data. In *Advances in neural information processing systems*, pages 657–664.

Hoff, P. D. (2008b). Multiplicative latent factor models for description and prediction of social networks. *Computational and Mathematical Organization Theory*, 15(4):261–272.

Hoff, P. D., Raftery, A. E., and Handcock, M. S. (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098.

Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137.

Holme, P. (2015). Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234.

Hunter, D. R. and Handcock, M. S. (2006). Inference in curved exponential family models for networks. *Journal of Computational and Graphical Statistics*, 15(3):565–583.

Jiang, Q., Zhang, Y., and Sun, M. (2009). Community detection on weighted networks: A variational bayesian method. In *Asian Conference on Machine Learning*, pages 176–190. Springer.

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

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5.

Killick, R., Fearnhead, P., and Eckley, I. E. (2012). Optimal detection of changepoints with a linear computational cost. *J. Amer. Statist. Assoc.*, 107(500):1590–1598.

Klimt, B. and Yang, Y. (2004). *Machine Learning: ECML 2004: 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004. Proceedings*, chapter The Enron Corpus: A New Dataset for Email Classification Research, pages 217–226. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kolaczyk, E. D. (2009). *Statistical Analysis of Network Data: Methods and Models*. Springer New York.

Krivitsky, P. N., Handcock, M. S., Raftery, A. E., and Hoff, P. D. (2009). Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social networks*, 31(3):204–213.

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

Lei, J. (2016). A goodness-of-fit test for stochastic block models. *The Annals of Statistics*, 44(1):401–424.

Lei, J., Rinaldo, A., et al. (2015). Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237.

Li, W., Ahn, S., and Welling, M. (2016). Scalable mcmc for mixed membership stochastic blockmodels. In *Artificial Intelligence and Statistics*, pages 723–731.

Linderman, S. and Adams, R. (2014). Discovering latent network structure in point process data. In *International Conference on Machine Learning*, pages 1413–1421.

Liu, J. and West, M. (2001). Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo Methods in Practice*, pages 197–223. Springer New York.

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

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

Lopes, P. C., Block, P., and König, B. (2016b). Infection-induced behavioural changes reduce connectivity and the potential for disease spread in wild mice contact networks. *Scientific Reports*, 6:31790.

Ludkin, M., Eckley, I., and Neal, P. (2017). Dynamic stochastic block models: parameter estimation and detection of changes in community structure. *Statistics and Computing*.

Mariadassou, M., Robin, S., and Vacher, C. (2010). Uncovering latent structure in valued graphs: a variational approach. *The Annals of Applied Statistics*, pages 715–742.

Matias, C. and Miele, V. (2017). Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(4):1119–1141.

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

Matias, C. and Robin, S. (2014). Modeling heterogeneity in random graphs through latent space models: a selective review. *ESAIM: Proc.*, 47:55–74.

Matteson, D. S. and James, N. A. (2014). A nonparametric approach for multiple change point analysis of multivariate data. *Journal of the American Statistical Association*, 109(505):334–345.

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

Mørup, M. and Schmidt, M. N. (2012). Bayesian community detection. *Neural computation*, 24(9):2434–2456.

Mørup, M. and Schmidt, M. N. (2013). Nonparametric bayesian modeling of complex networks: an introduction. *IEEE Signal Processing Magazine*, 30(3):110–128.

Mørup, M., Schmidt, M. N., and Hansen, L. K. (2011). Infinite multiple membership relational modeling for complex networks. In *Machine Learning for Signal Processing (MLSP), 2011 IEEE International Workshop on*, pages 1–6. IEEE.

Neal, R. M. (2000). Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265.

Négyessy, L., Nepusz, T., Kocsis, L., and Bazsó, F. (2006a). Prediction of the main cortical areas and connections involved in the tactile function of the visual cortex by network analysis. *European Journal of Neuroscience*, 23(7):1919–1930.

Négyessy, L., Nepusz, T., Kocsis, L., and Bazsó, F. (2006b). Prediction of the main cortical areas and connections involved in the tactile function of the visual cortex by network analysis. *European Journal of Neuroscience*, 23(7):1919–1930.

Newman, M. E. (2004a). Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):321–330.

Newman, M. E. J. (2001). The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409.

Newman, M. E. J. (2004b). Coauthorship networks and patterns of scientific collaboration. *Proceedings of the National Academy of Sciences*, 101(Supplement 1):5200–5205.

Newman, M. E. J. (2016). Equivalence between modularity optimization and maximum likelihood methods for community detection. *Physical Review E*, 94(5).

Newman, M. E. J. and Reinert, G. (2016). Estimating the number of communities in a network. *Physical Review Letters*, 117(7).

Nobile, A. and Fearnside, A. T. (2007). Bayesian finite mixtures with an unknown number of components: The allocation sampler. *Statistics and Computing*, 17(2):147–162.

Norris, J. R. (1998). *Markov chains*. Cambridge series on statistical and probabilistic mathematics. Cambridge university press.

Nowicki, K. and Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087.

Peixoto, T. P. (2013). Parsimonious module inference in large networks. *Physical Review Letters*, 110(14).

Perry, P. O. and Wolfe, P. J. (2013). Point process modelling for directed interaction networks. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(5):821–849.

Picard, F., Daudin, J.-J., Koskas, M., Schbath, S., and Robin, S. (2008). Assessing the exceptionality of network motifs. *Journal of Computational Biology*, 15(1):1–20.

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

Rasmussen, C. E. (2000). The infinite gaussian mixture model. In *Advances in neural information processing systems*, pages 554–560.

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

Robins, G., Snijders, T. A. B., Wang, P., Handcock, M., and Pattison, P. (2007).

Recent developments in exponential random graph (p^*) models for social networks. *Social networks*, 29(2):192–215.

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

Saldaña, D. F., Yu, Y., and Feng, Y. (2017). How many communities are there? *Journal of Computational and Graphical Statistics*, 26(1):171–181.

Snijders, T. and van Duijn, M. (1997). Simulation for statistical inference in dynamic network models. In *Simulating social phenomena*, pages 493–512. Springer.

Snijders, T. A., Koskinen, J., and Schweinberger, M. (2010). Maximum likelihood estimation for social network dynamics. *The Annals of Applied Statistics*, 4(2):567.

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

Snijders, T. A. B., Pattison, P. E., Robins, G. L., and Handcock, M. S. (2006). New specifications for exponential random graph models. *Sociological Methodology*, 36(1):99–153.

Storvik, G. (2002). Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on signal Processing*, 50(2):281–289.

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

Vu, D. Q., Hunter, D., Smyth, P., and Asuncion, A. U. (2011). Continuous-time

regression models for longitudinal networks. In *Advances in Neural Information Processing Systems*, pages 2492–2500.

Wang, Y. J. and Wong, G. Y. (1987). Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82(397):8–19.

Wang, Y. R., Bickel, P. J., et al. (2017). Likelihood-based model selection for stochastic block models. *The Annals of Statistics*, 45(2):500–528.

Wasserman, S. (1980a). Analyzing social networks as stochastic processes. *Journal of the American statistical association*, 75(370):280–294.

Wasserman, S. and Anderson, C. (1987). Stochastic a posteriori blockmodels: Construction and assessment. *Social Networks*, 9(1):1–36.

Wasserman, S. and Pattison, P. (1996). Logit models and logistic regressions for social networks: I. an introduction to markov graphs andp. *Psychometrika*, 61(3):401–425.

Wasserman, S. S. (1980b). A stochastic model for directed graphs with transition rates determined by reciprocity. *Sociological methodology*, 11:392–412.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of'small-world'networks. *nature*, 393(6684):440.

Wyse, J. and Friel, N. (2012). Block clustering with collapsed latent block models. *Statistics and Computing*, 22(2):415–428.

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

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

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

Xing, E. P., Fu, W., Song, L., et al. (2010). A state-space mixed membership blockmodel for dynamic network tomography. *The Annals of Applied Statistics*, 4(2):535–566.

Xu, K. S. and Hero, A. O. (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., and Jin, R. (2011). Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine Learning*, 82(2):157–189.

Zhao, Y., Levina, E., and Zhu, J. (2011). Community extraction for social networks. *Proceedings of the National Academy of Sciences*, 108(18):7321–7326.