Joint Bayesian stochastic block modelling and network inference from Ising-like dynamics

xxxxxxx (Candidate number)[a]

[a]Mathematical Institute & Department of Theoretical Physics, University of Oxford, Clarendon Laboratory, Parks Road, Oxford OX1 3PU, United Kingdom

This manuscript was compiled on October 31, 2021

Inferring modular structure via block modeling and its direct generalisation, stochastic block modeling, is among the most prominent approaches to the paradigmatic question of the inference of modular structure in network science. Another central problem in the theory of complex networks is the inference of networks from empirical data, where again the inference from observed network dynamics, for example from Ising-like models, is especially prominent. Bayesian stochastic blockmodelling forms a flexible framework, that provides a method to jointly solve both inference problems, which has been shown to be more effective than a separated approach (1). We review the general Bayesian stochastic blockmodeling framework, its generalisation to joint blockmodelling and network inference and the concrete realisation for an Ising-like 3-Potts model dynamic process. Furthermore we apply the method to a dataset of roll call votes of the German parliament *Bundestag* and discuss the results.

Stochastic Block Model | Bayesian Inference | Ising | Dynamics

Most complex networks of interest are not entirely homogeneous, but contain structures of different kinds and scales (2). Defining and determining these modular 'building blocks' of complex networks is one of the main paradigms of network science. One particularly flexible approach to grouping nodes into blocks consists of the idea to group nodes with identical patterns of connections to other nodes. To make this concept precise the definitions of structural equivalence - where nodes are grouped if they have the exact same neighbours - and regular equivalence - where nodes are grouped if their neighbours share the exact same groups - have been defined (3). The grouping according to these equivalences is generally called blockmodeling, which has an immediate probabilistic generalisation in the concept of stochastic block models (SBMs), where two nodes are grouped if and only if their probability of connection to any other node is the same and only depends on the group of the respective other node. Moreover, the idea that the modular structure and the dynamics on a network are intrinsically linked is a very powerful one (2) and has gained ever more research momentum in recent times.

Furthermore, many empirical complex networks are not known directly, as their full determination is, for example, too expensive or experimentally impractical. It is therefore another paradigmatic domain of network science to infer networks from empirical data, especially from observations of dynamical processes. One could imagine, for example, a social network, where the relations between persons are initially not known, but only the dynamics of an epidemic spreading are observed (for example, because only infected persons report their contacts to a public health department) and it is therefore necessary to reconstruct the network from these observations only. The physics-inspired Ising model, and closely related models like the Potts model, where each node gets assigned a 'spin' value and the probability ensemble takes the form of a Gibbs distribution with the energy being the sum of the nearest neighbour spin-product sum for all nodes, is an interesting model for different situations, such as for example voting behaviour.

Bayesian inference is a comprehensive and well-principled framework for a large domain of problems in complex networks. In particular it provides a method to simultaneously infer the network structure and its block model from empirical data, e.g. from dynamical processes such as an Ising model.

There is a considerable corpus of research on the problem of inferring only the network itself from empirical dynamics, as well as only the SBM from the network or the dynamics in the networks literature. While it would be possible to infer the network and the SBM separately from the observations of dynamics, or first infer the network and then the SBM from the network, the combined Bayesian approach however has been shown to allow for synergistic effects and improve the inference quality in comparison (1).

To understand this improved blockmodelling and inference approach from a theoretical and numerical perspective, we first review the general framework of Bayesian stochastic blockmodeling in Sec. 1, before describing its application to the method of joined network and block model inference from dynamical processes in Sec. 1 with the concrete example of the 3-Potts-model discussed in Sec. C. We then go on to apply this method numerically to a dataset of roll call votes in the German federal parliament, the *Bundestag*, and discuss the properties of the empirical data as well as modelling issues in Sec. 3.

Significance Statement

It is a central paradigm in network science to break down networks into simpler building blocks. One approach is to form groups of nodes with equivalent stochastical properties regarding the rest of the network - so called stochastic block models. Moreover it is a common problem to infer the network itself from empirical data of dynamics of the network. In the Bayesian inference framework these problems can be solved simultaneously to enable synergistic effects. We review the general Bayesian framework and its particular application to joint block modelling and network inference for a specific kind of dynamic to provide a consolidated introduction to the technique. We then apply the method numerically to an empirical parliamentary voting dataset and analyse the results.

The author declares no competing interest.

1. Bayesian stochastic blockmodeling

We will first describe how SBMs and their inference are defined in the Bayesian framework. We then discuss an important variant of conventional SBMs and review efficient inference algorithms.

A. SBMs as generative models. We decompose the network into B blocks, by associating the N vertices with a vector $\boldsymbol{b}$, where $b_i \in \{1, \ldots, B\}$. As a SBM groups all nodes into block in such a way, that the probability of an edge depends only on the group of the respective nodes, we can think of a SBM for a graph without edge weights and without self-loops as a particular generative model of the form

$$P(\boldsymbol{A} \mid \boldsymbol{p}, \boldsymbol{b}) = \prod_{i<j} p_{b_i,b_j}^{A_{ij}} \left(1 - p_{b_i,b_j}\right)^{1-A_{ij}}, \qquad [1]$$

where $\boldsymbol{A} \in \{0,1\}^{N \times N}$ is the network represented by its adjacency matrix and $p_{r,s}$ is the probability of an edge connecting any two nodes in blocks r and s (4). That is, in contrast to seeing the SBM as being determined a posteriori via an optimisation problem for a given network, we think of a SBM as a stochastic process generating networks with a certain probability.

We follow Peixoto (4) in arguing that $P(\boldsymbol{A} \mid \boldsymbol{p}, \boldsymbol{b})$ can be replaced by the easier to handle

$$P(\boldsymbol{A} \mid \boldsymbol{\lambda}, \boldsymbol{b}) = \prod_{i<j} \frac{e^{-\lambda_{b_i,b_j}} \lambda_{b_i,b_j}^{A_{ij}}}{A_{ij}!} \prod_i \frac{e^{-\lambda_{b_i,b_i}/2} \left(\lambda_{b_i,b_i}/2\right)^{A_{ii}/2}}{\left(\frac{A_{ii}}{2}\right)}, \qquad [2]$$

where the expected number of edges between any two vertices in two blocks $\boldsymbol{\lambda}$ now replaces $\boldsymbol{p}$, as the two distributions become asymptotically identical in the sparse limit $p_{r,s} = \mathcal{O}(\frac{1}{N})$.

B. Bayesian inference: SBM. However, we usually don't want to generate a network from a given SBM, but instead are given a particular network and care about its modular structure. In this setting, we assume the network was generated by a SBM and ask about which probability distribution the different possible block assignments $\boldsymbol{b}$ have; i.e. we want to determine $P(\boldsymbol{b} \mid \boldsymbol{A})$. By applying Bayes's rule

$$P(\boldsymbol{b} \mid \boldsymbol{A}) = \frac{P(\boldsymbol{A} \mid \boldsymbol{b}) P(\boldsymbol{b})}{P(\boldsymbol{A})} = \frac{P(\boldsymbol{A} \mid \boldsymbol{b}) P(\boldsymbol{b})}{\sum_{\boldsymbol{b}} P(\boldsymbol{A} \mid \boldsymbol{b}) P(\boldsymbol{b})} \qquad [3]$$

and trivially rewriting

$$P(\boldsymbol{A} \mid \boldsymbol{b}) = \int P(\boldsymbol{A} \mid \boldsymbol{\lambda}, \boldsymbol{b}) P(\boldsymbol{\lambda} \mid \boldsymbol{b}) d\boldsymbol{\lambda} \qquad [4]$$

we can express this quantity - the so called posterior distribution (i.e. 'belief after taking the network into account') - in terms of the prior distributions (i.e. 'assumption about the SBM before the network is considered')) $P(\boldsymbol{b})$ and $P(\boldsymbol{\lambda} \mid \boldsymbol{b})$ as well as an intractable normalisation factor in the denominator, which, however will turn out to be irrelevant (4).

C. Choice of priors. As a general principle from Bayesian statistics, if no prior observations already exists, Jaynes' principle should be applied - i.e. the prior distributions should be chosen in the most uninformative, maximum entropy way. Straightforwardly, one might first choose $P(\boldsymbol{b})$ by ascribing each possible graph partition the equal probability

$$P(\boldsymbol{b}) = \frac{1}{\sum_{B=1}^{N} \left\{ \begin{matrix} N \\ B \end{matrix} \right\} B!},$$

where $\left\{ \begin{matrix} N \\ B \end{matrix} \right\}$ denotes the Stirling numbers of the second kind and counts the number of ways to partition a set of size N into B nonempty sets - this, however gives partitions with small blocks a lot of weight, as $\left\{ \begin{matrix} N \\ B+1 \end{matrix} \right\} >> \left\{ \begin{matrix} N \\ B \end{matrix} \right\}$ (4). This motivates the introduction of hyperpriors - i.e. prior distributions for parameters of the prior distribution -, where we assume a flat prior for the number of blocks in the SBM, and even a hyperhyperprior in form of a flat prior for the sizes of each individual block $\boldsymbol{n}$:

$$P(\boldsymbol{n} \mid B) = \begin{pmatrix} N-1 \\ B-1 \end{pmatrix}^{-1}, \; P(\boldsymbol{b} \mid \boldsymbol{n}) = \frac{\prod_r n_r!}{N!}$$
$$\rightarrow P(\boldsymbol{b}) = P(\boldsymbol{b} \mid \boldsymbol{n}) P(\boldsymbol{n} \mid B) P(B) = \frac{\prod_r n_r!}{N!} \begin{pmatrix} N-1 \\ B-1 \end{pmatrix}^{-1} N^{-1}, \qquad [5]$$

which we use in the following without further hyperpriors as these can be shown to only logarithmically increase the entropy asymptotically (4). For $P(\boldsymbol{\lambda} \mid \boldsymbol{b})$ a similar argumentation, for which we refer to (4), gives

$$P(\boldsymbol{\lambda} \mid \boldsymbol{b}) = \prod_{r \le s} e^{-n_r n_s \lambda_{rs}/(1+\delta_{rs})\bar{\lambda}} n_r n_s / \left(1 + \delta_{rs}\right) \bar{\lambda}, \qquad [6]$$

where we stop the hierarchy of priors with assuming the hyperprior $\bar{\lambda} = 2E/B(B+1)$ (E total number of edges) this time. Combining Eq. (6) and Eq. (5) we finally arrive at an expression for Eq. (4)

$$P(\boldsymbol{A} \mid \boldsymbol{b}) = \frac{\bar{\lambda}^E}{(\bar{\lambda}+1)^{E+B(B+1)/2}} \frac{\prod_{r<s} e_{rs}! \prod_r e_{rr}!!}{\prod_r n_r^{e_r} \prod_{i<j} A_{ij}! \prod_i A_{ii}!!}, \qquad [7]$$

where $e_{rs} = \sum_{ij} A_{ij} \delta_{b_i,r} \delta_{b_j,s}$ is the total number of edges between groups r and s, as well as therefore also at an expression for $P(\boldsymbol{b} \mid \boldsymbol{A})$ up to a normalisation constant (4).

D. Degree correction. In the definition of the SBM we assumed that all nodes in a block are statistically equivalent, however many empirical networks show a different behaviour with a very heterogeneous degree distribution (4). For some applications, it is therefore beneficial to introduce a variant of the SBM, the degree-corrected SBM (DC-SBM) (5). In the DC-SBM the expected degree of node i is controlled by the prior Θ_i and $\lambda_{r,s}$ is generalised to the expected number of edges between two blocks by modifying Eq. (2) as

$$P(\boldsymbol{A} \mid \boldsymbol{\lambda}, \boldsymbol{\theta}, \boldsymbol{b}) = \prod_{i<j} \frac{e^{-\theta_i \theta_j \lambda_{b_i,b_j}} \left(\theta_i \theta_j \lambda_{b_i,b_j}\right)^{A_{ij}}}{A_{ij}!} \times \prod_i \frac{e^{-\theta_i^2 \lambda_{b_i,b_i}/2} \left(\theta_i^2 \lambda_{b_i,b_i}/2\right)^{A_{ii}/2}}{(A_{ii}/2)!} \qquad [8]$$

While this modification might seem ad-hoc at first, it is possible to justify the modification in a well principled manner, by assuming an uninformed prior on the expected degree and following a similar procedure as described earlier; in fact, the formula can be further refined by again introducing a hierarchy of hyperpriors and not only being uninformed about the expected degrees of each node, but also about the frequencies of expected degrees (refer to (5), (4)).

E. Inference using MCMC. At this point we derived the posterior distribution $P(\boldsymbol{b} \mid \boldsymbol{A})$ for the SBM and the DC-SBM, however we are usually not interested in the entire distribution, but only in a few likely or even only the most likely SBM given our network. This can be solved by sampling from or finding the maximum of the posterior probability distribution.

However the distributions involved are too complex to do this directly - in fact it can be shown that this is NP-hard in in general (4). However we can asymptotically exactly approximate the probability distribution via Markov chain Monte Carlo (MCMC) methods, which are an efficient standard approach for problems of this kind (4, 6, 7).

The central idea here is to construct a Markov chain with certain transition probabilities in such a way, that the stationary distribution approximates the posterior distribution. I.e. we start with an arbitrary initial state $\boldsymbol{b}_0$ and then repeatedly transition from state $\boldsymbol{b}$ to a new state $\boldsymbol{b}'$ with a probability $P(\boldsymbol{b}' \mid \boldsymbol{b})$. After sufficiently many steps the Markov chain has reached stationarity and the states $\boldsymbol{b}$ can then be understood as being sampled from the posterior distribution. $P(\boldsymbol{b}' \mid \boldsymbol{b})$ can, in principle, be chosen arbitrarily, if we employ the Metropolis-Hastings criterion (7), i.e. we accept a transition with a probability of

$$\min\left(1, \frac{P\left(\boldsymbol{b}' \mid \boldsymbol{A}\right)}{P\left(\boldsymbol{b} \mid \boldsymbol{A}\right)} \frac{P\left(\boldsymbol{b} \mid \boldsymbol{b}'\right)}{P\left(\boldsymbol{b}' \mid \boldsymbol{b}\right)}\right). \qquad [9]$$

Note that $\frac{P\left(\boldsymbol{b}' \mid \boldsymbol{A}\right)}{P(\boldsymbol{b} \mid \boldsymbol{A})}$ depends only on the uninformed priors we already derived, as the intractable 'normalisation' $P(\boldsymbol{A})$ from Eq. (3) cancels in the fraction.

If instead of sampling, the maximum of the posterior should be found, the standard technique of simulated annealing can be employed. Here, $P(\boldsymbol{b} \mid \boldsymbol{A})$ is modified to $P(\boldsymbol{b} \mid \boldsymbol{A})^\beta$ and β is slowly increased during the optimisation (8).

While the above is true in theory, in practice it might take an unfeasible amount of time to reach the stationary state. It is therefore desirable to not choose the initial state and the move proposals entirely arbitrarily, but as close to likely partitions of the posterior for the former and close to the posterior distribution for the latter. Peixoto (9) therefore proposes two optimisations. First, the transition proposals are made based on the typical neighborhood of nodes in the current block model. Specifically, the moves can be chosen according to the distribution

$$P\left(b_i = r \mid \boldsymbol{b}\right) = \sum_s \left(\sum_j A_{ij}\delta_{b_j,s}/k_i\right) \frac{e_{sr} + \varepsilon}{e_s + \varepsilon(B+1)}, \qquad [10]$$

where $\epsilon > 0$ is a sufficiently small but arbitrary parameter guaranteeing ergodicity, which does not bias the partitions in any way and can be sampled efficiently by an algorithm in $\mathcal{O}(k_i)$ (for details see (4), (9)). Secondly, the starting state can be chosen more effectively by performing a Fibonacci search, in which the best partition is chosen by combining blocks in a partition with more blocks via a heuristic (9). This heuristic consists of attempting the moves of Eq. (10) until no further improvement is observed and then repeatedly merging blocks by treating blocks as nodes and applying Eq. (10) as merge proposals (4).

F. Entropy & Model selection. It is possible to frame the Bayesian approach in an information theoretic way by determining the entropy Σ (also called description length) of the numerator of the posterior distribution via

$$\begin{aligned}\Sigma &= -log_2 P(\boldsymbol{A} \mid \boldsymbol{b})P(\boldsymbol{b}) \\ &= -\log_2 P(\boldsymbol{A}, \boldsymbol{\lambda}, \boldsymbol{b}) \\ &= -\log_2 P(\boldsymbol{A} \mid \boldsymbol{\lambda}, \boldsymbol{b}) - \log_2 P(\boldsymbol{\lambda}, \boldsymbol{b})\end{aligned} \qquad [11]$$

- the maximum posterior probability partition then corresponds to minimum entropy (4). Note that the second term in the last line incorporates a 'penalty' for the complexity of the generative model.

Suppose we are given two generative models g_1, g_2 and optimal partitions $\boldsymbol{b}_1, \boldsymbol{b}_2$, the posterior-odds ratio from Bayesian statistics

$$\Lambda = \frac{P\left(\boldsymbol{A} \mid \boldsymbol{b}_1, g_1\right) P\left(\boldsymbol{b}_1\right)}{P\left(\boldsymbol{A} \mid \boldsymbol{b}_2, g_2\right) P\left(\boldsymbol{b}_2\right)} = 2^{-\Delta\Sigma} \qquad [12]$$

gives a well-principled approach to determining which model fits the data better; $\Lambda > 1$ favours g_1, in practice very large values of the entropy difference can give an indication for model selection (4).

2. Joint network and block model inference from dynamical processes

We now move on to the setting where $\boldsymbol{A}$ is unknown and both the network as well as the SBM need to be determined by the observations of a dynamical process on the network. We first review how the Bayesian framework can be adapted to network reconstruction in general, and then discuss a particular dynamic and its concrete Bayesian implementation.

A. Bayesian network reconstruction. We denote the observations of a dynamical process $\boldsymbol{\mathcal{D}}$. When confronted with such measurement data, we need to assume a dynamical process that is responsible for it, i.e. we assume a generative forward model $P(\boldsymbol{\mathcal{D}} \mid \boldsymbol{A})$ (1). In the general paradigm of the Bayesian framework we now apply Bayes's rule

$$P(\boldsymbol{A} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \boldsymbol{A})P(\boldsymbol{A})}{P(\boldsymbol{\mathcal{D}})} \qquad [13]$$

to be able to integrate the results of the previous section via

$$P(\boldsymbol{A}, \boldsymbol{b} \mid \mathcal{D}) = \frac{P(\mathcal{D} \mid \boldsymbol{A})P(\boldsymbol{A} \mid \boldsymbol{b})P(\boldsymbol{b})}{P(\mathcal{D})}. \qquad [14]$$

This result demonstrates the elegance of the Bayesian framework, as it very seamlessly incorporates both network and SBM inference in a uniform manner. All aspects discussed in the prior section can now be applied to this posterior, i.e. for example maximising the posterior with simulated annealing to find an optimal network structure and SBM regarding to a maximum argument estimation.

Peixoto argues that the DC-SBM is preferable as a prior probability when the network is unknown, as it is more robust for the block model if a number of edges have been erroneously introduced or missed in the inference of the network (1). We will therefore follow Peixoto in focusing on this choice and continue with Eq. (8), only adapted by removing the second term to disallow self-loops in view of the later application.

B. Ising-like 3-Potts model. The 3-Potts model is a particular network dynamic and similar to the standard Ising model, in that each node takes a spin variable $s_i \in \{-1, 0, 1\}^N$ compared to $s_i \in \{-1, 1\}^N$ in the normal Ising model. With a physics-inspired analogy, we can think of each node as being able to enter a new 'neutral state' that does not increase nor lower its nearest-neighbour energy. We however keep the form of the Ising Gibbs-distribution

$$P(\boldsymbol{s} \mid \boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h}) = \frac{\exp\left(\beta \sum_{i<j} J_{ij} A_{ij} s_i s_j + \sum_i h_i s_i\right)}{Z(\boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h})}, \tag{15}$$

where $\beta \in \mathbb{R}$ is the inverse temperature, $J_{ij} \in \mathbb{R}$ is the nearest neighbour coupling between nodes i and j and $h_i \in \mathbb{R}$ is the local symmetry-breaking field (1). The denominator is the partition function, i.e. a normalisation parameter.

C. Posterior distribution for 3-Potts model. Unfortunately the partition function in Eq. (15) is intractable (1), which is why we need to refer to an approximation called the pseudolikelihood approximation (10):

$$P(\boldsymbol{s} \mid \boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h}) \approx \prod_i \frac{\exp\left(\beta s_i \sum_j J_{ij} A_{ij} s_j + h_i s_i\right)}{1 + 2\cosh\left(\beta \sum_j J_{ij} A_{ij} s_j + h_i\right)}. \tag{16}$$

A sample of a set of microstates according to $P\left(\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M\} \mid \boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h}\right) = \prod_l P\left(\boldsymbol{s}_l \mid \boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h}\right)$ yields the posterior distribution

$$P(\boldsymbol{A}, \boldsymbol{b}, \beta, \boldsymbol{J}, \boldsymbol{h} \mid \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M\}) = \frac{P(\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M\} \mid \boldsymbol{A}, \beta, \boldsymbol{J}, \boldsymbol{h}) P(\beta) P(\boldsymbol{h}) P(\boldsymbol{J} \mid \boldsymbol{A}) P(\boldsymbol{A} \mid \boldsymbol{b}) P(\boldsymbol{b})}{P(\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M\})} \tag{17}$$

(1). Following Jaynes' principle we then assume uniform priors $P(\boldsymbol{h}) \propto 1$, $P(\beta) \propto 1$ on the entire domain of $\boldsymbol{h}$ and β respectively; moreover we shift the interval of $J_{i,j}$ to $[-1/2, 1/2]$ without loss of generality, such that $P(\boldsymbol{J} \mid \boldsymbol{A}) = \prod_{ij} J_{ij}^{Aij}$ (1).

D. Inference algorithm. The MCMC algorithm is conceptually similar to the ideas discussed in Sec. E.

Now not only transitions of the partitions $\boldsymbol{b} \rightarrow \boldsymbol{b}'$ but also of the adjacency matrix $\boldsymbol{A} \rightarrow \boldsymbol{A}'$ and all remaining parameters $\boldsymbol{\Theta} \rightarrow \boldsymbol{\Theta}'$ need to be produced and accepted with the Metropolis-Hastings criterion

$$\min\left(1, \frac{P\left(\boldsymbol{A}', \boldsymbol{b}' \mid \mathcal{D}\right) P\left(\boldsymbol{A} \mid \boldsymbol{A}', \boldsymbol{b}'\right) P\left(\boldsymbol{b} \mid \boldsymbol{A}', \boldsymbol{b}'\right)}{P(\boldsymbol{A}, \boldsymbol{b} \mid \mathcal{D}) P\left(\boldsymbol{A}' \mid \boldsymbol{A}, \boldsymbol{b}\right) P\left(\boldsymbol{b}' \mid \boldsymbol{A}, \boldsymbol{b}\right)}\right) \tag{18}$$

; in practice, again, the transitions are chosen in an optimised way, for the transitions of the partitions the discussion in Sec. E applies (1). The network transitions used are

$$P\left(A'_{ij} = A_{ij} + \delta \mid \boldsymbol{G}\right) = \begin{cases} 1/2 & \text{if } G_{ij} > 0 \\ 1 & \text{if } A_{ij} = 0 \text{ and } \delta = 1 \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

Deviation category	# votes
no deviation	962867
excused absence	68954
unexcused absent	67976
strong deviation	18956
weak deviation	7977
no party line because no absolute majority within the party group	386
voting behavior not/wrongly protocolled	148
no party line due to tie within the party group	82
invalid vote	47

Table 1. Voting behaviour of MPs relative to party line

with $\delta = \pm 1$; the entry ij is chosen from a distribution dependent on the current DC-SBM

$$P(i, j \mid \boldsymbol{G}, \boldsymbol{b}) = \frac{k_i + 1}{\sum_l \delta_{b_l, b_i} k_l + 1} \frac{k_j + 1}{\sum_l \delta_{b_l, b_j} k_l + 1} \frac{e_{b_i b_j} + 1}{\sum_{tu} e_{tu} + 1} \tag{20}$$

which causes dense regions to be sampled more often, but still guarantees eventual transitions for every entry (1). With these choices the complexity for N node and E edge transitions is $O(\langle k \rangle N + EM)$ ($\langle k \rangle$ expected degree, M number of observations of the dynamic) (1).

3. Numerical simulation: roll call votes in the *Bundestag*

We now apply the method of jointly inferring network structure and SBM to an empirical dataset with an Ising-model like dynamic, specifically votes in the German federal parliament, the *Bundestag*.

A. Empirical data. The dataset `BTVote` (11) contains all votes cast by members of the German federal parliament in roll call votes (RCVs) (in total 1,127,359 decisions), i.e. votes where MPs are required to vote by name, for the first 17 electoral periods of the parliament (years 1949-2013). The voting behaviour is recorded as yes / no / abstention / absence as well as a number of residual categories. Moreover the dataset contains the party affiliation of the MP at the time of the vote as well as how the individual vote relates to the party line (12). Here it should be noted, that as per the political system there is strong party unity, i.e. MPs seldomly deviate from the party line (12), as can also be seen in Tab. 1. Regarding the block modelling approach, it can therefore be expected that the networks of MPs show a strong modular, community-like structure.

B. Modeling approach. We take a similar approach as presented in (1) for the Brazilian parliament and model each MP as a node in a network, whose edge structure is unknown. We then assume the votes form an Ising-like Potts-model dynamic process, as votes on different topic do likely not influence each other considerably in time. Agreement to a motion is mapped to a spin value of 1, disagreement to a spin value of -1 and all behaviours that signal indifference or do not allow to deduce the position of the MP are mapped to 0.

C. Numerical simulation. We make use of the `graph-tool` Python library (13), which implements implements a Potts-model dynamics Bayesian posterior, Bayesian SBM and efficient numerical MCMC sampling and optimisation methods.

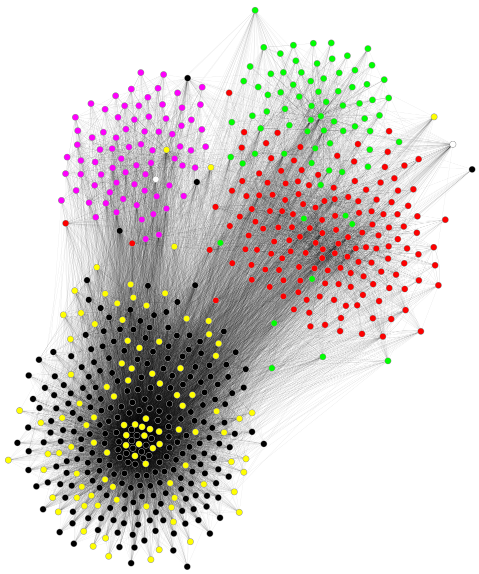


Fig. 1. Inferred SBM for election period 17 - vertex colour corresponds to party affiliation. Opposition: ■ Greens, ■ SPD, ■ Left; Government: ■ Liberals, ■ Conservatives, party edge cases shown in white. Edge width is proportional to the posterior probability of the edge.

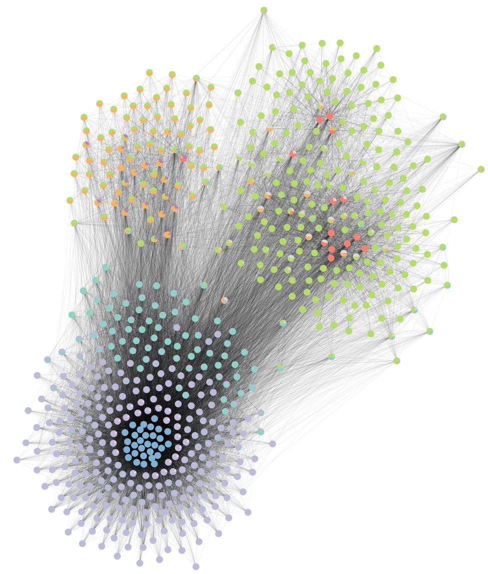


Fig. 2. Inferred SBM for election period 17 - vertex pie chart shows marginal probability of being in inferred SBM block of corresponding colour. Edge width is proportional to the posterior probability of the edge.

We first create a graph with the time series data of the `BTVote` dataset. The procedure then follows the theoretical discussion closely - we create a representation of the priors for a 3-Potts model. We then first carry out 1000 MCMC sweeps to bring the Markov chain closer to stationarity, before starting to record the partition after each MCMC transition proposal iteration. We monitor the entropy during the MCMC sampling and stop as soon as the entropy has not reached a new minimum since a fixed number of steps. We then infer the maximal and marginal probabilities of the partitions from the recorded set. Finally we draw the graphs with a spring layout visualisation.

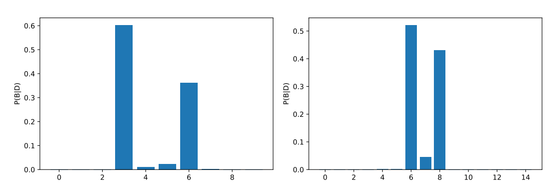
Simulated annealing was also attempted, however proved impractical on the available computing hardware.

For details, see the code listing in the Supplementary Information in Sec. A and the attached full code file.

D. Results & Discussion. Fig. 2 and 5 both show the network of MPs for the 17th election period of the *Bundestag* with the inferred block memberships of the nodes for the SBM and the DC-SBM respectively; more precisely, the color in the pie charts inside the nodes denote the marginal probability of the node being in the block denoted by that respective colour. Both graph layouts are spring layouts with parameters set in such a way, that the communities defined by the inferred maximum probability partition are well separated and can be easily identified. Fig. 1 and 4 show the same networks with the same spring layout respectively; here the nodes are instead colored by the party affiliation of the MP. Fig. 3 shows the posterior probabilities of the number of groups. The entropy difference between the SBM and the DC-SBM is 75965.4.

From Fig. 1 it can be seen, that the SBM clearly divides the opposition and the government parties (compare the color coding in the caption). However from Fig. 2 it becomes clear that there is also a core-periphery structure inferred: each party has a center, compare especially the structure of the two government parties, with a core, a periphery and a group of MPs that are ideologically closer to the opposition by deviating from party policy at times (blue, purple and cyan in Fig. 2). This result compares well with the simulations of the Brazilian parliament performed in (1); a similar structure is found.

In comparison the DC-SBM does not result in such a readily interpretable partition (the partition does not correspond to, e.g. the rate of absence of MPs, it is also not a hierarchy of deviation from the party line, compare Supplementary Information Figs. 6 7). This intuitive impression is supported quantitatively by the entropy difference according to the model selection criterion of Sec. F. This is somewhat surprising, given the argument of Peixoto we followed in Sec. A. However, one might argue that in this particular network most MPs have a similar degree inside each group, as parties are very uniform; the degree distribution could therefore in fact have a flat prior, and therefore the SBM faster MCMC convergence properties compared to the DC-SBM. The Markov chain was only run for a few hours on an average laptop and more extensive optimisation approaches were not practical on the available hardware, so it is also possible the partition is in a metastable state and would still equilibrate later (9); we discuss this briefly



(a) DC-SBM **(b)** SBM

Fig. 3. Posterior probabilities for the number of groups B

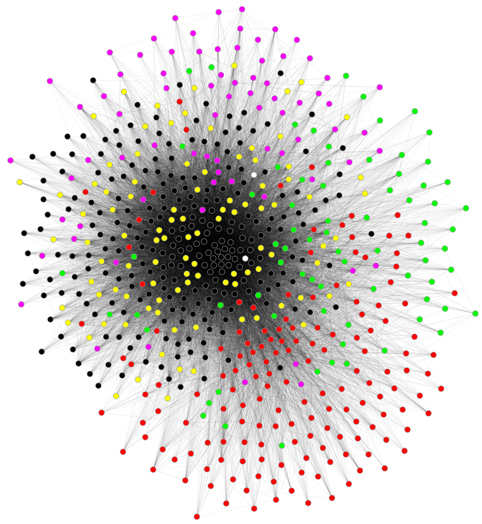


Fig. 4. Inferred DC-SBM for election period 17 - vertex colour corresponds to party affiliation. Opposition: ■ Greens, ■ SPD, ■ Left; Government: ■ Liberals, ■ Conservatives, party edge cases shown in white. Edge width is proportional to the posterior probability of the edge.

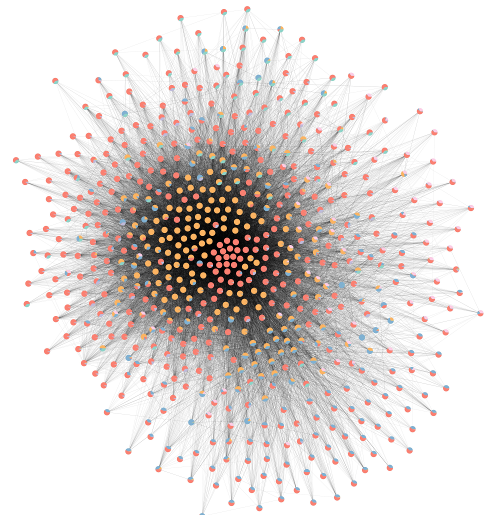


Fig. 5. Inferred DC-SBM for election period 17 - vertex pie chart shows marginal probability of being in inferred DC-SBM block of corresponding colour. Edge width is proportional to the posterior probability of the edge.

in Sec. 4.

Fig. 3 shows the entire posterior distribution of groups; first it can be observed that it is quite concentrated, however the more consistent results of the SBM can also be seen here in form of higher concentration. We also note that this inference of the number of groups themselves is a demonstration of the non-parametric nature of the Bayesian block modelling approach, which is also an advantageous feature compared to other methods, where such parameters need to be set artificially.

E. Modelling issues. One issue is that the Bayesian SBM inference has a resolution limit, i.e. a minimal size of blocks that can be found, which scales as $\mathcal{O}(\sqrt{N})$ (4). It is imaginable that there are communities of MPs with a size smaller than that, which could then not be resolved by the currently employed approach. Moreover in this approach a number of categories are mapped to a neutral state - e.g. a MP voting 'abstain' is equivalent to an MP being absent. This might lead to subtle misrepresentations; e.g. MPs sometime disagree with their party line, but then are absent at the vote instead of publicly voting against the policy. Furthermore in Sec. C we assumed that there is no change in time, i.e. relations and partitions of MPs stay constant over the entire election period. This might in fact not be true, as evidenced for example by MPs changing or leaving their parties during an election period (white vertices in network).

4. Conclusion & Future work

In this report we gave an introductory review of all the necessary general background of the Bayesian blockmodelling framework and its specific application to joint stochastic blockmodelling and network inference from dynamics. In particular we discussed the conceptual formulation, concrete implementation by choice of priors and efficient numerical MCMC algorithms for for each respectively. We then demonstrated the method with an empirical voting dataset and were able to show successes, such as the successful inference of plausible SBM structure with community and core-periphery behaviours, as well as problems with for example the potential non convergence and interpretability of the DC-SBM.

Sec. E already gave a starting point for future work - it might be interesting to apply the generalisation of a nested SBM to the voting dataset; this would allow to detect smaller blocks and build up a hierarchy of image graphs. Moreover, many more ideas for potentially more efficient sampling from the posterior distribution than discussed in this report exist, it might be interesting to compare their performance for the roll call voting inference problem. In this context, more advanced MCMC methods (9) might also allow to illuminate the non-convergence or metastability of the DC-SBM.

1. Peixoto TP (2019) Network reconstruction and community detection from dynamics. *Physical review letters* 123(12):128301.
2. Lambiotte R, Schaub M (2021) *Modularity and dynamics on complex networks.* (Cambridge University Press).
3. Reichardt J, White DR (2007) Role models for complex networks. *The European Physical Journal B* 60(2):217–224.
4. Peixoto TP (2019) Bayesian stochastic blockmodeling. *Advances in network clustering and blockmodeling* pp. 289–332.
5. Karrer B, Newman ME (2011) Stochastic blockmodels and community structure in networks. *Physical review E* 83(1):016107.
6. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. *The journal of chemical physics* 21(6):1087–1092.
7. Hastings WK (1970) Monte carlo sampling methods using markov chains and their applications.
8. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. *science* 220(4598):671–680.
9. Peixoto TP (2014) Efficient monte carlo and greedy heuristic for the inference of stochastic block models. *Physical Review E* 89(1):012804.
10. Besag J (1974) Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)* 36(2):192–225.
11. Bergmann H, Bailer S, Ohmura T, Saalfeld T, Sieberer U (2018) Codebook for BTVote datasets.
12. Bergmann H, Bailer S, Ohmura T, Saalfeld T, Sieberer U (2018) BTVote Voting Behavior.
13. Peixoto TP (2014) The graph-tool python library. *figshare.* figshare.com/articles/graph_tool/1164194, DOI: 10.6084/m9.figshare.1164194.
14. Peixoto TP (2020) Revealing consensus and dissensus between network partitions. *arXiv preprint arXiv:2005.13977.*

Supplementary Information

A. Code listings. The most important numerical procedures can be found in listings 1. The entire code used for data preparation and numerical simulations can be found in the attached `.ipynb` file.

B. Additional numerical results of *Bundestag* RCV Bayesian inference. Figs. 6,7 show the rate of absence and party line deviation for all MPs.

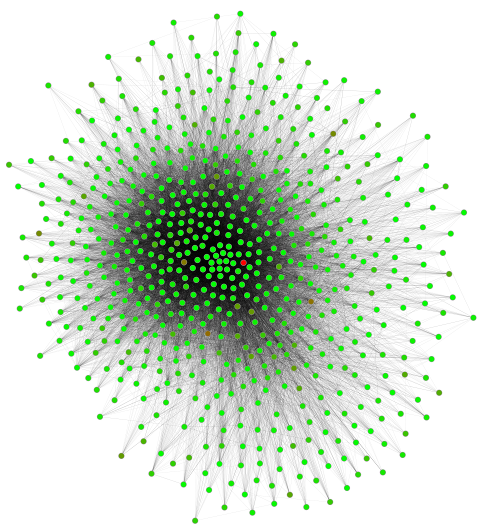


Fig. 6. DC-SBM inference for 17th election period: the greener a vertex the more often the corresponding MP has taken part in RCVs; i.e. red vertices correspond to MPs that are often absent. In general a high participation in RCVs can be seen.

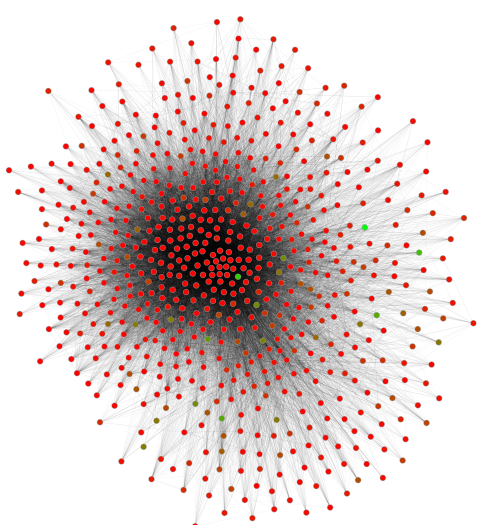


Fig. 7. DC-SBM inference for 17th election period: the redder a vertex the more often the corresponding MP has deviated from the party line. In general high levels of party unity can be seen.

```python
import graph_tool as gt

# read out BTVote, construct graph; see other helper functions in .ipynb file

state17nonnestedpseudonondegcorr = gt.PseudoIsingBlockState(g17, s=g17.vertex_properties['votingbeh'], beta=0.1, has_zero=True, nested=False, state_args=dict(deg_corr=False))
# mcmc equilibrate, so bs17nonnestedpseudonondegcorr, which are collected next can be used for marginal/maximal determination of b
for i in range(0, 1000):
    state17nonnestedpseudonondegcorr.mcmc_sweep()
# prepare mcmc equilibrate, collect_marginals_17nonnestedpseudonondegcorr will be callback after each mcmc_sweep in mcmc_equilibrate
gm17nonnestedpseudonondegcorr = None # contains final network reconstruction
bs17nonnestedpseudonondegcorr = [] # contains block association after each mcmc_sweep in mcmc_equilibrate
numgroupsnonnestedpseudonondegcorr = np.zeros(g17.num_vertices() + 1) #count a histogram of how many blocks were determined
def collect_marginals_17nonnestedpseudonondegcorr(s):
    global gm17nonnestedpseudonondegcorr, bs17nonnestedpseudonondegcorr, numgroupsnonnestedpseudonondegcorr
    gm17nonnestedpseudonondegcorr = s.collect_marginal(gm17nonnestedpseudonondegcorr)
    bs17nonnestedpseudonondegcorr.append(s.bstate.b.a.copy())
    numgroupsnonnestedpseudonondegcorr[s.get_block_state().get_nonempty_B()] += 1

# repeatedly call mcmc_sweep, until there is no new entropy extremum for 1000 steps
entropy17nonnestedpseudonondegcorr, nattempts, nmoves = gt.mcmc_equilibrate(state17nonnestedpseudonondegcorr, max_niter=50000,
        mcmc_args=dict(niter=10, xstep=0), callback=collect_marginals_17nonnestedpseudonondegcorr)

partitionmode17nonnestedpseudonondegcorr = gt.PartitionModeState(bs17nonnestedpseudonondegcorr)
bspropmap_17nonnestedpseudonondegcorr_max = partitionmode17nonnestedpseudonondegcorr.get_max(gm17nonnestedpseudonondegcorr)
bspropmap_17nonnestedpseudonondegcorr_marginal = partitionmode17nonnestedpseudonondegcorr.get_marginal(gm17nonnestedpseudonondegcorr)

# determine spring layout so communites / blocks are well visualised later
pos_nondegcorr=gt.sfdp_layout(gm17nonnestedpseudonondegcorr,
                                groups=bspropmap_17nonnestedpseudonondegcorr_max,
                                eweight=gm17nonnestedpseudonondegcorr.ep['eprob'],
                                mu=50.0, mu_p=2, gamma=0, K=100, C=100.0)
gt.graph_draw(gm17nonnestedpseudonondegcorr,
                pos=pos_nondegcorr,
                vertex_fill_color=gm17nonnestedpseudonondegcorr.own_property(g17.vp['partycolor']),
                vertex_size=8, output_size=(800, 800),
                edge_color=[0, 0, 0, 0.2],
                edge_pen_width=gt.prop_to_size(gm17nonnestedpseudonondegcorr.ep['eprob'], 0, 0.4))
gt.graph_draw(gm17nonnestedpseudonondegcorr,
                pos=pos_nondegcorr,
                vertex_fill_color=gm17nonnestedpseudonondegcorr.new_vertex_property('vector<float>', list(map(mapcolor, list(bspropmap_17nonnestedpseudonondegcorr_max)))),
                vertex_size=8, output_size=(800, 800),
                edge_color=[0, 0, 0, 0.2],
                edge_pen_width=gt.prop_to_size(gm17nonnestedpseudonondegcorr.ep['eprob'], 0, 0.4))
gt.graph_draw(gm17nonnestedpseudonondegcorr,
                pos=pos_nondegcorr,
                vertex_shape='pie',
                vertex_pie_fractions=bspropmap_17nonnestedpseudonondegcorr_marginal,
                vertex_size=8, output_size=(800, 800),
                edge_color=[0, 0, 0, 0.2],
                edge_pen_width=gt.prop_to_size(gm17nonnestedpseudonondegcorr.ep['eprob'], 0, 0.4))
```


Listing 1. Main numerical code for the joint inference of a SBM and the network itself utilising the `graph-tool` library; this is exemplary for the 17th election period and only for a SBM (as opposed to a DC-SBM).