
Inferring community characteristics in labelled networks

Anonymous Author(s)

Affiliation
Address
email

Abstract

1 Labelled networks form a very common and important class of data, naturally
2 appearing in numerous applications in science and engineering. A typical inference
3 goal is to determine how the vertex labels (or *features*) affect the network’s
4 graph structure. A standard approach has been to partition the network into blocks
5 grouped by distinct values of the feature of interest. A block-based random graph
6 model – typically a variant of the stochastic block model – is then used to test
7 for evidence of asymmetric behaviour within these feature-based communities.
8 Nevertheless, the resulting communities often do not produce a natural partition of
9 the graph. In this work we introduce a new generative model, the feature-first block
10 model (FFBM), which is more effective at describing vertex-labelled undirected
11 graphs and also facilitates the use of richer queries on labelled networks. We
12 develop a Bayesian framework for inference with this model, and we present a
13 method to efficiently sample from the posterior distribution of the FFBM parame-
14 ters. The FFBM’s structure is kept deliberately simple to retain easy interpretability
15 of the parameter values. We apply the proposed methods to a variety of network
16 data to extract the most important features along which the vertices are partitioned.
17 The main advantages of the proposed approach are that the whole feature-space
18 is used automatically, and features can be rank-ordered implicitly according to
19 impact. Any features that do not significantly impact the high-level structure can
20 be discarded to reduce the problem dimension. In cases where the vertex features
21 available do not readily explain the community structure in the resulting network,
22 the approach detects this and is protected against over-fitting. Results on several
23 real-world datasets illustrate the performance of the proposed methods.

24

1 Introduction

25 An important characteristic of many real-world networks is that they exhibit strong community
26 structure, with most nodes often belonging to a densely connected cluster. Finding ways to recover
27 the latent communities from the observed graphs is an important task in many applications, includng
28 graph/network compression [1] and link prediction in incomplete networks [4].

29 In this work, we restrict our attention to vertex-labelled networks, and we refer to the vertex labels as
30 *features*. A common goal is to determine whether a given feature impacts graphical structure. To
31 answer this from a Bayesian perspective requires the use of a random graph model; the standard is
32 the stochastic block model (SBM) [8]. This is a latent variable model where each vertex belongs to a
33 single block and the probability two nodes are connected depends only on the block memberships
34 of each. Numerous variants of this model have been considered – the most popular ones being the

35 mixed-membership stochastic block model (MMSBM) [2] and the overlapping stochastic block model
 36 (OSBM) [17]. Effectively, these extend the model to allow each vertex to belong to multiple blocks
 37 simultaneously. However, one drawback of these graphical models as applied to labelled networks
 38 is that they do not automatically include vertex features in the random graph generation process.
 39 Approaches based on graph neural networks [7] that utilise vertex features have been developed but
 40 these lack the easy interpretability of the simpler models.

41 To analyse a labelled network using one of the simple SBM variants, a typical inference procedure
 42 would be to first partition the graph into blocks grouped by distinct values of the feature of interest,
 43 and then use the associated model to test for evidence of heterogeneous connectivity between the
 44 feature-grouped blocks. Nevertheless, this approach is limited in that it can only consider one feature
 45 at a time. This makes it difficult to rank-order the features by magnitude of impact. Lastly, the
 46 feature-grouped blocks are often an unnatural partition of the graph, leading to a poor model fit. We
 47 would instead prefer to partition the graph into its most natural blocks and then find which of the
 48 available features – if any – best predict the resulting partition.

49 Thus motivated, we present a novel framework for modelling labelled networks, which we call the
 50 feature-first block model (FFBM). This is an extension of the SBM to labelled networks. We go on
 51 to present an efficient algorithm for sampling from the parameters of the feature-to-block generator,
 52 and to describe how the sampled FFBM parameters can be interpreted in order to determine which
 53 features have the largest impact on overall graphical structure.

54 2 Preliminaries

55 We first need a model for community-like structure in a network. For this we adopt the widely-used
 56 stochastic block model (SBM). Each node in the graph belongs to a unique community called a
 57 block. The probability that two nodes are connected depends only on the block memberships of
 58 each. Specifically, we will use the microcanonical variant of the SBM, proposed by Peixoto [13].
 59 To allow for degree-variability between members of the same block, we adopt the degree-corrected
 60 formulation (DC-SBM).

61 For each integer $K \geq 1$, we use the notation $[K] := \{1, 2, \dots, K\}$.

62 **Definition 2.1 (Microcanonical DC-SBM)** Let $N \geq 1$ denote the number of vertices in an undi-
 63 rected graph. The block memberships are encoded by a vector $b \in [B]^N$, where B is the number of
 64 non-empty blocks. Let $e = (e_{rs})$ be the $N \times N$ symmetric matrix of edge counts between blocks, so
 65 that e_{rs} is the number of edges from block r to block s – or twice that number if $r = s$. Let $k = (k_i)$
 66 denote the vector of length N containing the degree sequence of the graph, with k_i being the degree
 67 of vertex i .

68 The graph's adjacency matrix $A \in \{0, 1\}^{N \times N}$ is generated by placing edges uniformly at random,
 69 but respecting the constraints imposed by e , b and k (hence the qualification ‘microcanonical’).
 70 Specifically, A must satisfy the following, for all $r, s \in [B]$ and all $i \in [N]$:

$$e_{rs} = \sum_{i,j \in [N]} A_{ij} \mathbb{1}\{b_i = r\} \mathbb{1}\{b_j = s\} \quad \text{and} \quad k_i = \sum_{j \in [N]} A_{ij}. \quad (1)$$

71 We use the following notation to indicate this distribution:

$$A \sim \text{DC-SBM}_{\text{MC}}(b, e, k). \quad (2)$$

72 3 Feature-first block model

73 In this section we propose a novel generative model for labelled networks. We call this the feature-first
 74 block model (FFBM).

75 Let N denote the number of nodes, B the number of blocks in the graph, and \mathcal{X} the set of all possible
 76 features. We define the vector $x_i \in \mathcal{X}^D$ as the feature vector of vertex i , where D is the number of

77 features associated with each vertex. For the datasets we analyse, we deal with binary feature flags so
 78 $\mathcal{X} = \{0, 1\}$. We write X for the $N \times D$ feature matrix containing the feature vectors $\{x_i\}_{i=1}^N$ as its
 79 rows.

80 For the FFBM, we start with the feature matrix X and generate a random vector of block memberships
 81 $b \in [B]^N$. For each node i , the block membership $b_i \in [B]$ is generated based on the feature vector
 82 x_i , independently between nodes. The conditional distribution of b_i given x_i also depends on a
 83 collection of weight vectors $\theta = \{w_k\}_{k=1}^B$, where each w_k has dimension $D + 1$. Specifically, the
 84 distribution of b given X and θ is,

$$p(b|X, \theta) = \prod_{i=1}^N p(b_i|x_i, \theta) = \prod_{i=1}^N \phi_{b_i}(x_i; \theta) = \prod_{i=1}^N \frac{\exp(w_{b_i}^T x_i)}{\sum_{k=1}^B \exp(w_k^T x_i)} \quad (3)$$

85 We deliberately exclude a bias term to ensure that the relationships we model are based on features
 86 and not information about the size of each detected block; a more complete discussion on this topic is
 87 given in ???. We will later also find it convenient to write the parameters θ as a $B \times (D + 1)$ matrix
 88 of weights W ; this form has computational benefits as then $z_i := Wx_i$, which is the input to the
 89 softmax activation function.

90 More complex models based on different choices for the distributions ϕ_{b_i} above are also possible, but
 91 then deriving meaning from the inferred parameter distributions is more difficult.

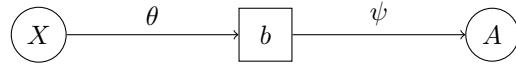


Figure 1: The feature-first block model (FFBM)

92 Once the block memberships b have been generated, we then draw the graph A from the microcanonical-
 93 ical DC-SBM with additional parameters $\psi = \{\psi_e, \psi_k\}$:

$$A \sim \text{DC-SBM}_{\text{MC}}(b, \psi_e, \psi_k). \quad (4)$$

94 3.1 Prior selection

95 To complete the description of our Bayesian framework, priors on θ and ψ must also be specified.
 96 We place a Gaussian prior on θ with zero mean and covariance matrix $\sigma_\theta^2 I$, so that each element of θ
 97 has an independent $\mathcal{N}(0, \sigma_\theta^2)$ prior, for a hyperparameter σ_θ^2 :

$$p(\theta) \sim \mathcal{N}(\theta; 0, \sigma_\theta^2 I). \quad (5)$$

98 Interestingly, this choice of prior leads to a particularly simple distribution on the block membership
 99 vector b . We show in Appendix A.2 that, after integrating out θ , b is uniformly distributed:

$$p(b|X) = \int p(b|X, \theta)p(\theta)d\theta = B^{-N}. \quad (6)$$

100 This is an important simplification as evaluating $p(b|X)$ does not require an expensive Monte Carlo
 101 integration over θ nor does it require the exact value of X . Peixoto [13] proposes careful choices
 102 for the additional microcanonical SBM parameters ψ , which we adopt. The idea is to write the joint
 103 distribution on (b, e, k) as a product of conditionals, $p(b, e, k) = p(b)p(e|b)p(k|e, b) = p(b)p(\psi|b)$.
 104 In our case, conditioning on X is also necessary, leading to,

$$p(b, \psi|X) = p(b|X)p(\psi|b, X) = p(b|X)p(\psi|b), \quad (7)$$

105 where we used the fact ψ and X are conditionally independent given b . The exact form of the prior
 106 $p(\psi|b)$ is described in Appendix A.1. All that concerns the main argument is that it has an easily
 107 computable form.

108 **4 Inference**

109 Having completed the definition of the FFBM, we wish to leverage it to perform inference. Given a
110 vertex-labelled graph (A, X) , the first step is to sample from the posterior distribution of θ :

$$\theta \sim p(\theta|A, X). \quad (8)$$

111 We propose an iterative approach to obtain samples $\theta^{(t)}$ from $p(\theta|A, X)$. We first draw samples $b^{(t)}$
112 from the block membership posterior distribution, and then use each $b^{(t)}$ to obtain a corresponding
113 sample $\theta^{(t)}$:

$$b^{(t)} \sim p(b|A, X), \quad (9)$$

$$\theta^{(t)} \sim p(\theta|X, b^{(t)}). \quad (10)$$

114 Both of these sampling steps can be implemented using Markov chain Monte Carlo via a Metropolis-
115 Hastings algorithm [5], as a two-level Markov chain.

116 As we show in the following section, the resulting samples for $\theta^{(t)}$ are unbiased in the sense that the
117 expectation of their distribution is the target posterior distribution:

$$\mathbb{E}_{b^{(t)}} \left[p(\theta|X, b^{(t)}) \right] = \sum_{b \in [B]^N} p(\theta|X, b) p(b|A, X) = \sum_{b \in [B]^N} p(\theta, b|A, X) = p(\theta|A, X). \quad (11)$$

118 This is an example of a pseudo-marginal approach. Indeed, Andrieu and Roberts [3] show that (11) is sufficient to prove that, for large enough t , $\theta^{(t)} \sim \mathbb{E}_{b^{(t)}} [p(\theta|X, b^{(t)})] = p(\theta|A, X)$, as required.

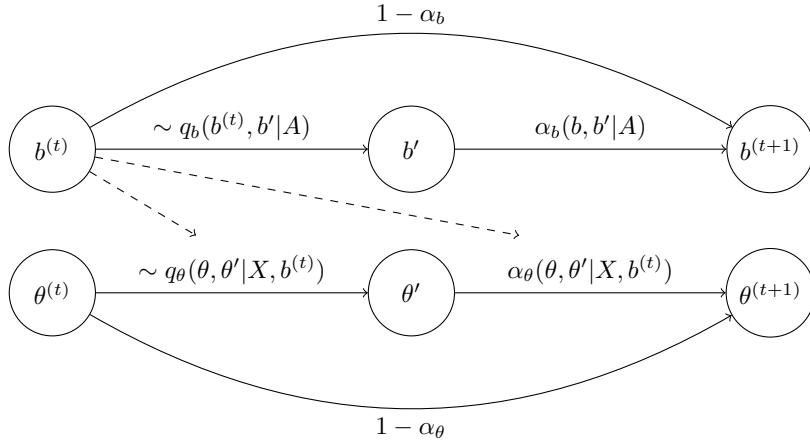


Figure 2: Sampling sequence.

119
120 The reason for splitting the Markov chain into two levels is because the summation over all latent
121 states $b \in [B]^N$ required to directly compute the likelihood, $p(A|X, \theta) = \sum_{b \in [B]^N} p(A|b)P(b|X, \theta)$,
122 is intractable, as the sum involves B^N terms. Figure 2 shows an overview of the proposed method.
123 The Metropolis-Hastings accept/reject probabilities for the proposed b and θ samples are denoted α_b
124 and α_θ , respectively.

125 Note the importance of the simplification in (6). As $p(b|X)$ in fact does not depend on X , knowing
126 X is not necessary in order to obtain samples of the block membership vector b . And on the other
127 level, in order to obtain samples of the parameter vector θ we use only b but not A , as θ and A are
128 conditionally independent given b .

129 **4.1 Sampling block memberships**

130 We adopt the Markov chain Monte Carlo procedure of [11], which relies on writing the posterior in
131 the following form:

$$p(b|A, X) \propto p(A|b, X) \cdot p(b|X) = \pi_b(b). \quad (12)$$

132 Now $\pi_b(\cdot)$ is the un-normalised target density, which can be expressed as:

$$\pi_b(b) = p(b|X) \sum_{\psi} p(A, \psi|b, X) = p(b|X)p(A, \psi^*|b, X) = p(A|b, \psi^*) \cdot p(\psi^*|b) \cdot p(b|X). \quad (13)$$

133 Since we are using the microcanonical SBM formulation, there is only one value of ψ that is
 134 compatible with the given (A, b) pair; recall the constraints in (1). We denote this value $\psi^* =$
 135 $\{\psi_k^*, \psi_e^*\}$. Therefore, the summation over all ψ reduces to just the single ψ^* term. We also define the
 136 microcanonical entropy of the configuration as:

$$S(b) = -\log \pi_b(b) = -\left(\log p(A|b, \psi^*) + \log p(\psi^*, b|X) \right). \quad (14)$$

137 This entropy can be thought of as the optimal ‘‘description length’’ of the graph. The exact form
 138 of the proposal q_b is explored thoroughly in [11] and not repeated here. There is a widely used
 139 library for Python made available under LGPL called `graph-tool` [12], which implements this
 140 algorithm. The only modification we make is in the block membership prior $p(b)$ that we replace
 141 with $p(b|X) = B^{-N}$, which cancels out in the MH accept-reject step as it is independent of b .

142 4.2 Sampling feature-to-block generator parameters

143 The invariant distribution we wish to target for the θ samples is the posterior of θ given the values of
 144 the pair (X, b) . We write this as,

$$\pi_\theta(\theta) \propto p(\theta|X, b) \propto p(b|X, \theta)p(\theta) \propto \pi_\theta(\theta) \propto \exp(-U(\theta)), \quad (15)$$

145 where we write $U(\theta)$ for the negative log-posterior. Let $y_{ij} := \mathbb{1}\{b_i = j\}$ and $a_{ij} := \phi_j(x_i; \theta)$.
 146 Discarding constant terms, we can write $U(\theta)$ as,

$$U(\theta) = \left(\sum_{i=1}^N \sum_{j=1}^B y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 = N \cdot \mathcal{L}(\theta) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2; \quad (16)$$

see Appendix A.3 for the derivation. The function $U(\theta)$ is a typical objective function for neural network training. The first term is introduced by the likelihood. We collect it into $N \cdot \mathcal{L}(\theta)$, which is the cross-entropy between the graph-predicted and feature-predicted block memberships summed over all vertices. The second, introduced by the prior, brings a form of regularisation, guarding against over-fitting. Unlike in many applications where the goal is to find the minimiser of $U(\theta)$, our goal is to draw samples from the posterior $\pi_\theta(\cdot) \propto \exp(-U(\cdot))$. We can use ∇U as a useful heuristic to bias our proposal towards regions of higher target density. We therefore adopt the Metropolis-adjusted Langevin algorithm (MALA) [14]. Given the current sample θ , we generate a new sample θ' as,

$$\theta' = \theta - h \nabla U(\theta) + \sqrt{2h} \cdot \xi,$$

where $\xi \sim \mathcal{N}(0, I)$ and h is a step-size parameter which may vary with the sample index (see Appendix A.5). This leads to the proposal distribution,

$$q_\theta(\theta, \theta') \sim \mathcal{N}(\theta'; \theta - h \nabla U(\theta), 2hI).$$

147 Without the injected noise term, MALA is equivalent to gradient descent. We require the noise term
 148 ξ to fully explore the parameter space. The term ∇U has an easy to compute analytic form (derived
 149 in Appendix A.4). By noting that $\theta = \{w_k\}_{k=1}^B$, we write the derivative with respect to each w_k as:

$$\frac{\partial U}{\partial w_k} = - \left(\sum_{i=1}^N \left\{ \tilde{x}_i(y_{ik} - a_{ik}) \right\} - \frac{w_k}{\sigma_\theta^2} \right). \quad (17)$$

150 After a proposed move is generated from q_θ , it is either accepted or rejected in the standard Metropolis-
 151 Hastings accept/reject step.

152 **4.3 Sampling sequence**

153 Up to this point, each $\theta^{(t)}$ update uses its corresponding $b^{(t)}$ sample. This means that the evaluation
 154 of $U(\theta)$ and $\nabla U(\theta)$ has high variance. This may lead to longer burn-in for the resulting Markov
 155 chain. The only link between $b^{(t)}$ and $\theta^{(t)}$ is in the evaluation of $U(\theta)$ and $\nabla U(\theta)$ which depends
 156 only on the matrix $y^{(t)}$ with entries $y_{ij}^{(t)} := \mathbb{1}\{b_i^{(t)} = j\}$. We would rather deal with the expectation
 157 of each $y_{ij}^{(t)}$:

$$\mathbb{E} \left[y_{ij}^{(t)} \right] = \mathbb{E}_{b^{(t)}} \left[\mathbb{1}\{b_i^{(t)} = j\} \right] = p(b_i = j | A, X). \quad (18)$$

158 We can obtain an unbiased estimate for this quantity using the thinned b -samples after burn-in; the
 159 length of the burn-in and the details of the thinning are described in Appendix A.6. Let \mathcal{T}_b denote the
 160 retained set of indices for the b -samples and \mathcal{T}_θ similarly for the θ -chain. The unbiased estimate for
 161 $y_{ij}^{(t)}$ using the restricted sample set \mathcal{T}_b is denoted \hat{y}_{ij} :

$$\hat{y}_{ij} := \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} y_{ij}^{(t)} = \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} \mathbb{1}\{b_i^{(t)} = j\}. \quad (19)$$

162 The same matrix \hat{y} is used in the update step for each $\theta^{(t)}$. This way, it is not necessary to run the b
 163 and θ Markov chains concurrently. Instead, we run the b -chain to completion and use it to generate \hat{y} .
 164 This affords us the flexibility to vary the lengths of the b and θ -chains. Furthermore, the changeover
 165 from $y^{(t)}$ to \hat{y} reduces the burn-in time for the θ -chain by reducing the variance in the evaluation of
 166 U and ∇U . A detailed description of the overall algorithms is given in Appendix A.8.

167 **4.4 Dimensionality reduction**

168 Once the samples $\{\theta^{(t)}\} \sim p(\theta | A, X)$ have been obtained, we can compute the empirical mean and
 169 standard deviation of each component of θ . Switching back to matrix notation we define $\theta = W$,
 170 such that W_{ij} is the weight component for block i and feature j , we can define:

$$\hat{\mu}_{ij} := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} W_{ij}^{(t)} \quad \text{and} \quad \hat{\sigma}_{ij}^2 := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} \left(W_{ij}^{(t)} - \hat{\mu}_{ij} \right)^2 \quad (20)$$

171 A simple heuristic to discard the least important features requires specifying a cutoff $c > 0$ and a
 172 multiplier $k > 0$. We define the function $\mathcal{F}_i(j)$ as in (21) and only keep features with indices $d \in \mathcal{D}'$,
 173 where \mathcal{D}' is given in (22).

$$\mathcal{F}_i(j) := (\hat{\mu}_{ij} - k\hat{\sigma}_{ij}, \hat{\mu}_{ij} + k\hat{\sigma}_{ij}) \cap (-c, +c), \quad (21)$$

$$\mathcal{D}' := \{j \in [D] : \exists i \in [B] \text{ s.t. } \mathcal{F}_i(j) = \emptyset\} \quad (22)$$

174 Intuitively, this means discarding any feature j for which $\hat{\mu}_{ij} \pm k\hat{\sigma}_{ij}$ lies outside the region $(-c, c)$
 175 for all block indices i . If we were to use the Laplace approximation for the posterior $p(W_{ij} | A, X) \approx$
 176 $\mathcal{N}(W_{ij}; \mu_{ij}, \sigma_{ij}^2)$, then this is analogous to a hypothesis test on the value of W_{ij} as in (23); then \mathcal{D}'
 177 comprises all features i for which H_1 is not rejected at least once for some $j \in [B]$

$$H_0 : |W_{ij}| \leq c \quad H_1 : |W_{ij}| > c \quad (23)$$

178 The multiplier k in equation 21 determines the degree of significance of the result. Appendix ??
 179 explores the derivation in more detail. However, as the Laplace approximation is not exact we
 180 will only treat this dimensionality reduction method as a useful heuristic and not an exact method.
 181 Conversely, we could fix $k = k_0$ and the dimension of our reduced feature set $|\mathcal{D}'| = D'$. We would
 182 then like to find the largest value of c such that $|\mathcal{D}'| = D'$ given $k = k_0$. This is summarised in
 183 equation 24. This approach is often preferred as it fixes the number of reduced dimensions.

$$c^* = \arg \max \{c > 0 : |\mathcal{D}'| = D', k = k_0\}. \quad (24)$$

184 For an algorithmic implementation of this method refer to appendix A.8. This approach has the
 185 advantage that it fixes the number of reduced dimensions.

186 **5 Experimental results**

187 We apply the developed methods to a variety of datasets. These are chosen to span a range of node
 188 counts N , edge counts E and feature space dimension D . We consider the following:

- 189 • **Political books** [9] ($N = 105, E = 441, D = 3$) – network of Amazon book sales about U.S.
 190 politics, published close to the presidential election in 2004. Two books are connected if they were
 191 frequently co-purchased by customers. Vertex features encode the political affiliation of the author
 192 (liberal, conservative, or neutral).
- 193 • **Primary school dynamic contacts** [15] ($N = 238, E = 5539, D = 13$) – network of face-to-face
 194 contacts amongst students and teachers at a primary school in Lyon, France. Two nodes are
 195 connected if the two parties shared a face-to-face interaction over the school-day. Vertex features
 196 include class membership (one of 10 values: 1A-5B), gender (male, female) and teacher status
 197 encoded as an 11th school-class. No further identifiable information is retained. We choose to
 198 analyse just the second day of results.
- 199 • **Facebook egonet** [6] ($N = 747, E = 30025, D = 480$) – an assortment of Facebook users'
 200 friends lists. Vertex features are extracted from each user's profile and are fully anonymised. They
 201 include information about education history, languages spoken, gender, home-town, birthday etc.
 202 We focus on the egonet with id 1912.

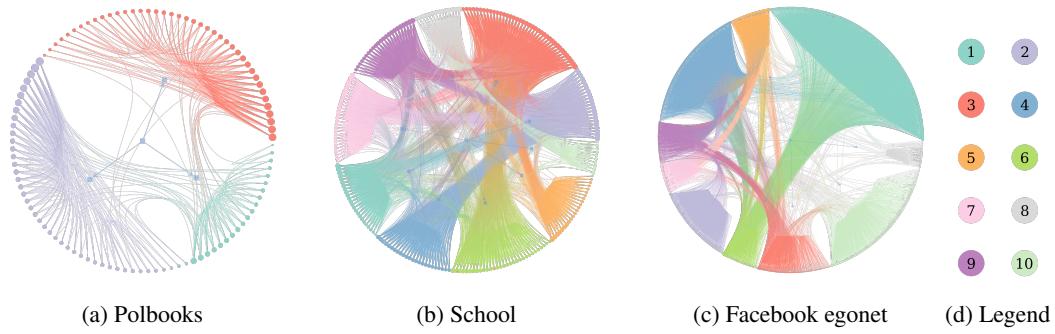


Figure 3: Networks laid out and coloured according by inferred block memberships \hat{y} for a given experiment iteration. Visualisation performed using *graph-tool* [12].

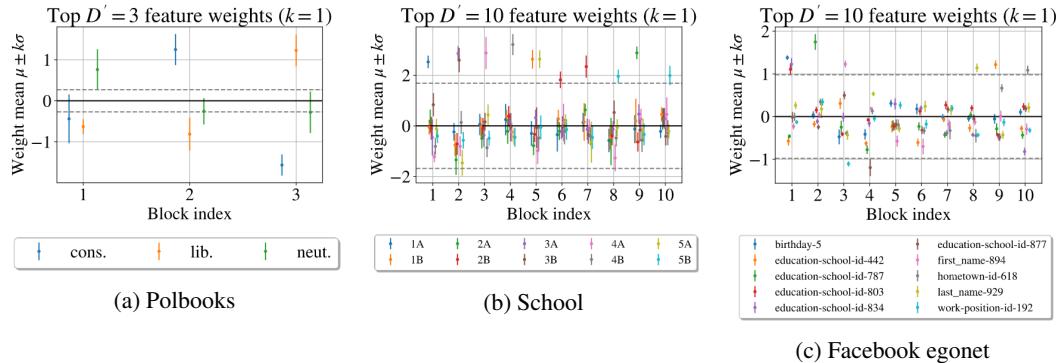


Figure 4: Reduced dimension feature-to-block generator weight samples.

203 We require metrics to assess performance. This can be split into two separate components: the
 204 microcanonical SBM fit (concerned with the b -samples) and the fit of the feature-to-block generator
 205 (concerned with the θ -samples).

206 Starting with the SBM, recall that the quantity $S(b)$ in (14) can be interpreted as an ideal “description
 207 length” of the partition imposed by b . We define a simple metric \bar{S}_e to gauge the fit of the SBM,

Table 1: Experimental results averaged over $n = 10$ iterations (mean \pm standard deviation).

Dataset	B	D	D'	\bar{S}_e	$\bar{\mathcal{L}}_0$	$\bar{\mathcal{L}}_1$	c^*	$\bar{\mathcal{L}}'_0$	$\bar{\mathcal{L}}'_1$
Polbooks	3	3	–	2.250 ± 0.000	0.563 ± 0.042	0.595 ± 0.089	–	–	–
School	10	13	10	1.894 ± 0.004	0.787 ± 0.127	0.885 ± 0.129	1.198 ± 0.249	0.793 ± 0.132	0.853 ± 0.132
FB egonet	10	480	10	1.626 ± 0.003	1.326 ± 0.043	1.538 ± 0.069	0.94 ± 0.019	1.580 ± 0.150	1.605 ± 0.106

as the description length per entity (i.e., divided by the total number $N + E$ of nodes plus edges), averaged over the b -samples:

$$\bar{S}_e := \frac{1}{(N+E)|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} S(b^{(t)}). \quad (25)$$

Next, in order to assess the performance of the feature-to-block predictor, we partition the vertex set $[N]$ into training and test sets. We choose to randomly partition the vertices on each experiment run, so that a constant fraction f of the available vertices go to form the training set \mathcal{G}_0 and the remainder are held out to form the test set \mathcal{G}_1 . The b -chain is run using the whole network but we only use vertices $v \in \mathcal{G}_0$ to train the θ -chain. Because $|\mathcal{G}_0| \neq |\mathcal{G}_1|$ in general, we cannot use the un-normalised log-target U from (16) for comparison, as the total cross-entropy loss scales with the size of each data set but the prior term stays constant. We therefore use the average cross-entropy loss over each set,

$$\bar{\mathcal{L}}_\star := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} \mathcal{L}_\star(\theta^{(t)}), \quad \text{where } \mathcal{L}_\star(\theta^{(t)}) := \frac{1}{|\mathcal{G}_\star|} \sum_{i \in \mathcal{G}_\star} \sum_{j \in [B]} \hat{y}_{ij} \log \frac{1}{\phi_j(x_i; \theta^{(t)})}, \quad (26)$$

and where $\star \in \{0, 1\}$ indicates whether the training or test set is being considered.

Table 1 summarises the results for each experiment; a complete list of all the hyperparameters used is given in Appendix A.9. We also apply the dimensionality reduction method of Section 4.4 to the two higher dimensional datasets (the school and FB egonet). For this we use equation (24) with $k = 1$, in order to reduce the dimension from D to a desired D' with $k = 1$ based on the maximal cutoff c^* . We then retrain the feature-block predictor using only the retained feature set \mathcal{D}' , and report the log-loss over the training and test sets for the reduced classifier – denoted $\bar{\mathcal{L}}'_0$ and $\bar{\mathcal{L}}'_1$ respectively. These values are also shown in Table 1.

Based on these results, some remarks are in order. Firstly, the variance of the test loss $\bar{\mathcal{L}}_1$ tends to be higher than the training loss $\bar{\mathcal{L}}_0$. This is expected, as the test set is smaller than the training set and hence more susceptible to variability in its construction. Indeed, most of the variance in the evaluation of $\bar{\mathcal{L}}_0$ and $\bar{\mathcal{L}}_1$ comes from the random partitioning of the graph into training and test sets. Secondly, it can be seen that the dimensionality reduction procedure brings the training and test losses closer together. This implies that the features we keep are indeed correlated with the underlying graphical partition and that the approach generalises correctly.

The average description length per entity, \bar{S}_e , of the graph, has very small variance, suggesting that the detected communities can be found reliably (to within an arbitrary relabelling of blocks). For reference, we plot an inferred partition for each of the graphs in Figure 3. The polbooks graph yields the cleanest separation between blocks but nonetheless the inferred partitions for the other datasets do succeed at partitioning the graph into densely connected clusters.

5.1 Political books

The goal here is to determine whether the authors' political affiliations are a good predictor of the overall network structure. We choose to partition the network into $B = 3$ communities as we only have this many distinct values for political affiliation (conservative, liberal, or neutral). From Figure 4a we see that all 3 blocks have a distinct political affiliation as their largest positive component. This strongly indicates that political affiliation is indeed the axis which best predicts the 3-way natural partition of the graph into blocks. Furthermore, in Table 1 we see that the training and test losses are very similar and both are low in magnitude. This provides further evidence to the claim that political affiliation is a very appropriate explanatory variable for the overall network structure.

246 **5.2 Primary school dynamic contacts**

247 We choose the number of communities $B = 10$, in line with the total number of school classes.
248 As before, we sample the block-generator parameters θ and employ the dimensionality reduction
249 technique of Section 4.4, with standard deviation multiplier $k = 1$ to pick out the top $D' = 10$
250 features. We then plot the weights for the resulting features $d \in \mathcal{D}'$ in Figure 4b. It is rather
251 obvious that only the pupils’ class memberships (1A-5B) have remained significant; gender and
252 teacher/student status have been discarded, meaning that these are not good predictors of overall
253 macro-structure.

254 The vast majority of blocks are composed of a single class. However, some blocks have two
255 comparably good classes as their predictor. For example, block 2 contains classes 3A and 3B as its 2
256 best predictors. This suggests that the social divide between classes is less pronounced for pupils in
257 year 3. Conversely, some classes are found to extend over two detected blocks (class 2B spans blocks
258 6 and 7) but we nonetheless do not have a feature which explains the division. The most surprising
259 block is number 5, which has comparable weightings for classes 5A and 1B. Perhaps there was a
260 joint event between those two classes on the day the data were collected.

261 **5.3 Facebook egonet**

262 We choose $B = 10$ and $D' = 10$ for this experiment. The selected features (Figure 4c) are those
263 that best explain the high-level community structure. The majority of them are education related.
264 Nevertheless, for $D' = 10$ we only have good explanations for the makeup of some of the detected
265 blocks; several blocks in Figure 4c do not have high-magnitude components for $D' = 10$.

266 When the feature dimension is very large, it becomes increasingly likely that a particular feature
267 may uniquely identify a small set of nodes. If these nodes are all part of the same community, then
268 the classifier will overfit for that particular parameter. The regularisation term imposed by the prior
269 goes some way towards alleviating this problem. Nevertheless, we see in Figure 4c that the feature
270 `birthday-5` has a very high weight as it relates to block 1 – but it is highly unlikely that birthdays
271 determine graphical structure.

272 **6 Conclusion**

273 The Feature-First Block Model (FFBM) introduced in this paper is a new generative model for
274 labelled networks, developed in order to address difficulties of other graphical models when testing
275 how vertex features affect community structure. The idea is to divide the graph into its most natural
276 partition and test whether the vertex features can accurately explain this partition. It is relatively easy
277 to find vertex features that are in some way correlated with the graphical structure. Nonetheless, only
278 when we find the feature that best describes the most pronounced partition do we have a stronger
279 case for causation.

280 Using this new model, we go on to describe an efficient inference algorithm to sample the parameters
281 θ of the feature-to-block generator. This takes the form of two concurrent Markov chains, used
282 to sample the block memberships b and block generator parameters θ . These chains can either be
283 executed in parallel or the empirical mean of the b -samples can be used as the input to the θ -chain.
284 The latter option reduces the variance in our evaluation of the target distribution and thus shortens
285 burn-in.

286 The overall method is shown to be effective at extracting and describing the most natural communities
287 in a labelled network. Nevertheless, the approach can only currently explain the structure at the
288 macro-scale. We cannot explain structure within each block. Future work will benefit from extending
289 the FFBM to be hierarchical in nature. That way, the structure of the network can be explained at all
290 several scales of interest. So long as data collection techniques remain ethical and care is taken to
291 respect personal privacy, such empowered decision-making can only help humankind.

292 **References**

- 293 [1] Emmanuel Abbe. Graph compression: The effect of clusters. In *2016 54th Annual Allerton
294 Conference on Communication, Control, and Computing (Allerton)*, pages 1–8, 2016. doi:
295 10.1109/ALLERTON.2016.7852203.
- 296 [2] Edo M Airoldi, David Blei, Stephen Fienberg, and Eric Xing. Mixed membership
297 stochastic blockmodels. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou,
298 editors, *Advances in Neural Information Processing Systems*, volume 21. Curran As-
299 sociates, Inc., 2009. URL <https://proceedings.neurips.cc/paper/2008/file/8613985ec49eb8f757ae6439e879bb2a-Paper.pdf>.
- 300 [3] Christophe Andrieu and Gareth O. Roberts. The pseudo-marginal approach for efficient Monte
301 Carlo computations. *The Annals of Statistics*, 37(2):697 – 725, 2009. doi: 10.1214/07-AOS574.
302 URL <https://doi.org/10.1214/07-AOS574>.
- 303 [4] Solenne Gaucher, Olga Klopp, and Geneviève Robin. Outliers detection in networks with
304 missing links, 2020.
- 305 [5] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications.
306 *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL <http://www.jstor.org/stable/2334940>.
- 307 [6] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego net-
308 works. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, ed-
309 itors, *Advances in Neural Information Processing Systems*, volume 25. Curran As-
310 sociates, Inc., 2012. URL <https://proceedings.neurips.cc/paper/2012/file/7a614fd06c325499f1680b9896beedeb-Paper.pdf>.
- 311 [7] Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. Stochastic blockmodels meet graph
312 neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the
313 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine
314 Learning Research*, pages 4466–4474. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/mehta19a.html>.
- 315 [8] Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic block-
316 structures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi:
317 10.1198/016214501753208735. URL <https://doi.org/10.1198/016214501753208735>.
- 318 [9] Boris Pasternak and Ivor Ivask. Four unpublished letters. *Books Abroad*, 44(2):196–200, 1970.
319 ISSN 00067431. URL <http://www.jstor.org/stable/40124305>.
- 320 [10] Tiago P. Peixoto. Parsimonious module inference in large networks. *Physical Review Letters*,
321 110(14), Apr 2013. ISSN 1079-7114. doi: 10.1103/physrevlett.110.148701. URL <http://dx.doi.org/10.1103/PhysRevLett.110.148701>.
- 322 [11] Tiago P. Peixoto. Efficient monte carlo and greedy heuristic for the inference of stochastic
323 block models. *Physical Review E*, 89(1), Jan 2014. ISSN 1550-2376. doi: 10.1103/physreve.89.
324 012804. URL <http://dx.doi.org/10.1103/PhysRevE.89.012804>.
- 325 [12] Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare.
326 1164194. URL http://figshare.com/articles/graph_tool/1164194.
- 327 [13] Tiago P. Peixoto. Nonparametric bayesian inference of the microcanonical stochastic block
328 model. *Physical Review E*, 95(1), Jan 2017. ISSN 2470-0053. doi: 10.1103/physreve.95.012317.
329 URL <http://dx.doi.org/10.1103/PhysRevE.95.012317>.
- 330 [14] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of Langevin distributions
331 and their discrete approximations. *Bernoulli*, 2(4):341 – 363, 1996. doi: [bj/1178291835](http://dx.doi.org/10.1178291835). URL
332 <https://doi.org/>.

- 338 [15] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton,
339 Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems.
340 High-resolution measurements of face-to-face contact patterns in a primary school. *PLOS ONE*,
341 6(8):1–13, 08 2011. doi: 10.1371/journal.pone.0023176. URL <https://doi.org/10.1371/journal.pone.0023176>.
- 343 [16] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynam-
344 ics. In *Proceedings of the 28th International Conference on International Conference on*
345 *Machine Learning*, ICML’11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN
346 9781450306195.
- 347 [17] Jun Zhu, Jiaming Song, and Bei Chen. Max-margin nonparametric latent feature models for
348 link prediction, 2016.

349 **Checklist**

- 350 1. For all authors...
- 351 (a) Do the main claims made in the abstract and introduction accurately reflect the paper's
352 contributions and scope? [Yes] See experiments section 5
- 353 (b) Did you describe the limitations of your work? [Yes] See conclusion 6 and experiments
354 5 sections
- 355 (c) Did you discuss any potential negative societal impacts of your work? [Yes] See
356 conclusion 6
- 357 (d) Have you read the ethics review guidelines and ensured that your paper conforms to
358 them? [Yes]
- 359 2. If you are including theoretical results...
- 360 (a) Did you state the full set of assumptions of all theoretical results? [Yes] See inference
361 section 4
- 362 (b) Did you include complete proofs of all theoretical results? [Yes] See inference section
363 4
- 364 3. If you ran experiments...
- 365 (a) Did you include the code, data, and instructions needed to reproduce the main experi-
366 mental results (either in the supplemental material or as a URL)? [Yes] See supplemen-
367 tary code
- 368 (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they
369 were chosen)? [Yes] See appendix A.9
- 370 (c) Did you report error bars (e.g., with respect to the random seed after running experi-
371 ments multiple times)? [Yes] See table 1
- 372 (d) Did you include the total amount of compute and the type of resources used (e.g., type
373 of GPUs, internal cluster, or cloud provider)? [Yes] See appendix A.10
- 374 4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
- 375 (a) If your work uses existing assets, did you cite the creators? [Yes] See section experi-
376 ments 5
- 377 (b) Did you mention the license of the assets? [Yes] See section inference 4
- 378 (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
379 Supplementary material
- 380 (d) Did you discuss whether and how consent was obtained from people whose data you're
381 using/curating? [Yes] Referred to original papers
- 382 (e) Did you discuss whether the data you are using/curating contains personally identifiable
383 information or offensive content? [Yes] Referred to original papers
- 384 5. If you used crowdsourcing or conducted research with human subjects...
- 385 (a) Did you include the full text of instructions given to participants and screenshots, if
386 applicable? [N/A]
- 387 (b) Did you describe any potential participant risks, with links to Institutional Review
388 Board (IRB) approvals, if applicable? [N/A]
- 389 (c) Did you include the estimated hourly wage paid to participants and the total amount
390 spent on participant compensation? [N/A]

391 **A Appendix**

392 **A.1 SBM prior choice explanation**

393 Here we recall for reference the priors $p(\psi|b)$ from [13]:

$$p(\psi_e = e, \psi_k = k|b) = p(e|b)p(\psi_k|e, b) = \left[\begin{Bmatrix} \binom{n}{2} \\ E \end{Bmatrix} \right]^{-1} \cdot \left[\prod_r \frac{\prod_j \eta_j^r!}{n_r! q(e_r, n_r)} \right], \quad (27)$$

394 where $\binom{n}{m}$ is shorthand for $\binom{n+m-1}{m} = \frac{(n+m-1)!}{(n-1)!(m)!}$, which can be thought of as the total number
 395 of distinct histograms produced by m samples in n bins. The value $E = \frac{1}{2} \sum_{r,s} e_{rs}$ is the total
 396 number of edges in the graph. Importantly, E is not allowed to vary and so $p(e|b)$ is uniform in
 397 e . The variable η_j^r denotes the number of vertices in block r that have degree j ; formally, $\eta_j^r :=$
 398 $\sum_i \mathbb{1}\{b_i = r\} \mathbb{1}\{k_i = j\}$. The denominator $q(m, n)$ denotes the number of different histograms
 399 produced by m samples in at most n non-zero bins that sum to m . Finally, $e_r := \sum_s e_{rs}$ is the total
 400 number of half edges in block r and $n_r := \sum_i \mathbb{1}\{b_i = r\}$ is the number of vertices assigned to block
 401 r .

402 These were chosen carefully in [13] to more closely match the structure of empirical networks than
 403 simple uniform priors. We do not repeat these arguments here.

404 **A.2 Derivation of conditional block distribution given feature matrix**

405 We determine the form of $p(b|X)$ by integrating out the parameters θ . From the definitions we have:

$$\begin{aligned} p(b|X) &= \int p(b, \theta|X, \theta) d\theta = \int p(b|X, \theta)p(\theta|X) d\theta \\ &= \int p(b|X, \theta)p(\theta) d\theta = \int \prod_{i \in [N]} \phi_{b_i}(x_i; \theta) p(\theta) d\theta \\ &= \prod_{i \in [N]} \int \frac{\exp(w_{b_i}^T \tilde{x}_i) \prod_{j \in [B]} \mathcal{N}(w_j; 0, \sigma_\theta^2 I)}{\sum_{k \in [B]} \exp(w_k^T \tilde{x}_i)} dw_{1:B}. \end{aligned}$$

406 The key observation here is that the value of the integral is independent of the value of $b_i \in [B]$: The
 407 integrand has the same form regardless of b_i , because the prior is the same for each w_j . Therefore,
 408 the integral is only a function of \tilde{x}_i and σ_θ^2 , which means that, as function of b , $p(b|X) \propto 1$. And as b
 409 takes values in $[B]^N$, we necessarily have:

$$p(b|X) = \frac{1}{|[B]^N|} = B^{-N}. \quad (28)$$

410 **A.3 Derivation of $U(\theta)$**

Recall from (15) is Section 4.2 that,

$$\pi_\theta(\theta) \propto p(\theta|X, b) \propto p(b|X, \theta)p(\theta) \propto \exp(-U(\theta)),$$

so that U can be expressed as,

$$U(\theta) = -(\log p(b|X, \theta) + \log p(\theta)) + \text{const.}$$

411 Writing, $y_{ij} := \mathbb{1}\{b_i = j\}$ and $a_{ij} := \phi_j(x_i; \theta)$, we have,

$$\log p(b|X, \theta) = \sum_{i \in [N]} \sum_{j \in [B]} y_{ij} \log a_{ij} \quad \text{and} \quad \log p(\theta) = -\frac{(D+1)(B)}{2} \log 2\pi - \frac{1}{2\sigma_\theta^2} \|\theta\|^2, \quad (29)$$

412 where $\|\theta\|^2 = \sum_i \theta_i^2 = \sum_{j=1}^B \|w_j\|^2$ is the Euclidean norm of the vector of parameters θ . Therefore,
 413 discarding constant terms, we obtain exactly the representation (16), as claimed.

414 **A.4 Derivation of $\nabla U(\theta)$**

415 Here we show how the gradient $\nabla U(\theta)$ can be computed explicitly. Recall the expression for $U(\theta)$
 416 in (16). Writing θ as $\theta = [w_1^T, w_2^T \dots w_B^T]^T$, in order to compute the gradient $\nabla U(\theta)$ we need to
 417 compute each of its components, $\partial U / \partial w_k$, $1 \leq k \leq B$. To that end, we first compute,

$$\begin{aligned} \frac{\partial a_{ij}}{\partial w_k} &= \frac{\tilde{x}_i \exp(w_j^T \tilde{x}_i) \delta_{jk} \cdot \sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) - \exp(w_j^T \tilde{x}_i) \cdot \tilde{x}_i \exp(w_k^T \tilde{x}_i)}{\left(\sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) \right)^2} \\ &= \tilde{x}_i (a_{ij} \delta_{jk} - a_{ij} a_{ik}), \end{aligned} \quad (30)$$

418 where $\delta_{jk} := \mathbb{1}\{j = k\}$, and we also easily find,

$$\frac{\partial}{\partial w_k} \|\theta\|^2 = \frac{\partial}{\partial w_k} \left(\sum_{r \in [B]} \|w_r\|^2 \right) = 2w_k. \quad (31)$$

419 Using (30) and (31), we obtain,

$$\begin{aligned} \frac{\partial U}{\partial w_k} &= \sum_{i=1}^N \sum_{j=1}^B y_{ij} \left(-\frac{\tilde{x}_i}{a_{ij}} (a_{ij} \delta_{jk} - a_{ij} a_{ik}) \right) + \frac{w_k}{\sigma_\theta^2} \\ &= - \left(\sum_{i=1}^N \tilde{x}_i \left(y_{ik} - a_{ik} \sum_{j=1}^B y_{ij} \right) - \frac{w_k}{\sigma_\theta^2} \right) \\ &= - \left(\sum_{i=1}^N \left\{ \tilde{x}_i (y_{ik} - a_{ik}) \right\} - \frac{w_k}{\sigma_\theta^2} \right). \end{aligned} \quad (32)$$

420 This can be computed efficiently through matrix operations. The only property of y_{ij} we have used
 421 in the derivation is the constraint $\sum_{j=1}^B y_{ij} = 1$, for all i .

422 **A.5 Choosing the MALA step-size**

423 Recall that in Section 4.2 we used the Metropolis Adjusted Langevin Algorithm (MALA) in order to
 424 sample from the θ -chain of the block membership generator parameters. At iteration t , the proposed
 425 sample is generated by:

$$\theta' = \theta^{(t)} - h_t \nabla U(\theta^{(t)}) + \sqrt{2h_t} \cdot \xi. \quad (33)$$

426 There are two competing objectives when choosing the step-size h_t . On the one hand, h_t needs to be
 427 large so that the sampler arrives at a high density region quickly, while too large a step-size would
 428 lead to low acceptance rates and thus inefficient sampling. An effective strategy is to use *simulated
 annealing*: allow h_t to slowly decrease with t , as long as $h_t > 0$ for all t and also:

$$\sum_{t=1}^{\infty} h_t = \infty, \quad \text{and} \quad \sum_{t=1}^{\infty} h_t^2 < \infty. \quad (34)$$

430 Following Welling and Teh [16], we adopt the polynomially decaying step-sizes, $h_t = \alpha(\beta + t)^{-\gamma}$,
 431 where $\alpha > 0$, $\beta > 0$ and $\gamma \in (1/2, 1]$ are hyper-parameters. We make the specific choices,

$$\alpha = \frac{250 \cdot s}{N}, \quad \beta = 1000, \quad \gamma = 0.8, \quad (35)$$

432 where N is the number of data-points and s , the *step-size scaling*, is the only free parameter.

433 **A.6 Burn-in and thinning**

434 When sampling from the b - and θ -chains described in Section 4, we generate T_b and T_θ samples
 435 total, respectively. We discard an initial proportion $\kappa_* \in (0, 1)$ of the samples as corresponding to a

436 “burn-in” period required for the distribution of the chain to reach a distribution close to our target,
 437 and we also “thin” the remaining samples to obtain a less-dependent version. For $\star \in \{b, \theta\}$, the
 438 remaining sample sets are denoted \mathcal{T}_\star in the notation of Section 4.3,

$$\mathcal{T}_\star = \{T_\star \kappa_\star + i \lambda_\star : 0 \leq i \leq \lfloor T_\star (1 - \kappa_\star) / \lambda_\star \rfloor\}, \quad (36)$$

439 where λ_\star controls the thinning. The choice of κ_\star can be determined by plotting the log-target (either
 440 $S(b^{(t)})$ or $U(\theta^{(t)})$) as a function of t , and choosing κ_\star to encompass the region where the log-target
 441 has roughly reached equilibrium. As we do not leverage sample independence, λ_\star can be chosen less
 442 rigorously; we often simply use $\lambda_b = 5$ and $\lambda_\theta = 10$.

443 **A.7 Initializing the b-chain**

444 For the purposes of the FFBM model, the number of blocks B is a constant which must be specified.
 445 If the choice of B is influenced by the observed data, then the analysis is no longer “fully Bayesian”
 446 and belongs to the class of methods referred to as “empirical Bayes.” However, as the number of
 447 blocks only specifies the coarseness of the analysis, it is reasonable to allow it to vary. Indeed,
 448 Peixoto [10] shows that for a fixed average degree the maximum number of detectable blocks scales
 449 as $O(\sqrt{N})$ where N is the number of vertices.

450 If B is allowed to vary in the b -chain (i.e., when new blocks can be created and empty blocks are
 451 allowed), then the chain can be run until a minimum description length (MDL) solution is reached.
 452 We take the number of non-empty blocks in the MDL solution to be our fixed block number B
 453 for subsequent analysis. Indeed, it is prudent to start the b -chain at this MDL solution as then the
 454 necessary burn-in time can be greatly reduced.

455 **A.8 Algorithms**

Algorithm 1 Block membership sample generation

```

 $b^{(0)} \leftarrow \arg \min_b S(b|A)$                                  $\triangleright$  Implemented as greedy heuristic in graph-tool library
for  $t \in \{0, 1 \dots T_b - 1\}$  do
     $b' \leftarrow \sim q_b(b^{(t)}, b'|A)$ 
     $\log \alpha_b \leftarrow \log \alpha_b(b^{(t)}, b'|A)$ 
     $\eta \leftarrow \sim \text{Unif}(0, 1)$ 
    if  $\log \eta < \log \alpha_b$  then
         $b^{(t+1)} \leftarrow b'$ 
    else
         $b^{(t+1)} \leftarrow b^{(t)}$ 
    end if
end for
return  $\{b^{(t)}\}_{t=1}^{T_b}$ 

```

Algorithm 2 FFBM parameter pseudo-marginal inference

$$\hat{Y}_{ij} \leftarrow \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} \mathbb{1}\{b_i^{(t)} = j\} \quad \forall i, j$$

$$\theta^{(0)} \leftarrow \sim \mathcal{N}(0, \sigma_\theta I)$$

```

for  $t \in \{0, 1 \dots T_\theta - 1\}$  do
     $\xi \leftarrow \sim \mathcal{N}(0, I)$ 
     $\theta' \leftarrow \theta^{(t)} - h_t \nabla U(\theta^{(t)} | X, \hat{Y}) + \sqrt{2h_t} \cdot \xi$ 
     $\log \alpha_\theta \leftarrow \log \alpha_\theta(\theta^{(t)}, \theta' | A, \hat{Y})$ 
     $\eta \leftarrow \sim \text{Unif}(0, 1)$ 
    if  $\log \eta < \log \alpha_\theta$  then
         $\theta^{(t+1)} \leftarrow \theta'$ 
    else
         $\theta^{(t+1)} \leftarrow \theta^{(t)}$ 
    end if
end for
return  $\{\theta^{(t)}\}_{t=1}^{T_\theta}$ 
```

456 **A.9 Hyperparameter values**

Table 2: Hyper-parameter values for each experiment.

Dataset	B	f	σ_θ	T_b	κ_b	λ_b	T_θ	κ_θ	λ_θ	s	k	D'	T'_θ	κ'_θ	λ'_θ	s'
Polbooks	3	0.7	1	1,000	0.2	5	10,000	0.4	10	0.05	—	—	—	—	—	—
School	10	0.7	1	1,000	0.2	5	10,000	0.4	10	0.2	1	10	10,000	0.4	10	0.2
FB Egonet	10	0.7	1	1,000	0.2	5	10,000	0.4	10	0.017	1	10	10,000	0.4	10	0.5

457 **A.10 Implementation details**

458 All data analysis and visualisation was implemented in Python. Full source code is available in the
459 supplementary material. The scripts were run using a standard PC using the Windows Subsystem for
460 Linux (WSL) environment. Specs are:

- 461 • **CPU:** Intel(R) Core(TM) i7-1065G7
462 • **RAM:** 8GB
463 • **GPU:** Intel(R) Iris(R) Plus Graphics

464 On this hardware each experiment iteration took the following amount of time to execute:

Table 3: Compute-time for each experiment.

Dataset	b -chain	θ -chain	Reduced θ -chain	Overall compute time
Polbooks	~1s	~4s	—	~5s
School	~10s	~10s	~10s	~30s
FB Egonet	~20s	~180s	~10s	~210s