
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 (called 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 (SBM) – is then used to
7 test 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 A somewhat surprising property of many real-world networks is that they exhibit strong community
26 structure; most nodes often belong to a densely connected cluster. There is high interest in recovering
27 the latent communities from the observed graphs. The inferred communities can be exploited for
28 compression algorithms [1] or used for link prediction in incomplete networks [4] to name a few
29 applications.

30 We restrict our analysis to vertex-labelled networks. We shall refer to the vertex-labels as features. A
31 common goal is to determine whether a given feature impacts graphical structure. To answer this
32 from a Bayesian perspective we must use a random graph model; the standard is the stochastic block
33 model (SBM) [8]. This is a latent variable model where each vertex belongs to a single block and the
34 probability two nodes are connected depends only on the block memberships of each. There have

35 been many variants to this model – the most popular being the mixed-membership stochastic block
 36 model (MMSBM) [2] and the overlapping stochastic block model (OSBM) [18]. Effectively, these
 37 just extend the model to allow each vertex to belong to multiple blocks simultaneously. However,
 38 a major drawback of these graphical models as applied to labelled networks is that they do not
 39 automatically include vertex features in the random graph generation process. Approaches based on
 40 graph neural networks [7] that utilise vertex features have been developed but these lack the easy
 41 interpretability of the simpler models.

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

50 With these desiderata in mind, we present a novel framework for modelling labelled networks, which
 51 we call the feature-first block model (FFBM). This is an extension of the SBM to labelled networks.
 52 We go on to present an efficient algorithm for sampling from the parameters of the feature-to-block
 53 generator. We can interpret the sampled FFBM parameters to determine which features have the
 54 largest impact on overall graphical structure.

55 2 Preliminaries

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

62 **Definition 2.1 (Microcanonical DC-SBM)** Let $N \in \mathbb{Z}^+$ denote the number of vertices in an undi-
 63 rected graph. The block memberships are encoded by a vector $b \in [B]^N$ ¹, where $B \in \mathbb{Z}^+$ is the
 64 number of non-empty blocks. Let $e \in (\mathbb{Z}_0^+)^{N \times N}$ be the matrix of edge counts between blocks. e_{rs} is
 65 then the number of edges from block r onto block s – or twice that number if $r = s$. For undirected
 66 graphs, e is symmetric. Let $k \in (\mathbb{Z}_0^+)^N$ be a vector denoting the degree sequence of the graph. k_i is
 67 then the degree of vertex i . The graph's adjacency matrix $A \in \{0, 1\}^{N \times N}$ is generated as follows:

$$A \sim DC-SBM_{MC}(b, e, k) \quad (1)$$

68 Where edges are placed uniformly at random but respecting the constraints imposed by e , b and k –
 69 hence the microcanonical moniker. Specifically, A must satisfy the following:

$$e_{rs} = \sum_{i,j \in [N]} A_{ij} \mathbb{1}\{b_i = r\} \mathbb{1}\{b_j = s\} \quad \forall r, s \in [B] \quad \text{and} \quad k_i = \sum_{j \in [N]} A_{ij} \quad \forall i \in [N] \quad (2)$$

70 3 Feature-first block model

71 In this section we propose a novel generative model for labelled networks. We call this the feature-first
 72 block model (FFBM) and outline its structure in 1 As before, we let N denote the number of nodes
 73 and B the number of blocks in our graph. We define the vector $x_i \in \mathcal{X}^D$ as the feature vector for the
 74 i 'th vertex. D is the number of features. For the datasets we analyse, we deal with binary feature

¹We introduce the notation $[K] := \{1, 2 \dots K\}$ to compactly define a set of K indices. Clearly, $[K]$ is only defined for $K \in \mathbb{Z}^+$.

75 flags so $\mathcal{X} = \{0, 1\}$. The feature vectors $\{x_i\}_{i=1}^N$ may be compactly subsumed into the feature matrix
 76 $X \in \mathcal{X}^{N \times D}$.

77 For the FFBM, we start with the feature matrix X and probabilistically generate a vector of block
 78 memberships $b \in [B]^N$. The parameters of this step are encapsulated by θ . Each feature vector x_i is
 79 treated independently and used to generate the corresponding block membership $b_i \in [B]$. We choose
 80 a single softmax layer to model $p(b_i|x_i, \theta)$. More complex models are possible but then deriving
 81 meaning from the inferred parameter distributions is more difficult. Summarising, we write $p(b|X, \theta)$
 82 as follows:

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

83 We deliberately exclude a bias term to ensure that the relationships we model are based on features
 84 and not information about the size of each detected block; a more complete discussion on this topic
 85 is given in A.5. The parameter vector θ for this stage contains all the weight vectors $\theta = \{w_k\}_{k=1}^B$.
 86 Each w_k has dimension D . We could instead write the parameters θ as a $B \times D$ matrix of weights W ;
 87 this form has computational benefits as then $z_i := Wx_i$, which is the input to the softmax activation
 88 function.

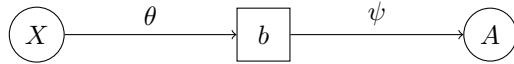


Figure 1: The feature-first block model (FFBM)

89 Once the block memberships b have been generated, we then draw the graph A from the microcanonical
 90 DC-SBM (equation 4) with additional parameters encapsulated by $\psi = \{\psi_e, \psi_k\}$.

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

91 3.1 Prior selection

92 Before performing any inference, we must specify priors on θ and ψ . For θ it seems sensible to
 93 choose a Gaussian prior, with zero mean and variance matrix $\sigma_\theta^2 I$ such that each element of θ is
 94 independent and distributed like $\mathcal{N}(0, \sigma_\theta^2)$. In vector form, the prior for θ is therefore:

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

95 In our model, the block memberships vector b is an intermediate latent variable and so we are not
 96 free to choose a prior for it. Nevertheless, as far as inference on the right-hand-side of figure 1, we
 97 regard $p(b|X)$ as a pseudo-prior on b . We can show (appendix B.1) that our choice of prior for $p(\theta)$
 98 in equation 5 leads to a uniform $p(b|X)$ in equation 6.

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

99 This is an enormously important simplification as evaluating $p(b|X)$ does not require an expensive
 100 Monte-Carlo integration over the θ -domain nor does it require the exact value of X . Peixoto
 101 [13] proposes careful choices for the additional microcanonical SBM parameters ψ which we
 102 adopt. Peixoto's idea is to write the joint prior on (b, e, k) as a product of conditionals $p(b, e, k) =$
 103 $p(b)p(e|b)p(k|e, b) = p(b)p(\psi|b)$. For our purposes we must insert a conditioning on X , to form our
 104 pseudo-prior for b and ψ , to give equation 7.

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

105 Where we leverage the fact $(\psi \perp\!\!\!\perp X)|b$. We then borrow the priors proposed by Peixoto [13] for
 106 $p(\psi|b)$ to complete our model. Please refer to appendix A.1 for the exact form of $p(\psi|b)$. All that
 107 concerns the main argument is we have a computable form.

108 **4 Inference**

109 Now that we have defined the FFBM, we wish to leverage it to perform inference. Suppose we are
110 presented with a vertex-labelled graph (A, X) ; the goal is to draw samples for θ according to the
111 posterior given the observed data:

$$\theta^{(t)} \sim p(\theta|A, X) \quad (8)$$

112 However, generating these samples is not easily done in practice. We therefore propose an iterative
113 approach. We first draw samples $b^{(t)}$ from the block membership posterior (equation 9) and then use
114 each $b^{(t)}$ to draw samples for θ as in equation 10.

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

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

115 Both of these sampling steps can be implemented with a Markov Chain through the Metropolis-
116 Hastings algorithm [5]. We just need to define a proposal distribution $q(x, x')$ for proposing a move
117 $x \rightarrow x'$ and be able to evaluate an un-normalised form of the target distribution, denoted $\pi(\cdot)$,
118 point-wise. The proposed move is then accepted with probability α (equation 11) else it is rejected
119 and we stay at x .

$$\alpha(x, x') = \min \left(\frac{\pi(x')q(x', x)}{\pi(x)q(x, x')}, 1 \right) \quad (11)$$

120 This accept-reject step ensures the resulting Markov Chain is in detailed balance with the target
121 distribution $\pi(\cdot)$. What we propose in equations 9 and 10 is therefore implemented through a 2-level
122 Markov chain. The resulting samples for $\theta^{(t)}$ are unbiased in the sense that the expectation of their
123 distribution is the posterior we are targeting:

$$\mathbb{E}_{b^{(t)}} \left[p(\theta|X, b^{(t)}) \right] = \sum_{b \in [B]^N} p(\theta|X, b)p(b|A, X) = \sum_{b \in [B]^N} p(\theta, b|A, X) = p(\theta|A, X) \quad (12)$$

124 This is an example of a pseudo-marginal approach. Indeed, Andrieu and Roberts [3] show that the un-
125 biased result in equation 12 is sufficient to prove that for large enough t , $\theta^{(t)} \sim \mathbb{E}_{b^{(t)}} [p(\theta|X, b^{(t)})] = p(\theta|A, X)$ which is exactly the distribution we are targeting (equation 8).

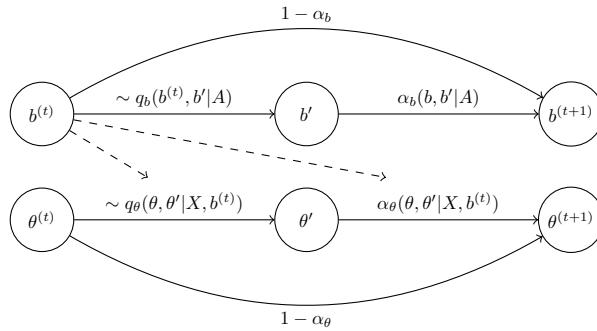


Figure 2: Sampling sequence

126
127 The reason we split the Markov chain into two stages is because the summation over all latent states
128 $b \in [B]^N$ required to directly compute the likelihood $p(A|X, \theta) = \sum_{b \in [B]^N} p(A|b)P(b|X, \theta)$ is
129 intractable – $O(B^N)$. Figure 2 shows an overview of the proposed method. We have introduced
130 subscripts and conditionings to make explicit what variables each step utilises. We note the power of
131 the simplification given by equation 6. As $p(b|X)$ does not depend on the exact value of X , we do
132 not need to know the value of X to perform the sampling on b . Conversely, for the $\theta^{(t)}$ samples, we
133 use only $b^{(t)}$ but not A as $(\theta \perp\!\!\!\perp A)|b$.

134 **4.1 Sampling block memberships**

135 Peixoto [11] proposes a Monte Carlo method which we will base our approach on. It relies on writing
 136 the posterior in the following form:

$$p(b|A, X) \propto p(A|b, X) \cdot p(b|X) = \pi_b(b) \quad (13)$$

137 Now $\pi_b(\cdot)$ is the un-normalised density we wish to sample from for the b -chain. In other words, we
 138 wish to construct a Markov chain that has $\pi_b(\cdot)$ as its invariant distribution. We can break π_b down as
 139 follows:

$$\pi_b(b) = p(b|X) \sum_{\psi} p(A, \psi|b, X) = p(b|X)p(A, \psi^*|b, X) = p(A|b, \psi^*) \cdot p(\psi^*|b) \cdot p(b|X) \quad (14)$$

140 Since we are using the microcanonical SBM formulation, there is only one value of ψ that is
 141 compatible with the given (A, b) pair (given in equation 2). We denote this value $\psi^* = \{\psi_k^*, \psi_e^*\}$.
 142 Therefore, the summation over all ψ reduces to just the single ψ^* term; this is the power of the
 143 microcanonical formulation. We also define the microcanonical entropy of the configuration as.

$$S(b) = -\log \pi_b(b) = -\left(\log p(A|b, \psi^*) + \log p(\psi^*, b|X) \right) \quad (15)$$

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

149 **4.2 Sampling feature-to-block generator parameters**

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

$$\pi_\theta(\theta) \propto p(\theta|X, b) \propto p(b|X, \theta)p(\theta) \propto \pi_\theta(\theta) \propto \exp(-U(\theta)) \quad (16)$$

152 Where we have introduced $U(\theta)$ equal to the negative log posterior. We define $y_{ij} := \mathbb{1}\{b_i = j\}$ and
 153 $a_{ij} := \phi_j(x_i; \theta)$. Discarding constant terms, we can write $U(\theta)$ as in equation 17 (refer to appendix
 154 B.2 for the derivation).

$$U(\theta) = \left(\sum_{i=1}^N \sum_{j=1}^B y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 = N \cdot \mathcal{L}(\theta) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (17)$$

155 $U(\theta)$ in equation 17 appears a typical objective function for neural network training. The first term
 156 is introduced by the likelihood. We collect it into $N \cdot \mathcal{L}(\theta)$, which is the cross-entropy between the
 157 graph-predicted and feature-predicted block memberships summed over all vertices. The second
 158 term of equation 17 – introduced by the prior – brings a form of regularisation, guarding against
 159 over-fitting. Different to traditional applications, our goal is not to find the minimiser of $U(\theta)$ but to
 160 draw samples from the posterior $\pi_\theta(\cdot) \propto \exp(-U(\cdot))$. We can use ∇U as a useful heuristic to bias
 161 our proposal towards regions of higher target density. We therefore adopt the Metropolis-adjusted
 162 Langevin algorithm (MALA) – first proposed by Roberts and Tweedie [15]. Given the current sample
 163 θ , we generate a new sample θ' according to equation 18.

$$\theta' = \theta - h \nabla U(\theta) + \sqrt{2h} \cdot \xi \quad (18)$$

$$\therefore q_\theta(\theta, \theta') = \mathcal{N}(\theta'; \theta - h \nabla U(\theta), 2hI) \quad (19)$$

164 Where $\xi \sim \mathcal{N}(0, I)$ and h is a step-size parameter – which may vary with the sample index (appendix
 165 A.2 explores this more fully). Without the injected noise term, MALA is equivalent to gradient
 166 descent. We require the noise term ξ to fully explore the parameter space. We can write the proposal
 167 distribution q_θ as in equation 19. The term ∇U has an easy to compute analytic form (derived in
 168 Appendix B.3). By noting that $\theta = \{w_k\}_{k=1}^B$, we write the derivative with respect to each w_k as:

$$\frac{\partial U}{\partial w_k} = - \left(\sum_{i=1}^N \left\{ \tilde{x}_i (y_{ik} - a_{ik}) \right\} - \frac{w_k}{\sigma_\theta^2} \right) \quad (20)$$

169 After a proposed move is generated, in typical Metropolis-Hastings fashion we accept the move with
 170 probability α_θ , as in equation 11.

171 **4.3 Sampling sequence**

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

$$\mathbb{E} [y_{ij}^{(t)}] = \mathbb{E}_{b^{(t)}} [\mathbb{1}(b_i^{(t)} = j)] = p(b_i = j | A, X) \quad (21)$$

177 We can obtain an unbiased estimate for this quantity using the set of b -samples. However, as with
 178 all MCMC methods, we must only use samples after burn-in and thinning have been applied. We
 179 introduce \mathcal{T}_b to denote the retained set of indices for the b -samples and \mathcal{T}_θ similarly for the θ -chain.
 180 An in-depth discussion of how these sets are chosen is given in appendix A.3. The unbiased estimate
 181 for $y_{ij}^{(t)}$ using the restricted sample set \mathcal{T}_b is denoted \hat{y}_{ij} and has form:

$$\hat{y}_{ij} := \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} y_{ij}^{(t)} = \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} \mathbb{1}\{b_i^{(t)} = j\} \quad (22)$$

182 We choose to feed each $\theta^{(t)}$ update step the same matrix \hat{y} for all t rather than the corresponding $y^{(t)}$.
 183 This means we no longer need to run the b and θ Markov chains concurrently. Instead, we run the
 184 b -chain to completion and use it to generate \hat{y} . This affords us the flexibility to vary the lengths of the
 185 b and θ -chains. Furthermore, the changeover from $y^{(t)}$ to \hat{y} reduces the burn-in time for the θ -chain
 186 by reducing the variance in our evaluation of U and ∇U . A description of the overall algorithms is
 187 given in appendix C.1.

188 **4.4 Dimensionality reduction**

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

$$\hat{\mu}_{ij} := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} W_{ij}^{(t)} \quad \text{and} \quad \hat{\sigma}_{ij}^2 := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} (W_{ij}^{(t)} - \hat{\mu}_{ij})^2 \quad (23)$$

192 A simple heuristic to discard the least important features requires specifying a cutoff $c > 0$ and a
 193 multiplier $k > 0$. We define the function $\mathcal{F}_i(j)$ as in 24 then only keep features with indices $d \in \mathcal{D}'$,
 194 where \mathcal{D}' is constructed as in equation 25.

$$\mathcal{F}_i(j) := (\hat{\mu}_{ij} - k\hat{\sigma}_{ij}, \hat{\mu}_{ij} + k\hat{\sigma}_{ij}) \cap (-c, +c) \quad (24)$$

$$\mathcal{D}' := \{j \in [D] : \exists i \in [B] \text{ s.t. } \mathcal{F}_i(j) = \emptyset\} \quad (25)$$

195 Intuitively, this means discarding any feature for which $\hat{\mu}_{ij} \pm k\hat{\sigma}_{ij}$ lies within or spans the null
 196 region $(-c, c)$ for all block indices. If we were to use the Laplace approximation for the posterior
 197 $p(W_{ij} | A, X) \approx \mathcal{N}(W_{ij}; \mu_{ij}, \sigma_{ij}^2)$, then this is analogous to a hypothesis test on the value of W_{ij}
 198 (equation 26). \mathcal{D}' then comprises all features i for which H_1 is accepted at least once for some
 199 $j \in [B]$.

$$H_0 : |W_{ij}| \leq c \quad H_1 : |W_{ij}| > c \quad (26)$$

200 The multiplier k in equation 24 determines the degree of significance of the result. Appendix B.4
 201 explores the derivation in more detail. However, as the Laplace approximation is not exact we
 202 will only treat this dimensionality reduction method as a useful heuristic and not an exact method.
 203 Conversely, we could fix $k = k_0$ and the dimension of our reduced feature set $|\mathcal{D}'| = D'$. We would
 204 then like to find the largest value of c such that $|\mathcal{D}'| = D'$ given $k = k_0$. This is summarised in
 205 equation 27. This approach is often preferred as it fixes the number of reduced dimensions.

$$c^* = \arg \max_{c>0} (c : |\mathcal{D}'| = D', k = k_0) \quad (27)$$

206 For an algorithmic implementation of this method refer to appendix C.1.

207 **5 Experiments**

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

- 210 • **Political books** [9] ($N = 105, E = 441, D = 3$) – network of Amazon book sales about U.S.
 211 politics, published close to the presidential election in 2004. Two books are connected if they were
 212 frequently co-purchased by customers. Vertex features encode the political affiliation of the author
 213 (liberal, conservative or neutral).
- 214 • **Primary school dynamic contacts** [16] ($N = 238, E = 5539, D = 13$) – network of face-to-face
 215 contacts amongst students and teachers at a primary school in Lyon, France. Two nodes are
 216 connected if the two parties shared a face-to-face interaction over the school-day. Vertex features
 217 include class membership (one of 10 values: 1A-5B), gender (male, female) and teacher status
 218 encoded as an 11th school-class. No further identifiable information is retained. We choose to
 219 analyse just the second day of results.
- 220 • **Facebook egonet** [6] ($N = 747, E = 30025, D = 480$) – an assortment of Facebook users'
 221 friends lists. Vertex features are extracted from each user's profile and are fully anonymised. They
 222 include information about education history, languages spoken, gender, home-town, birthday etc.
 223 We focus on the egonet with id 1912.

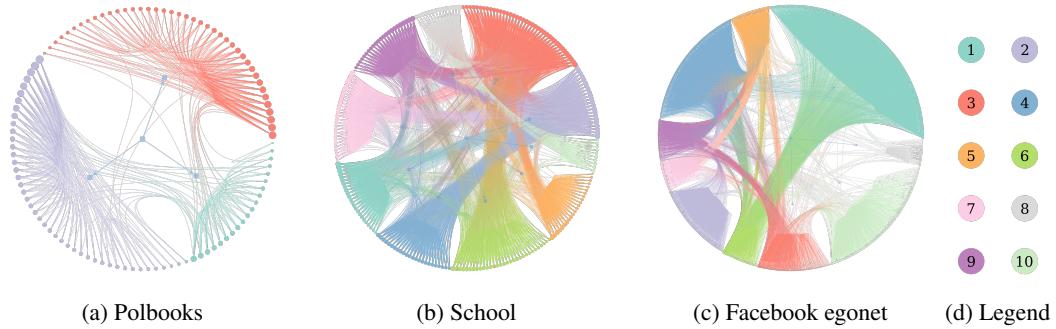


Figure 3: Networks laid out and coloured according by inferred block memberships \hat{y} for a given experiment iteration. Visualisation performed using *graph-tool* [12]

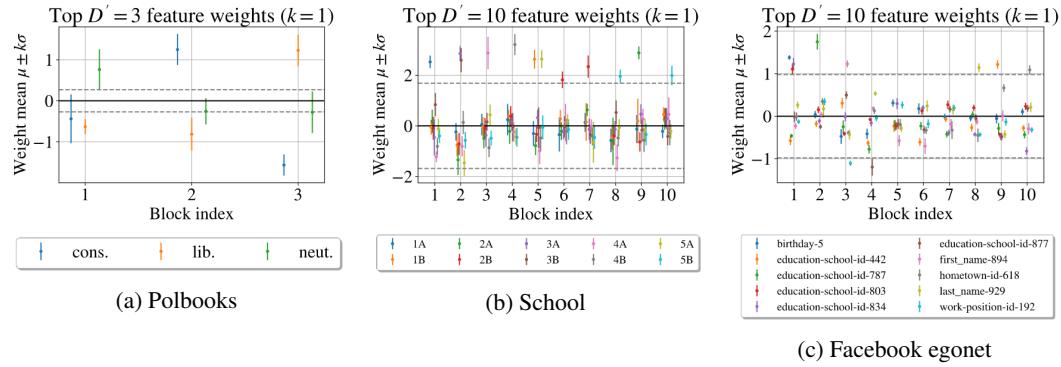


Figure 4: Reduced dimension feature-to-block generator weight samples

224 We require metrics to assess performance. This can be split into two separate components: the
 225 microcanonical SBM fit (concerned with the b -samples) and the fit of the feature-to-block generator
 226 (concerned with the θ -samples). Starting with the SBM, $S(b)$ (equation 15) can be interpreted as
 227 the description length of the partition imposed by b . It is only natural to divide this quantity by the
 228 number of entities (nodes and edges) in our graph $N + E$ to allow for rough comparison between

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

graphs. This defines a simple metric to gauge the fit of the SBM: the description length per entity averaged over the b -samples (equation 28):

$$\bar{S}_e := \frac{1}{(N+E)|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} S(b^{(t)}) \quad (28)$$

However, to assess the performance of the feature-to-block predictor, we must partition the vertex set $[N]$ into training and test sets. We choose to randomly partition the vertices on each experiment run such that a constant fraction f of the available vertices go to form our training set \mathcal{G}_0 and the remainder are held out to form our 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. As $|\mathcal{G}_0| \neq |\mathcal{G}_1|$ in general, we cannot use the un-normalised log target U (equation 17) for comparison as the total cross-entropy loss is scaled by the size of each set but the prior term stays constant. We therefore must use the average cross-entropy loss over each set (equation 29):

$$\bar{\mathcal{L}}_\star := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} \mathcal{L}_\star(\theta^{(t)}) \quad \text{where} \quad \mathcal{L}_\star(\theta^{(t)}) := \frac{1}{|\mathcal{G}_\star|} \sum_{i \in \mathcal{G}_\star} \sum_{j \in [B]} \hat{y}_{ij} \log \frac{1}{\phi_j(x_i; \theta^{(t)})} \quad (29)$$

Where $\star \in \{0, 1\}$ has been introduced to toggle between training and test sets. Table 1 summarises the results for each experiment.² We also apply the dimensionality reduction method on the two higher dimensional datasets (the school and FB egonet). For this we leverage equation 27, to reduce the dimension from D to D' with $k = 1$ to yield the maximal cutoff c^* . We then retrain the feature-block predictor using just the retained feature set D' and report the 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 included in table 1.

Table 1 already highlights some general trends in the results. 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 our test set is smaller than the training set and so 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 of the graph \bar{S}_e has very low variance implying 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 on 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

We wish to determine whether the author’s political affiliation is 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 is strong evidence that political affiliation is indeed the axis which best predicts the 3-way natural partition of the graph into blocks. Furthermore, from 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 the best explanatory variable for the overall network structure.

²For a comprehensive list of the hyper-parameters used for each experiment please see appendix C.2

266 **5.2 Primary school dynamic contacts**

267 We choose $B = 10$ in line with the total number of school-classes. As before, we sample the
268 block-generator parameters θ and employ the dimensionality reduction technique with standard
269 deviation multiplier $k = 1$ to pick out the top $D' = 10$ features. We then plot the weights for the
270 surviving features $d \in D'$ on figure 4b. Immediately, we see that only the pupils' class memberships
271 have survived (1A-5B); gender and teacher/student status have been discarded meaning that these are
272 not good predictors of overall macro-structure.

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

280 **5.3 Facebook egonet**

281 We choose $B = 10$ and $D' = 10$ for this experiment. The remaining features (figure 4c) are those
282 that best explain the high-level community structure. The majority of the surviving features are
283 education related. Nevertheless, for $D' = 10$ we only have good explanations for the makeup of
284 some of the detected blocks; several blocks in figure 4c do not have high-magnitude components for
285 $D' = 10$.

286 When the feature dimension is very large, it becomes increasingly likely that a particular feature may
287 uniquely identify a small set of nodes. If these nodes are all part of the same community then the
288 classifier will overfit for that particular parameter. The regularisation term imposed by the prior goes
289 some way to alleviating this problem. Nevertheless, we see in figure 4c that the feature `birthday-5`
290 has a very high weight as it relates to block 1 – but it would be preposterous to conclude that birthdays
291 determine graphical structure. The analyst must remain vigilant of such problems.

292 **6 Conclusion**

293 The FFBM was developed to address the shortcomings of other graphical models when testing how
294 vertex features affect community structure. The idea is to divide the graph into its most natural
295 partition and test whether the vertex features can accurately explain this partition. It is very easy to
296 find vertex features that are in some way correlated with the graphical structure. Nonetheless, only
297 when we find the feature that best describes the most pronounced partition do we have a stronger
298 case for causation.

299 With the newly-defined FFBM, we go on to present an efficient inference algorithm to sample the
300 parameters θ of the feature-to-block generator. This is introduced as two concurrent Markov chains
301 to sample the block memberships b and block generator parameters θ . Nevertheless, we can serialise
302 the chains and use the empirical mean of the b -samples as the input to our θ -chain. This reduces the
303 variance in our evaluation of the target distribution and thus shortens burn-in.

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

310 **References**

- 311 [1] Emmanuel Abbe. Graph compression: The effect of clusters. In *2016 54th Annual Allerton
312 Conference on Communication, Control, and Computing (Allerton)*, pages 1–8, 2016. doi:
313 10.1109/ALLERTON.2016.7852203.
- 314 [2] Edo M Airoldi, David Blei, Stephen Fienberg, and Eric Xing. Mixed membership
315 stochastic blockmodels. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou,
316 editors, *Advances in Neural Information Processing Systems*, volume 21. Curran As-
317 sociates, Inc., 2009. URL <https://proceedings.neurips.cc/paper/2008/file/8613985ec49eb8f757ae6439e879bb2a-Paper.pdf>.
- 319 [3] Christophe Andrieu and Gareth O. Roberts. The pseudo-marginal approach for efficient Monte
320 Carlo computations. *The Annals of Statistics*, 37(2):697 – 725, 2009. doi: 10.1214/07-AOS574.
321 URL <https://doi.org/10.1214/07-AOS574>.
- 322 [4] Solenne Gaucher, Olga Klopp, and Geneviève Robin. Outliers detection in networks with
323 missing links, 2020.
- 324 [5] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications.
325 *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL <http://www.jstor.org/stable/2334940>.
- 327 [6] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego net-
328 works. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, ed-
329 itors, *Advances in Neural Information Processing Systems*, volume 25. Curran As-
330 sociates, Inc., 2012. URL <https://proceedings.neurips.cc/paper/2012/file/7a614fd06c325499f1680b9896beedeb-Paper.pdf>.
- 332 [7] Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. Stochastic blockmodels meet graph
333 neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the
334 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine
335 Learning Research*, pages 4466–4474. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/mehta19a.html>.
- 337 [8] Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic block-
338 structures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi:
339 10.1198/016214501753208735. URL <https://doi.org/10.1198/016214501753208735>.
- 340 [9] Boris Pasternak and Ivor Ivask. Four unpublished letters. *Books Abroad*, 44(2):196–200, 1970.
341 ISSN 00067431. URL <http://www.jstor.org/stable/40124305>.
- 342 [10] Tiago P. Peixoto. Parsimonious module inference in large networks. *Physical Review Letters*,
343 110(14), Apr 2013. ISSN 1079-7114. doi: 10.1103/physrevlett.110.148701. URL <http://dx.doi.org/10.1103/PhysRevLett.110.148701>.
- 345 [11] Tiago P. Peixoto. Efficient monte carlo and greedy heuristic for the inference of stochastic
346 block models. *Physical Review E*, 89(1), Jan 2014. ISSN 1550-2376. doi: 10.1103/physreve.89.
347 012804. URL <http://dx.doi.org/10.1103/PhysRevE.89.012804>.
- 348 [12] Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare.
349 1164194. URL http://figshare.com/articles/graph_tool/1164194.
- 350 [13] Tiago P. Peixoto. Nonparametric bayesian inference of the microcanonical stochastic block
351 model. *Physical Review E*, 95(1), Jan 2017. ISSN 2470-0053. doi: 10.1103/physreve.95.012317.
352 URL <http://dx.doi.org/10.1103/PhysRevE.95.012317>.
- 353 [14] Jaakko Riihimäki and Aki Vehtari. Laplace approximation for logistic gaussian process density
354 estimation and regression, 2013.

- 355 [15] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of Langevin distributions
356 and their discrete approximations. *Bernoulli*, 2(4):341 – 363, 1996. doi: [bj/1178291835](https://doi.org/10.1177/0946162096410405). URL
357 [https://doi.org/](https://doi.org/10.1177/0946162096410405).
- 358 [16] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton,
359 Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems.
360 High-resolution measurements of face-to-face contact patterns in a primary school. *PLoS ONE*,
361 6(8):1–13, 08 2011. doi: [10.1371/journal.pone.0023176](https://doi.org/10.1371/journal.pone.0023176). URL <https://doi.org/10.1371/journal.pone.0023176>.
- 363 [17] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynam-
364 ics. In *Proceedings of the 28th International Conference on International Conference on*
365 *Machine Learning*, ICML’11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN
366 9781450306195.
- 367 [18] Jun Zhu, Jiaming Song, and Bei Chen. Max-margin nonparametric latent feature models for
368 link prediction, 2016.

369 **Checklist**

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

411 **A Appendix: Additional material**

412 **A.1 SBM likelihood and prior**

413 For reference, we provide the likelihood and priors for the microcanonical SBM proposed by Peixoto
414 [13]. We have that the graph A is drawn from the SBM:

$$A \sim \text{DC-SBM}_{\text{MC}}(b, e, k) \quad (30)$$

415 With edges placed uniformly at random but respecting the constraints imposed by b, e and k . The
416 likelihood calculation for $p(A|k, e, b)$ then reduces to a case of counting configurations that yield
417 the same adjacency matrix, $\Xi(A)$, and dividing by the total number of configurations possible, $\Omega(e)$.
418 This is why this formulation is given the microcanonical moniker. If we consider the half-edges to be
419 distinguishable for a moment, the total number of configurations that satisfy the e constraint is:

$$\Omega(e) = \frac{\prod_r e_r!}{\prod_{r,s:r < s} e_{rs}! \cdot \prod_r e_{rr}!!} \quad (31)$$

420 Where $e_r := \sum_s e_{rs}$ and $(2m)!! := 2^m m!$. Nevertheless, a great number of these configurations
421 yield the same graph A . We denote the number of configurations that yield the adjacency matrix A
422 with $\Xi(A)$, which can be computed as:

$$\Xi(A) := \frac{\prod_i k_i!}{\prod_{i,j:i < j} A_{ij}! \prod_i A_{ii}!!} \quad (32)$$

423 Note the similarity between the forms of $\Omega(e)$ and $\Xi(A)$ as e is effectively the adjacency matrix of
424 the block-graph. With these defined we can write the overall likelihood:

$$p(A|k, e, b) = \frac{\Xi(A)}{\Omega(e)} \quad (33)$$

425 Obviously, this form is only defined if A respects the constraints imposed by (k, e, b) else the
426 likelihood is 0. With the likelihood defined, we move on to the prior. As discussed in the main text,
427 the block membership vector b is an intermediate variable in the FFBM and not a parameter so we
428 are not free to choose a prior for it. Nevertheless, we can borrow the conditional prior proposed by
429 Peixoto [13] for the other parameters to give:

$$p(e, k|b) = p(e|b)p(k|e, b) = \left[\left\{ \begin{array}{c} \{B\} \\ E \end{array} \right\} \right]^{-1} \cdot \left[\prod_r \frac{\prod_j \eta_j^r!}{n_r! q(e_r, n_r)} \right] \quad (34)$$

430 Where $\left\{ \begin{array}{c} n \\ m \end{array} \right\}$ is shorthand for $\binom{n+m-1}{m} = \frac{(n+m-1)!}{(n-1)!(m)!}$ which can be thought of as the total number
431 of distinct histograms with n bins under the constraint they sum to m . $E = \frac{1}{2} \sum_{r,s} e_{rs}$ is the total
432 number of edges in the graph. Importantly, E is not allowed to vary and so $p(e|b)$ is uniform with
433 respect to e . The variable η_j^r is introduced to denote the number of vertices in block r that have degree
434 j . Formally, $\eta_j^r := \sum_i \mathbb{1}\{b_i = r\} \mathbb{1}\{k_i = j\}$. Furthermore, $q(m, n)$ is the number of different
435 histograms with at most n non-zero bins that sum to m . $q(m, n)$ is related to but different from $\left\{ \begin{array}{c} n \\ m \end{array} \right\}$.
436 Recall that $e_r := \sum_s e_{rs}$ is the total number of half edges in block r and $n_r := \sum_i \mathbb{1}\{b_i = r\}$ is the
437 number of vertices assigned to block r .

438 The form of these priors were chosen carefully by Peixoto [13] to more closely match the structure of
439 empirical networks than simple uniform priors. We do not repeat his arguments here.

440 **A.2 Choosing the MALA step-size**

441 For sampling from the θ -chain of the block membership generator parameters, we employ the
442 Metropolis Adjusted Langevin Algorithm (MALA). At iteration t , the proposed sample is generated
443 by:

$$\theta' = \theta^{(t)} - h_t \nabla U(\theta^{(t)}) + \sqrt{2h_t} \cdot \xi \quad (35)$$

444 There are two competing objectives when choosing the step-size h_t . On the one hand, we want the
 445 step-size to be large so that we arrive at a high density region quickly. However, too large a step-size
 446 will lead to a lower acceptance ratio and thus inefficient sampling. A solution to this problem would
 447 be to slowly decrease the step-size with t - often called simulated annealing. Therefore, we still have
 448 a short burn-in time but will not bounce around the mode for large t . As well as the trivial constraint
 449 for h_t to be strictly positive, we introduce two further constraints as outlined by Welling and Teh
 450 [17]:

$$\sum_{t=1}^{\infty} h_t = \infty \quad \text{and} \quad \sum_{t=1}^{\infty} h_t^2 < \infty \quad (36)$$

451 The first constraint ensures that we can cover sufficient distance to arrive at any arbitrary point in
 452 our domain, no matter the starting point. The second constraint ensures that once we converge to the
 453 mode we will stay there, rather than simply bouncing around it. Welling and Teh [17] propose the
 454 following form for a polynomially decaying step-size which we adopt:

$$h_t = \alpha(\beta + t)^{-\gamma} \quad (37)$$

455 Where α, β, γ are hyper-parameters to be chosen. We require $\alpha, \beta > 0$ and $\gamma \in (0.5, 1]$ to satisfy
 456 equation 36. To reduce the number of hyperparameters we set these to have values given by the
 457 equations in 38.

$$\alpha = \frac{250 \cdot s}{N} \quad \beta = 1000 \quad \gamma = 0.8 \quad (38)$$

458 Where N is the number of data-points we are considering and now s is the only free variable which
 459 we call the step-size scaling. For approximate methods, we can choose to bypass the MH accept-reject
 460 entirely to speed up computation. If this is done, the algorithm is instead called stochastic gradient
 461 Langevin diffusion (SGLD) [17]. This speeds up computation at the expense of the exactness of the
 462 method. Indeed, it relies on the fact that as the step-size is annealed to 0, the acceptance probability
 463 is close 1.

464 Nevertheless, to choose the value of s we must quantify the trade-off between acceptance ratio and
 465 burn-in time. We do this by way of an example with the primary school dataset [16]. We run the
 466 b -chain for $T_b = 1,000$ iterations and subsample such that \bar{T}_b is computed with $\kappa_b = 0.2$ and $\lambda_b = 5$.
 467 We then use this to run the θ -chain for $T_\theta = 10,000$ iterations. The acceptance ratio – which we
 468 denote r_α – is easy enough to compute as simply the fraction of proposed moves which we accept.
 469 However, to quantify burn-in time we must use a different metric. We can plot the mean of the
 470 objective function averaged over all samples as a proxy for burn-in time:

$$\bar{U} := \frac{1}{T_\theta} \sum_{t \in [T_\theta]} U(\theta^{(t)}) \quad (39)$$

471 As the chain will equilibrate in the vicinity of a minima in $U(\theta)$, the average over all samples is a
 472 rough indicator of the speed of burn-in. This assumes that the random starting point has high $U(\theta)$.
 473 This assumption holds as the domain of θ is a high dimensional space so we are unlikely to get a low
 474 $U(\theta)$ simply from a lucky guess. The higher the value of \bar{U} , the longer we infer the chain took to
 475 burn in and reach equilibrium. We plot the acceptance ratio r_α and average objective \bar{U} for 10 values
 476 of the step-size multiplier s logarithmically spaced in the range $(10^{-2}, 10^1)$ on figure 5.

477 We see immediately that acceptance ratio r_α steadily decreases with s . This is expected as it is
 478 unlikely we stay in a high density region of our target distribution if we are making very large
 479 steps each time. A low acceptance ratio leads to wasted samples and this inefficiency is obviously
 480 undesirable. Indeed, figure 5 suggests that a low step-size s yields high acceptance probability thus
 481 justifying the approximation used in SGLD.

482 Nevertheless, \bar{U} decreases with s indicating shorter burn-in times for larger step-sizes. However, this
 483 effect plateaus for $s > 0.2$ for this particular dataset. Therefore, we choose $s = 0.2$ as our step-size
 484 multiplier for the primary school datasets as this yields the best trade-off between r_α and \bar{U} . Indeed,
 485 the acceptance ratio is still high for $r_\alpha \approx 0.8$ for $s = 0.2$. The step-sizes for the other datasets were
 486 chosen following the same argument but we do not repeat the process here.

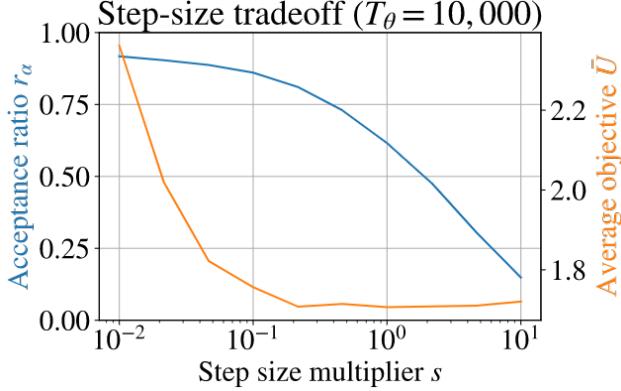


Figure 5: Primary school dataset [16] illustration of step-size tradeoff

487 A.3 Burn-in and thinning

488 As with any MCMC method, we must deal with the issues presented by burn-in and thinning. We
489 have introduced the notation \mathcal{T}_b and \mathcal{T}_θ to denote the set of samples we keep from the b and θ chains
490 respectively. Note that we generate \mathcal{T}_b and \mathcal{T}_θ samples total. The burn-in period refers to the time
491 taken for the Markov Chain to converge to the stationary distribution. Sample thinning is necessary
492 to ensure that neighbouring samples satisfy independence. However, as we do not leverage the
493 independence property this is less important in our analysis. We can write the general set \mathcal{T}_* as:

$$\mathcal{T}_* = \{T_* \kappa_* + i \lambda_* : 0 \leq i \leq \lfloor T_*(1 - \kappa_*)/\lambda_* \rfloor\} \quad (40)$$

494 Where the parameter $\kappa_* \in (0, 1)$ controls our burn-in and λ_* controls our thinning. κ_* can be
495 determined by plotting the log-target (either $S(b^{(t)})$ or $U(\theta^{(t)})$) with respect to the epoch t . κ_* is then
496 chosen to encompass the region where the log-target has roughly equilibrated. As we do not leverage
497 sample independence λ_* can be chosen less rigorously. We often just use $\lambda_b = 5$ and $\lambda_\theta = 10$.

498 By way of illustration, we can consider the primary school dataset [16]. We plot the normalised
499 objective function $U(\theta^{(t)})/N$ with respect to MALA iteration on figure 6a. We see that the chain
500 converges to the modal neighbourhood quickly (within about 2000 iterations of a total 10,000).
501 Nevertheless, we pick $\kappa_\theta = 0.4 > 0.2$ just to be on the safe-side. $\lambda_\theta = 10$ is chosen somewhat
502 arbitrarily as we do not require neighbouring samples to be independent. Nevertheless, this thinning
503 does speed up computation of quantities that require an average over all the retained samples – as
 $|\mathcal{T}_\theta| < T_\theta$.

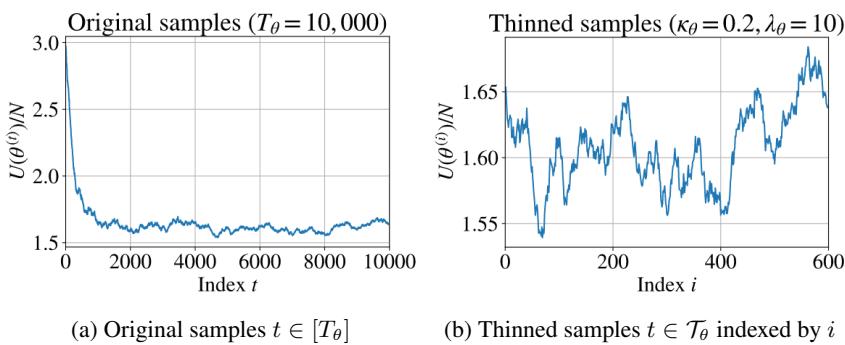


Figure 6: Primary school dataset normalised objective function against sample index

504

505 We see on figure 6b that $U(\theta^{(i)})$ does bounce around a small amount. This is to be expected we are
506 sampling from the posterior rather than converging to the MAP. There does appear to be variation
507 over a long time-scale but we can live with this as we do not require sample independence.

508 **A.4 Initializing the b-chain**

509 For the purposes of our model (the FFBM), the number of blocks B is a constant which must be
 510 specified by the data scientist. We could however, allow our choice of B to be influenced by the
 511 observed data. This places us in the domain of empirical Bayes, which must be negotiated carefully.
 512 Prior beliefs must be determined a priori else they are not prior. However, as the number of blocks
 513 only specifies the coarseness of the analysis, it is fine to allow it to vary. Indeed, Peixoto [10] shows
 514 that for a fixed average degree the maximum number of detectable blocks scales as $O(\sqrt{N})$ where N
 515 is the number of vertices.

516 If we allow B to vary in the b -chain (i.e. new blocks can be created and we permit empty blocks) then
 517 it can be run until a minimum description length (MDL) solution is reached. We take the number
 518 of non-empty blocks at the MDL solution to be our fixed block number B for subsequent analysis.
 519 Indeed, it is prudent to start our b -chain at this MDL solution as then we can burn-in time is greatly
 520 reduced.

521 **A.5 Dimensionality discussion**

522 There is a challenge when dealing with vertex-features. Many vertex feature often take only one
 523 of several discrete values within a set. For example, with the primary school dataset [16], a pupil's
 524 school-class can take one of 10 values from {"1A", "1B", "2A", "2B" ... "5A", "5B"}.

525 It is not immediately obvious how to encode this for the feature-to-block classifier. Although this is
 526 technically only a single dimension, we choose to expand the data into a set of 10 binary feature flags
 527 $\in \mathcal{X} = \{0, 1\}$. Although only one flag will be set at a time it is the simplest method for representing
 528 discrete-valued data. The reason we choose $\mathcal{X} = \{0, 1\}$ and not $\mathcal{X} = \{-1, 1\}$ is that the former
 529 allows for simpler analysis of the resulting weight values; when a feature d is switched off then
 530 $x_d = 0$ given $\mathcal{X} = \{0, 1\}$, and the feature has no impact on the softmax layer.

531 It is more intuitive to only model positive relationships. In other words, we prefer to say that this
 532 block is comprised of the vertices with feature d turned on rather than this block contains the vertices
 533 with feature d switched off. To make this explicit, say we have a feature encoding a pupil's gender:
 534 {"male", "female"}, we represent this as 2 binary feature flags "male": {0, 1} and "female": {0, 1}.
 535 This approach may seem inefficient but it is vital to ensure we can interpret the resulting parameter
 536 distributions.

537 Nevertheless, this dimensionality expansion means that we cannot include a bias term in our classifier
 538 as then the MAP solution is less peaked and our θ -samples will have higher variance. To illustrate
 539 this point we need only look at the form of the soft-max classifier with bias vector β , operating on a
 540 vector x which only has one non-zero entry $x_d = 1$. Component j of the output is given by:

$$\phi_j(x) = \frac{\exp(w_j^T x + \beta_j)}{\sum_{k \in [B]} \exp(w_k^T x + \beta_k)} = \frac{\exp(w_{jd} + \beta_j)}{\sum_{k \in [B]} \exp(w_{kd} + \beta_k)} \quad (41)$$

541 The term $w_{kd} + \beta_k$ is effectively the new weight for component k as this term always appears together.
 542 The bias term β_k is an unnecessary extra degree of freedom. To illustrate this point, we set $\beta_k = \beta_0 \forall j$.
 543 If this is the case then equation 41 can be simplified to:

$$\phi_j(x) = \frac{\exp(\beta_0) \cdot \exp(w_{jd})}{\sum_{k \in [B]} \exp(\beta_0) \cdot \exp(w_{kd})} = \frac{\exp(w_{jd})}{\sum_{k \in [B]} \exp(w_{kd})} \quad (42)$$

544 This expression is independent of β_0 and so it is free to vary. This is not the whole picture as we also
 545 have the prior which introduces a regularisation term which favours weights closer to 0. Nevertheless,
 546 for mutually exclusive binary feature flags, the bias term does not add to the expressiveness of the
 547 model and only serves to complicate analysis. We therefore remove the bias term from the Softmax
 548 classifier. Even for data comprised of several such features (e.g. school class and gender) we prefer
 549 to discard the bias term. The bias term only serves to decrease the loss function by leveraging
 550 information about the size of each detected block and not feature information. We do not wish to be

551 overly confident in our block predictions when such predictions are influenced by block size and not
552 feature information. Indeed, the approach was initially developed with a bias term but we found its
553 removal yielded more reliable and reproducible results.

554 **B Appendix: Derivations**

555 **B.1 Derivation of conditional block distribution given feature matrix**

556 We wish to determine the form of $p(b|X)$. This can be done by integrating over the joint probability
 557 with respect to θ .

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

558 We note that $b_i \in [B]$ and so the integral's value is unchanged with respect to b_i . In other words, the
 559 integrand has the same form no matter which value b_i takes as the prior is the same for each w_j . As
 560 such the integral can only be a function of at most \tilde{x}_i and σ_θ^2 as it is symmetric with respect to b_i and
 561 all the various w_j are integrated out as they are dummy variables. Therefore, denoting the integral by
 562 the (unknown) function $f(\tilde{x}_i, \sigma_\theta^2)$, we write $p(b|X)$ as follows:

$$p(b|X) = \prod_{i=1}^N f(\tilde{x}_i, \sigma_\theta^2) = \text{const w.r.t } b = c$$

553 As this is a constant with respect to b we conclude that $p(b|X)$ must be a uniform distribution. $1/c$
 554 is simply the size of the set of values that b can take. We know $b_i \in [B]$. Therefore, $b \in [B]^N$ and
 555 $|[B]^N| = B^N = 1/c$. Putting this all together we conclude that:

$$p(b|X) = B^{-N} \quad (43)$$

566 This is an extremely important result as it allows samples to be drawn from $p(b|A, X) \propto p(A|b, X) \cdot$
 567 $p(b|X) = p(A|b) \cdot p(b|X)$ without a complicated Monte-Carlo integration over θ to determine $p(b|X)$.
 568 Indeed, we do not even need to know the exact value of X to compute $p(b|X)$.

569 **B.2 Derivation of U form**

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

$$\pi_\theta(\theta) \propto p(\theta|X, b) \propto p(b|X, \theta)p(\theta) \propto \exp(-U(\theta)) \quad (44)$$

$$\therefore U(\theta) = -(\log p(b|X, \theta) + \log p(\theta)) + \text{const} \quad (45)$$

572 Where we have introduced $U(\theta)$ equal to the negative log posterior. Each of the constituent terms of
 573 $U(\theta)$ are easily computed (equation 46) by defining $y_{ij} := \mathbb{1}\{b_i = j\}$ and $a_{ij} := \phi_j(x_i; \theta)$.

$$\log p(b|X, \theta) = \sum_{i \in [N]} \sum_{j \in [B]} y_{ij} \log a_{ij} \quad \text{and} \quad \log p(\theta) = -\frac{(D+1)(B)}{2} \log 2\pi - \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (46)$$

574 Discarding constant terms, we write $U(\theta)$ as in equation 47. Note that $\|\theta\|^2 = \sum_i \theta_i^2 = \sum_{j=1}^B \|w_j\|^2$
 575 is the Euclidean norm of the vector of parameters θ .

$$U(\theta) = \left(\sum_{i=1}^N \sum_{j=1}^B y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 = N \cdot \mathcal{L}(\theta) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (47)$$

576 **B.3 Derivation of U gradient with respect to feature parameters**

577 The goal is to determine $\nabla U(\theta)$, the gradient of the negative log posterior with respect to the
 578 parameters. We repeat the form of $U(\theta)$ in equation 48.

$$U(\theta) = \left(\sum_{i \in [N]} \sum_{j \in [B]} y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (48)$$

579 Where y_{ij} is independent of θ and a_{ij} is the output from the softmax layer, with form as given in
 580 equation 49.

$$a_{ij} := \phi_j(x_i; \theta) = \frac{\exp(w_j^T \tilde{x}_i)}{\sum_{r \in [B]} \exp(w_r^T \tilde{x}_i)} \quad (49)$$

581 We note that $\theta = \{w_k\}_{k=1}^B$, and as such we can write this in vector form $\theta = [w_1^T, w_2^T \dots w_B^T]^T$.
 582 Therefore, $\nabla U(\theta) = [\partial U / \partial w_1^T, \partial U / \partial w_2^T \dots \partial U / \partial w_B^T]^T$; to compute $\nabla U(\theta)$ it suffices to find the
 583 form of $\partial U / \partial w_k$ with respect to a general k . To this end, we must first find partial derivatives of a_{ij}
 584 and $\|\theta\|$ with respect to w_k . Starting with a_{ij} :

$$\begin{aligned} \frac{\partial a_{ij}}{\partial w_k} &= \frac{\tilde{x}_i \exp(w_j^T \tilde{x}_i) \delta_{jk} \cdot \sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) - \exp(w_j^T \tilde{x}_i) \cdot \tilde{x}_i \exp(w_k^T \tilde{x}_i)}{\left(\sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) \right)^2} \\ &= \tilde{x}_i (a_{ij} \delta_{jk} - a_{ij} a_{ik}) \end{aligned} \quad (50)$$

585 Where $\delta_{jk} := \mathbb{1}\{j = k\}$. Now moving onto the derivative of $\|\theta\|^2$:

$$\frac{\partial}{\partial w_k} \|\theta\|^2 = \frac{\partial}{\partial w_k} \left(\sum_{r \in [B]} \|w_r\|^2 \right) = 2w_k \quad (51)$$

586 We are ready to put this all together, to find the partial derivative of $U(\theta)$ with respect to each w_k :

$$\begin{aligned} \frac{\partial U}{\partial w_k} &= \sum_{i=1}^N \sum_{j=1}^B y_{ij} \left(\frac{-\tilde{x}_i}{a_{ij}} (a_{ij} \delta_{jk} - a_{ij} a_{ik}) \right) + \frac{w_k}{\sigma_\theta^2} \\ &= - \left(\sum_{i=1}^N \tilde{x}_i \left(y_{ik} - a_{ik} \sum_{j=1}^B y_{ij} \right) - \frac{w_k}{\sigma_\theta^2} \right) \\ &= - \left(\sum_{i=1}^N \left\{ \tilde{x}_i (y_{ik} - a_{ik}) \right\} - \frac{w_k}{\sigma_\theta^2} \right) \end{aligned} \quad (52)$$

587 This is the required result. This form can be computed efficiently through matrix operations. The only
 588 property of y_{ij} we have used in the derivation is the sum-to-one constraint $\sum_{j=1}^B y_{ij} = 1$ for all i .

589 **B.4 Hypothesis test on feature weights**

590 We are given samples $\{\theta^{(t)}\}_{t \in \mathcal{T}_\theta} \sim p(\theta | A, X)$ and wish to determine the statistical significance of
 591 the weights. We adopt matrix notation for simplicity by representing θ with the matrix $B \times D$ matrix
 592 of feature weights W . The question is to determine the significance of a particular feature d by
 593 examining the value of W_{id} for all the values $i \in [B]$. We know that the posterior is proportional to
 594 the prior multiplied by the likelihood (assuming that the feature matrix X is already given):

$$p(\theta | A, X) \propto p(\theta) \cdot p(A | \theta, X) \quad (53)$$

595 The prior term can be evaluated but the likelihood is intractable. The closest we can get is through a
 596 Monte-Carlo integration over b :

$$\begin{aligned}
 p(A|\theta, X) &= \sum_{b \in [B]^N} p(A, b|\theta, X) \\
 &= \sum_{b \in [B]^N} p(A|b, \theta, X) \cdot p(b|\theta, X) \\
 &= \sum_{b \in [B]^N} p(A|b) \cdot p(b|\theta, X) \\
 &\approx \sum_i p\left(A|b^{(i)}\right) \quad \text{with } b^{(i)} \sim p(b|\theta, X)
 \end{aligned} \tag{54}$$

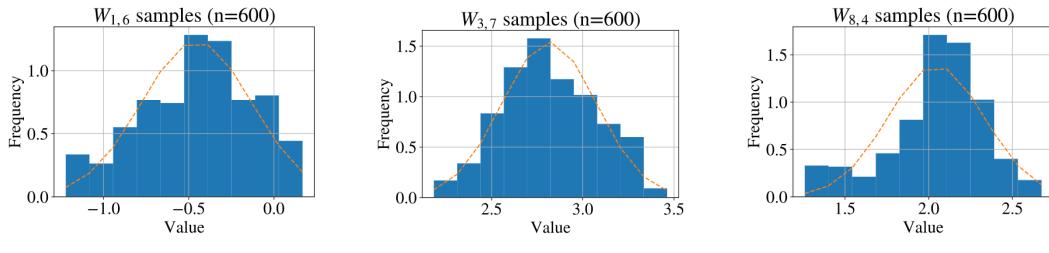
597 This could be implemented for a single value of θ but such a form cannot be used to characterise the
 598 overall form of the posterior. Nevertheless, the form in equation 54 still does highlight something
 599 interesting. The likelihood is peaked around areas of θ that generate a partition b that is highly likely
 600 in the SBM sense – high $p(A|b)$. This provides the motivation for using the Laplace approximation
 601 for modelling the posterior $p(\theta|A, X) \approx p(\theta; \mu, \Sigma)$. Indeed, the Laplace approximation is often
 602 used for modelling the posterior in logistic classification [14]. This approximation is not exact but
 603 it motivates the derivation of the dimensionality reduction technique we construct by analogy with
 604 hypothesis testing. Marginalising over the other elements, the posterior for each element of the weight
 605 matrix W can be approximated by:

$$p(W_{ij}|A, X) \approx \mathcal{N}(W_{ij}|\hat{\mu}_{ij}, \hat{\sigma}_{ij}^2) \tag{55}$$

606 Where we have used the set of samples for W drawn according to the exact posterior, to calculate
 607 unbiased estimates for the mean and standard deviation:

$$\hat{\mu}_{ij} := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} W_{ij}^{(t)} \quad \text{and} \quad \hat{\sigma}_{ij}^2 := \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} \left(W_{ij}^{(t)} - \hat{\mu}_{ij} \right)^2 \tag{56}$$

608 This approximation is not exact but we can show it is accurate empirically. Indeed, if we run the
 609 primary school experiment with hyper-parameters given in appendix C.2, we can then plot histograms
 610 of the collected W -samples and compare these to the Laplace approximation. The results are given
 611 on figure 7. Even though the Laplace approximation is not exact, it is remarkably reliable.



(a) Weight for block 1, class 3B (b) Weight for block 3, class 4A (c) Weight for block 8, class 2B

Figure 7: Histograms of sampled weights for the primary school experiment [16]. Dotted line is the applied Laplace approximation.

612 Using the approximation, we wish to construct a test to determine whether a particular weight value
 613 has $|W_{ij}| > c$ with high probability, where $c > 0$. More specifically, we wish to determine:

$$\begin{aligned}
 H_0 : |W_{ij}| &\leq c \\
 H_1 : |W_{ij}| &> c
 \end{aligned} \tag{57}$$

614 We can consider the probabilities $p(W_{ij} > c)$ and $p(W_{ij} < c)$ separately. Starting with $p(W_{ij} > c)$
 615 we can write:

$$\begin{aligned} p(W_{ij} > c) &= p\left(Z > \frac{c - \hat{\mu}_{ij}}{\hat{\sigma}_{ij}}\right) \\ &= p\left(-Z < \frac{\hat{\mu}_{ij} - c}{\hat{\sigma}_{ij}}\right) \\ &= \Phi\left(\frac{\hat{\mu}_{ij} - c}{\hat{\sigma}_{ij}}\right) \end{aligned}$$

616 Where $Z \sim \mathcal{N}(0, 1)$ was introduced as a temporary variable and $\Phi(\cdot)$ is the standard Gaussian cdf.
 617 Now we introduce the variable k to control the degree of significance of the result. By enforcing k to
 618 be less than the argument of Φ we show that:

$$\begin{aligned} \frac{\hat{\mu}_{ij} - c}{\hat{\sigma}_{ij}} &> k \\ \hat{\mu}_{ij} - k\hat{\sigma}_{ij} &> c \end{aligned}$$

619 As $\Phi(\cdot)$ is monotonically increasing, this result gives us that:

$$p(W_{ij} > c) > \Phi(k) \iff \hat{\mu}_{ij} - k\hat{\sigma}_{ij} > c \quad (58)$$

620 A similar argument can be made to show that:

$$p(W_{ij} < -c) > \Phi(k) \iff \hat{\mu}_{ij} + k\hat{\sigma}_{ij} < -c \quad (59)$$

621 Clearly equations 58 and 59 cannot both hold simultaneously as $c, \hat{\sigma}_{ij}, k > 0$. Nevertheless, a
 622 necessary and sufficient condition for one of them to hold is:

$$(\hat{\mu}_{ij} - k\hat{\sigma}_{ij}, \hat{\mu}_{ij} + k\hat{\sigma}_{ij}) \cap (-c, +c) = \emptyset \quad (60)$$

623 We therefore choose to reject the null hypothesis H_0 if and only if equation 60 holds. This tells
 624 us with confidence greater than $\Phi(k)$ that either $W_{ij} < -c$ or $W_{ij} > c$ thus rejecting the null
 625 $H_0 : |W_{ij}| \leq c$. Note that we choose to accept H_0 for the case where the combined probabilities
 626 of $p(W_{ij} < -c) + p(W_{ij} > c) > \Phi(k)$, if equation 60 is not satisfied. This is because we wish
 627 to apply this for dimensionality reduction purposes. We only want to keep weights that have a
 628 confident prediction outside the null-space not those that succeed by summing the two extremes of
 629 their distribution. Note that this case is very rare empirically.

630 **C Appendix: Implementation details**

631 **C.1 Algorithms**

Algorithm 1 Block membership sample generation

```

procedure SAMPLEBLOCKMEMBERSHIPS( $A, T_b$ )
     $b^{(0)} \leftarrow \arg \min_b S(b|A)$                                  $\triangleright$  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  $\{b^{(t)}\}_{t=1}^{T_b}$ 
end procedure

```

Algorithm 2 FFBM parameter pseudo-marginal inference

```

procedure SAMPLEFEATUREWEIGHTS( $X, \{b^{(t)}\}, T_b, \sigma_\theta, s$ )
     $\hat{Y}_{ij} \leftarrow \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} \mathbb{1}\{b_i^{(t)} = j\} \quad \forall i, j$ 
     $\theta^{(0)} \leftarrow \sim \mathcal{N}(0, \sigma_\theta I)$ 

    for  $t \in \{0, 1 \dots T_\theta - 1\}$  do
         $\xi \leftarrow \sim \mathcal{N}(0, I)$ 
         $h_t \leftarrow \frac{s}{N} \cdot 250(1000 + t)^{-0.8}$ 
         $g_t \leftarrow \nabla U(\theta^{(t)}|X, \hat{Y})$ 

         $\theta' \leftarrow \theta^{(t)} - h_t \cdot g_t + \sqrt{2h_t} \cdot \xi$ 
         $\log \alpha_\theta \leftarrow \log \alpha_\theta(\theta^{(t)}, \theta'|A, \hat{Y})$ 
         $\eta \leftarrow \sim \text{Unif}(0, 1)$ 
        if  $\log \eta < \log \alpha_\theta$  then
             $\theta^{(t+1)} \leftarrow \theta'$ 
        else
             $\theta^{(t+1)} \leftarrow \theta^{(t)}$ 
        end if
    end for
    return  $\{\theta^{(t)}\}_{t=1}^{T_\theta}$ 
end procedure

```

Algorithm 3 Dimensionality reduction

```

procedure REDUCEDIMENSION( $\{W^{(t)}\}, \mathcal{T}_\theta, k, D'$ )
     $(B, D) \leftarrow W^{(0)}.shape$ 
     $\hat{\mu}_{ij} \leftarrow \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} W_{ij}^{(t)} \quad \forall i \in [B], j \in [D]$ 
     $\hat{\sigma}_{ij} \leftarrow \frac{1}{|\mathcal{T}_\theta|} \sum_{t \in \mathcal{T}_\theta} \left( W_{ij}^{(t)} - \hat{\mu}_{ij} \right)^2 \quad \forall i \in [B], j \in [D]$ 

    for  $d \in [D]$  do
        for  $i \in [B]$  do
             $l_i \leftarrow \hat{\mu}_{id} - k \cdot \hat{\sigma}_{id}$ 
             $u_i \leftarrow \hat{\mu}_{id} + k \cdot \hat{\sigma}_{id}$ 
            if  $l_i \leq 0$  and  $u_i \geq 0$  then
                 $l_i, u_i \leftarrow 0$ 
            end if
        end for
         $c_d \leftarrow \max_i \min(|l_i|, |u_i|)$ 
    end for

    indexArray  $\leftarrow$  indexSort( $c$ , descending=True)[ $0 : D'$ ]
     $d^* \leftarrow$  indexArray[-1]
     $\mathcal{D}' \leftarrow$  Set(indexArray)
     $c^* \leftarrow c_{d^*}$ 
    return  $\mathcal{D}', c^*$ 
end procedure

```

632 **C.2 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

633 **C.3 Hardware specification**

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

- 637 • **CPU:** Intel(R) Core(TM) i7-1065G7
638 • **RAM:** 8GB
639 • **GPU:** Intel(R) Iris(R) Plus Graphics

640 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	~10s	—	~11s
School	~1s	~7s	~7s	~15s
FB Egonet	~2s	~50s	~8s	~60s