
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) [10]. This is a latent variable model where each vertex belongs to a single block and
34 the 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) [20]. 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 [9] 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 [15]. 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:

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

83 Where $\tilde{x} := [x_1, x_2, \dots, x_D, 1]^T$ is an augmented version of x that allows for a bias term. The
 84 parameter vector θ for this stage contains all the weight vectors $\theta = \{w_k\}_{k=1}^B$. Each w_k has
 85 dimension $D + 1$. We could instead write the parameters θ as a $B \times (D + 1)$ matrix of weights W ;
 86 this form has computational benefits as then $z_i := W\tilde{x}_i$, which is the input to the softmax activation
 87 function.

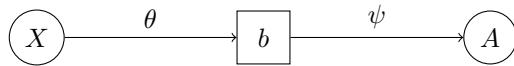


Figure 1: The feature-first block model (FFBM)

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

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

90 3.1 Prior selection

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

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

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

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

98 This is an enormously important simplification as evaluating $p(b|X)$ does not require an expensive
 99 Monte-Carlo integration over the θ -domain nor does it require the exact value of X . Peixoto
 100 [15] proposes careful choices for the additional microcanonical SBM parameters ψ which we
 101 adopt. Peixoto's idea is to write the joint prior on (b, e, k) as a product of conditionals $p(b, e, k) =$
 102 $p(b)p(e|b)p(k|e, b) = p(b)p(\psi|b)$. For our purposes we must insert a conditioning on X , to form our
 103 pseudo-prior for b and ψ , to give equation 7.

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

104 Where we leverage the fact $(\psi \perp\!\!\!\perp X)|b$. We then borrow the priors proposed by Peixoto [15] for
 105 $p(\psi|b)$ to complete our model. Please refer to appendix A.1 for the exact form of $p(\psi|b)$. All that
 106 concerns the main argument is we have a computable form.

107 **4 Inference**

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

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

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

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

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

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

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

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

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

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

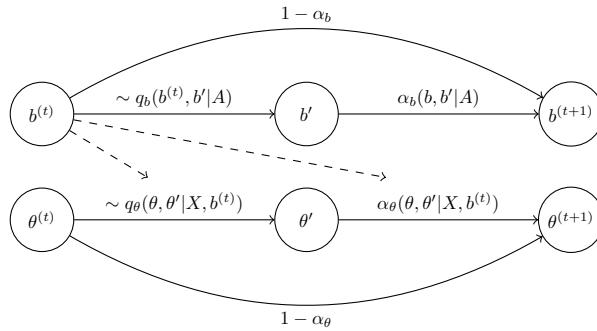


Figure 2: Sampling sequence

125
 126 The reason we split the Markov chain into two stages is because the summation over all latent states
 127 $b \in [B]^N$ required to directly compute the likelihood $p(A|X, \theta) = \sum_{b \in [B]^N} p(A|b)P(b|X, \theta)$ is
 128 intractable – $O(B^N)$. Figure 2 shows an overview of the proposed method. We have introduced
 129 subscripts and conditionings to make explicit what variables each step utilises. We note the power of
 130 the simplification given by equation 6. As $p(b|X)$ does not depend on the exact value of X , we do
 131 not need to know the value of X to perform the sampling on b . Conversely, for the $\theta^{(t)}$ samples, we
 132 use only $b^{(t)}$ but not A as $(\theta \perp\!\!\!\perp A)|b$.

133 **4.1 Sampling block memberships**

134 Peixoto [13] proposes a Monte Carlo method which we will base our approach on. It relies on writing
 135 the posterior in the following form:

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

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

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

139 Since we are using the microcanonical SBM formulation, there is only one value of ψ that is
 140 compatible with the given (A, b) pair (given in equation 2). We denote this value $\psi^* = \{\psi_k^*, \psi_e^*\}$.

141 Therefore, the summation over all ψ reduces to just the single ψ^* term; this is the power of the
 142 microcanonical formulation. We also define the microcanonical entropy of the configuration as.

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

143 This entropy can equally be thought of as the description length of the graph. The exact form of
 144 the proposal q_b is explored thoroughly by Peixoto [13] and not repeated here. There is a widely
 145 used library for Python made available under LGPL called `graph-tool` [14], which implements this
 146 algorithm. The only modification we make is in the block membership prior $p(b)$ which we replace
 147 with $p(b|X) = B^{-N}$, which cancels out in the MH accept-reject step as it is independent of b .

148 **4.2 Sampling feature-to-block generator parameters**

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

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

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

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

154 $U(\theta)$ in equation 17 appears a typical objective function for neural network training. The first term
 155 is introduced by the likelihood. We collect it into $N \cdot \mathcal{L}(\theta)$, which is the cross-entropy between the
 156 graph-predicted and feature-predicted block memberships summed over all vertices. The second
 157 term of equation 17 – introduced by the prior – brings a form of regularisation, guarding against
 158 over-fitting. Different to traditional applications, our goal is not to find the minimiser of $U(\theta)$ but to
 159 draw samples from the posterior $\pi_\theta(\cdot) \propto \exp(-U(\cdot))$. We can use ∇U as a useful heuristic to bias
 160 our proposal towards regions of higher target density. We therefore adopt the Metropolis-adjusted
 161 Langevin algorithm (MALA) – first proposed by Roberts and Tweedie [16]. Given the current sample
 162 θ , we generate a new sample θ' according to equation 18.

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

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

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

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

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

170 **4.3 Sampling sequence**

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

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

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

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

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

187 **4.4 Dimensionality reduction**

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

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

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

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

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

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

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

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

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

205

206 **5 Experiments**

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

- 209 • **Political books** [11] ($N = 105, E = 441, D = 3$) – network of Amazon book sales about U.S.
 210 politics, published close to the presidential election in 2004. Two books are connected if they were
 211 frequently co-purchased by customers. Vertex features encode the political affiliation of the author
 212 (liberal, conservative or neutral).
- 213 • **Primary school dynamic contacts** [18] ($N = 238, E = 5539, D = 13$) – network of face-to-face
 214 contacts amongst students and teachers at a primary school in Lyon, France. Two nodes are
 215 connected if the two parties shared a face-to-face interaction over the school-day. Vertex features
 216 include class membership (one of 10 values: 1A-5B), gender and teacher/student distinction. No
 217 further identifiable information is retained. We choose to analyse just the second day of results.
- 218 • **Facebook egonet** [7] ($N = 747, E = 30025, D = 480$) – an assortment of Facebook users'
 219 friends lists. Vertex features are extracted from each user's profile and are fully anonymised. They
 220 include information about education history, languages spoken, gender, home-town, birthday etc.
 221 We focus on the egonet with id 1912.

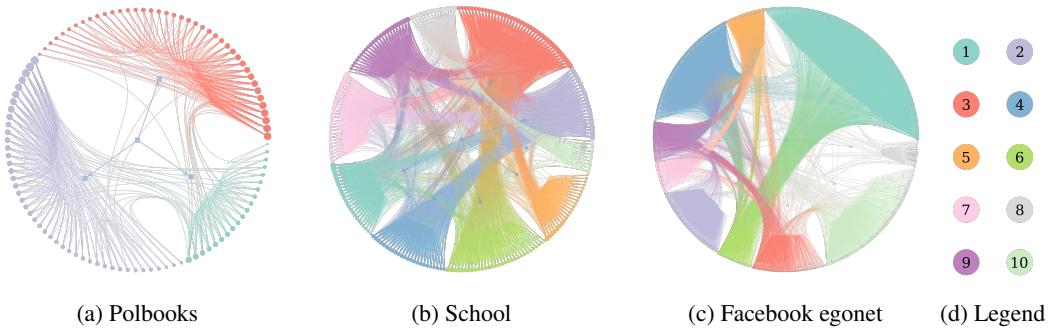


Figure 3: Networks laid out and coloured according by inferred block memberships \hat{y} for a given experiment iteration. Visualisation performed using *graph-tool* [14]

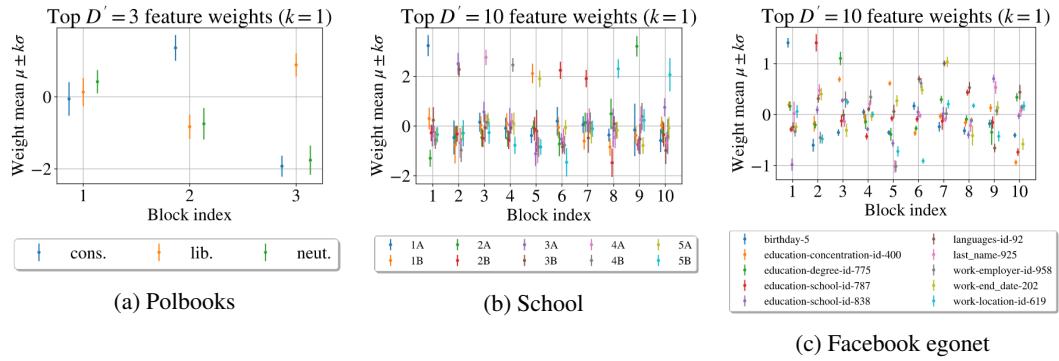


Figure 4: Reduced dimension feature-to-block generator weight samples

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

Dataset	B	D	D'	S_e	$\bar{\mathcal{L}}_0$	$\bar{\mathcal{L}}_1$	c^*	$\bar{\mathcal{L}}'_0$	$\bar{\mathcal{L}}'_1$
Polbooks	3	3	–	2.250 ± 0.001	0.584 ± 0.033	0.557 ± 0.061	–	–	–
School	10	13	10	1.894 ± 0.006	0.793 ± 0.096	0.914 ± 0.112	1.112 ± 0.230	0.791 ± 0.096	0.864 ± 0.139
FB egonet	10	480	10	1.626 ± 0.003	1.305 ± 0.034	1.539 ± 0.087	0.942 ± 0.042	1.496 ± 0.093	1.578 ± 0.104

222 We require metrics to assess performance. This can be split into two separate components: the
 223 microcanonical SBM fit (concerned with the b -samples) and the fit of the feature-to-block generator
 224 (concerned with the θ -samples). Starting with the SBM, $S(b)$ (equation 15) can be interpreted as
 225 the description length of the partition imposed by b . It is only natural to divide this quantity by the
 226 number of entities (nodes and edges) in our graph $N + E$ to allow for rough comparison between
 227 graphs. This defines a simple metric to gauge the fit of the SBM: the description length per entity
 228 averaged over the b -samples (equation 28):

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

229 However, to assess the performance of the feature-to-block predictor, we must partition the vertex
 230 set $[N]$ into training and test sets. We choose to randomly partition the vertices on each experiment
 231 run such that a constant fraction f of the available vertices go to form our training set \mathcal{G}_0 and the
 232 remainder are held out to form our test set \mathcal{G}_1 . The b -chain is run using the whole network but we only
 233 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
 234 log target U (equation 17) for comparison as the total cross-entropy loss is scaled by the size of each
 235 set but the prior term stays constant. We therefore must use the average cross-entropy loss over each
 236 set (equation 29):

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

237 Where $\star \in \{0, 1\}$ has been introduced to toggle between training and test sets. Table 1 summarises
 238 the results for each experiment.² We also apply the dimensionality reduction method on the two higher
 239 dimensional datasets (the school and FB egonet). For this we leverage equation 27, to reduce the
 240 dimension from D to D' with $k = 1$ to yield the maximal cutoff c^* . We then retrain the feature-block
 241 predictor using just the retained feature set \mathcal{D}' and report the loss over the training and test sets for
 242 the reduced classifier – denoted $\bar{\mathcal{L}}'_0$ and $\bar{\mathcal{L}}'_1$ respectively. These values are also included in table 1.

243 Table 1 already highlights some general trends in the results. Firstly, the variance of the test loss
 244 $\bar{\mathcal{L}}_1$ tends to be higher than the training loss $\bar{\mathcal{L}}_0$. This is expected as our test set is smaller than the
 245 training set and so more susceptible to variability in its construction. Indeed, most of the variance
 246 in the evaluation of $\bar{\mathcal{L}}_0$ and $\bar{\mathcal{L}}_1$ comes from the random partitioning of the graph into training and
 247 test sets. Secondly, it can be seen that the dimensionality reduction procedure brings the training
 248 and test losses closer together. This implies that the features we keep are indeed correlated with the
 249 underlying graphical partition and that the approach generalises correctly.

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

255 5.1 Political books

256 We wish to determine whether the author’s political affiliation is a good predictor of the overall
 257 network structure. We choose to partition the network into $B = 3$ communities as we only have this
 258 many distinct values for political affiliation (conservative, liberal or neutral). From, figure 4a, we see
 259 that all 3 blocks have a distinct political affiliation as their largest positive component. This is strong
 260 evidence that political affiliation is indeed the axis which best predicts the 3-way natural partition of
 261 the graph into blocks. Furthermore, table 1 we see that the training and test losses are very similar
 262 (the test loss mean is even below the training loss) and both are low in magnitude. This provides
 263 further evidence to the claim that political affiliation is the best explanatory variable for the overall
 264 network structure.

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

265 **5.2 Primary school dynamic contacts**

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

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

279 **5.3 Facebook egonet**

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

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

291 **6 Conclusion**

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

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

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

309 **References**

- 310 [1] Emmanuel Abbe. Graph compression: The effect of clusters. In *2016 54th Annual Allerton*
311 *Conference on Communication, Control, and Computing (Allerton)*, pages 1–8, 2016. doi:
312 10.1109/ALLERTON.2016.7852203.
- 313 [2] Edo M Airoldi, David Blei, Stephen Fienberg, and Eric Xing. Mixed membership
314 stochastic blockmodels. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou,
315 editors, *Advances in Neural Information Processing Systems*, volume 21. Curran As-
316 sociates, Inc., 2009. URL <https://proceedings.neurips.cc/paper/2008/file/8613985ec49eb8f757ae6439e879bb2a-Paper.pdf>.
- 318 [3] Christophe Andrieu and Gareth O. Roberts. The pseudo-marginal approach for efficient Monte
319 Carlo computations. *The Annals of Statistics*, 37(2):697 – 725, 2009. doi: 10.1214/07-AOS574.
320 URL <https://doi.org/10.1214/07-AOS574>.
- 321 [4] Solenne Gaucher, Olga Klopp, and Geneviève Robin. Outliers detection in networks with
322 missing links, 2020.
- 323 [5] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications.
324 *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL <http://www.jstor.org/stable/2334940>.
- 326 [6] Matthew Kraatz, Nina Shah, and Emmanuel Lazega. The collegial phenomenon: The social
327 mechanisms of cooperation among peers in a corporate law partnership. *Administrative Science
328 Quarterly*, 48:525, 09 2003. doi: 10.2307/3556688.
- 329 [7] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego net-
330 works. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, ed-
331 itors, *Advances in Neural Information Processing Systems*, volume 25. Curran As-
332 sociates, Inc., 2012. URL <https://proceedings.neurips.cc/paper/2012/file/7a614fd06c325499f1680b9896beedeb-Paper.pdf>.
- 334 [8] Benjamin F. Maier and Dirk Brockmann. Cover time for random walks on arbitrary complex
335 networks. *Phys. Rev. E*, 96:042307, Oct 2017. doi: 10.1103/PhysRevE.96.042307. URL
336 <https://link.aps.org/doi/10.1103/PhysRevE.96.042307>.
- 337 [9] Nikhil Mehta, Lawrence Carin Duke, and Piyush Rai. Stochastic blockmodels meet graph
338 neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the
339 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine
340 Learning Research*, pages 4466–4474. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/mehta19a.html>.
- 342 [10] Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic block-
343 structures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi:
344 10.1198/016214501753208735. URL <https://doi.org/10.1198/016214501753208735>.
- 345 [11] Boris Pasternak and Ivor Ivask. Four unpublished letters. *Books Abroad*, 44(2):196–200, 1970.
346 ISSN 00067431. URL <http://www.jstor.org/stable/40124305>.
- 347 [12] Tiago P. Peixoto. Parsimonious module inference in large networks. *Physical Review Letters*,
110(14), Apr 2013. ISSN 1079-7114. doi: 10.1103/physrevlett.110.148701. URL <http://dx.doi.org/10.1103/PhysRevLett.110.148701>.
- 350 [13] Tiago P. Peixoto. Efficient monte carlo and greedy heuristic for the inference of stochastic
351 block models. *Physical Review E*, 89(1), Jan 2014. ISSN 1550-2376. doi: 10.1103/physreve.89.
352 012804. URL <http://dx.doi.org/10.1103/PhysRevE.89.012804>.
- 353 [14] Tiago P. Peixoto. The graph-tool python library. *figshare*, 2014. doi: 10.6084/m9.figshare.
354 1164194. URL http://figshare.com/articles/graph_tool/1164194.

- 355 [15] Tiago P. Peixoto. Nonparametric bayesian inference of the microcanonical stochastic block
356 model. *Physical Review E*, 95(1), Jan 2017. ISSN 2470-0053. doi: 10.1103/physreve.95.012317.
357 URL <http://dx.doi.org/10.1103/PhysRevE.95.012317>.
- 358 [16] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of Langevin distributions
359 and their discrete approximations. *Bernoulli*, 2(4):341 – 363, 1996. doi: bj/1178291835. URL
360 <https://doi.org/>.
- 361 [17] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding,
362 2019.
- 363 [18] Juliette Stehlé, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-François Pinton,
364 Marco Quaggiotto, Wouter Van den Broeck, Corinne Régis, Bruno Lina, and Philippe Vanhems.
365 High-resolution measurements of face-to-face contact patterns in a primary school. *PLOS ONE*,
366 6(8):1–13, 08 2011. doi: 10.1371/journal.pone.0023176. URL <https://doi.org/10.1371/journal.pone.0023176>.
- 368 [19] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynam-
369 ics. In *Proceedings of the 28th International Conference on International Conference on*
370 *Machine Learning*, ICML’11, page 681–688, Madison, WI, USA, 2011. Omnipress. ISBN
371 9781450306195.
- 372 [20] Jun Zhu, Jiaming Song, and Bei Chen. Max-margin nonparametric latent feature models for
373 link prediction, 2016.

374 **Checklist**

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

416 **A Appendix: Additional material**

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

418 For reference, we provide the likelihoods and priors for the mircocanonical SBM proposed by Peixoto
419 [15]. We have that the graph A is drawn from the SBM:

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

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

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

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

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

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

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

430 Obviously, this form is only defined if A respects the constraints imposed by (k, e, b) else the
431 likelihood is 0. With the likelihood defined, we move on to the prior. As discussed in the main text,
432 for the FFBM b is an intermediate variable and not a parameter so we are not free to choose a prior
433 for it. Nevertheless, we can borrow the conditional prior proposed by Peixoto [15] for $p(e, k|b)$:

$$p(e, k|b) = p(e|b)p(k|e, b) = \left[\begin{Bmatrix} \{ \frac{B}{2} \} \\ E \end{Bmatrix} \right]^{-1} \cdot \left[\prod_r \frac{\prod_j \eta_j^r!}{n_r! q(e_r, n_r)} \right] \quad (34)$$

434 Where $\{ \frac{n}{m} \}$ is shorthand for $\binom{n+m-1}{m} = \frac{(n+m-1)!}{(n-1)!(m)!}$ which can be thought of as the total number
435 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
436 number of edges in the graph. Importantly, E is not allowed to vary and so $p(e|b)$ is uniform with
437 respect to e . The variable η_j^r is introduced to denote the number of vertices in block r that have degree
438 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
439 histograms with at most n non-zero bins that sum to m . $q(m, n)$ is related to but different from $\{ \frac{n}{m} \}$.
440 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
441 number of vertices assigned to block r .

442 The form of these priors were chosen carefully by Peixoto [15] to more closely match the structure of
443 empirical networks than simple uniform priors. We do not repeat his arguments here.

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

445 For sampling from the θ -chain of the block membership generator parameters, we employed the
446 Metropolis Adjusted Langevin Algorithm (MALA). At iteration t , the proposed sample is generated
447 by:

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

448 There are two competing objectives when choosing the step-size h_t . On the one hand, we want the
449 step-size to be large so that we arrive at a high density region quickly. However, too large a step-size

450 will lead to a lower acceptance ratio and thus inefficient sampling. A solution to this problem would
451 be to slowly decrease the step-size with t - often called simulated annealing. Therefore, we still have
452 a short burn-in time but will not bounce around the mode for large t . As well as the trivial constraint
453 for h_t to be strictly positive, we introduce two further constraints as outlined by Welling and Teh
454 [19]:

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

455 The first constraint ensures that we have cover sufficient distance to arrive at any arbitrary point in
456 our domain, no matter the starting point. The second constraint ensures that once we converge to the
457 mode rather than simply bouncing around it. Welling and Teh [19] propose the following form for a
458 polynomially decaying step-size which we adopt:

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

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

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

462 Where N is the number of data-points we are considering and now s is the only free variable which
463 we call the step-size scaling. For approximate methods, we can choose to bypass the MH accept-reject
464 entirely to speed up computation. If this is done, the algorithm is instead called stochastic gradient
465 Langevin diffusion (SGLD) [19]. This speeds up computation at the expense of exactness of the
466 method.

467 A.3 Burn-in and thinning

468 As with any MCMC method, we must deal with the issues presented by burn-in and thinning. We
469 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
470 respectively. Note that we generate T_b and T_{θ} samples total. The burn-in period refers to the time
471 taken for the Markov Chain to converge to the stationary distribution. Sample thinning is necessary
472 to ensure that neighbouring samples satisfy independence. However, as we do not leverage the
473 independence property this is less important in our analysis. We can write the general set \mathcal{T}_* as:

$$\mathcal{T}_* = \{T_* \kappa_* + i \lambda_* : 0 \leq i \leq \lfloor T_*(1 - \kappa_*) / \lambda_* \rfloor\} \quad (39)$$

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

478 A.4 Initializing the b-chain

479 For the purposes of our model (the FFBM), the number of blocks B is a constant which must be
480 specified by the data scientist. We could however, allow our choice of B to be influenced by the
481 observed data. This places us in the domain of empirical Bayes, which must be negotiated carefully.
482 Prior beliefs must be determined a priori else they are not prior. However, as the number of blocks
483 only specifies the coarseness of the analysis, it is fine to allow it to vary. Indeed, Peixoto [12] shows
484 that for a fixed average degree the maximum number of detectable blocks scales as $O(\sqrt{N})$ where N
485 is the number of vertices.

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

490 **A.5 Dimensionality discussion**

491 There is a challenge when dealing with vertex-features. Many vertex feature often take only one
492 of several discrete values within a set. For example, with the primary school dataset [18], a pupil's
493 school-class can take one of 10 values from {"1A", "1B", "2A", "2B" ... "5A", "5B"}.

494 It is not immediately obvious how to encode this for the feature-to-block classifier. Although this is
495 technically only a single dimension, we choose to expand the data into a set of 10 binary feature flags
496 $\in \mathcal{X} = \{0, 1\}$. Although only one flag will be set at a time it is the simplest method for representing
497 discrete-valued data. The reason we choose $\mathcal{X} = \{0, 1\}$ and not $\mathcal{X} = \{-1, 1\}$ is that the former
498 allows for simpler analysis of the resulting weight values. When a feature is switched off, it has
499 no impact on the Softmax classifier. It is simpler to only model positive relationships. We prefer
500 to say that this block is comprised of the vertices with feature d on rather than this block contains
501 the vertices with feature d switched off. Nevertheless, this dimensionality expansion means that we
502 cannot include a bias term in our classifier as then there is no unique MAP solution.

503 **B Appendix: Derivations**

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

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

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

507 We note that $b_i \in [B]$ and so the integral's value is unchanged with respect to b_i . The integrand
508 has the same form no matter which value b_i takes as the prior is the same for each w_j . As such the
509 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 all the
510 various w_j are integrated out as they are dummy variables. Therefore, denoting the integral by the
511 (unknown) function $f(\tilde{x}_i, \sigma_\theta^2)$, we write $p(b|X)$ as follows:

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

512 As this is a constant with respect to b we conclude that $p(b|X)$ must be a uniform distribution. $1/c$
513 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
514 $|[B]^N| = B^N = 1/c$. Putting this all together we conclude that:

$$p(b|X) = B^{-N} \quad (40)$$

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

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

$$\pi_\theta(\theta) \propto p(\theta|X, b) \propto p(b|X, \theta)p(\theta) \propto \exp(-U(\theta)) \quad (41)$$

$$\therefore U(\theta) = -(\log p(b|X, \theta) + \log p(\theta)) + \text{const} \quad (42)$$

518 Where we have introduced $U(\theta)$ equal to the negative log posterior. Each of the constituent terms of
519 $U(\theta)$ are easily computed (equation 43) by defining $y_{ij} := \mathbb{1}\{b_i = j\}$ and $a_{ij} := \phi_j(x_i; \theta)$.

$$\log p(b|X, \theta) = \sum_{i \in [N]} \sum_{j \in [B]} y_{ij} \log a_{ij} \quad \text{and} \quad \log p(\theta) = -\frac{(D+1)(B)}{2} \log 2\pi - \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (43)$$

520 Discarding constant terms, we write $U(\theta)$ as in equation 44. Note that $\|\theta\|^2 = \sum_i \theta_i^2 = \sum_{j=1}^B \|w_j\|^2$
521 is the Euclidean norm of the vector of parameters θ .

$$U(\theta) = \left(\sum_{i=1}^N \sum_{j=1}^B y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 = N \cdot \mathcal{L}(\theta) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (44)$$

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

523 The goal is to determine $\nabla U(\theta)$, the gradient of the negative log posterior with respect to the
524 parameters. We repeat the form of $U(\theta)$ in equation 45.

$$U(\theta) = \left(\sum_{i \in [N]} \sum_{j \in [B]} y_{ij} \log \frac{1}{a_{ij}} \right) + \frac{1}{2\sigma_\theta^2} \|\theta\|^2 \quad (45)$$

525 Where y_{ij} is independent of θ and a_{ij} is the output from the softmax layer, with form as given in
 526 equation 46.

$$a_{ij} := \phi_j(x_i; \theta) = \frac{\exp(w_j^T \tilde{x}_i)}{\sum_{r \in [B]} \exp(w_r^T \tilde{x}_i)} \quad (46)$$

527 We note that $\theta = \{w_k\}_{k=1}^B$, and as such we can write this in vector form $\theta = [w_1^T, w_2^T \dots w_B^T]^T$.
 528 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
 529 form of $\partial U / \partial w_k$ with respect to a general k .

530 To this end, we must first find partial derivatives of a_{ij} and $\|\theta\|$ with respect to w_k . Starting with a_{ij} :

$$\begin{aligned} \frac{\partial a_{ij}}{\partial w_k} &= \frac{\tilde{x}_i \exp(w_j^T \tilde{x}_i) \delta_{jk} \cdot \sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) - \exp(w_j^T \tilde{x}_i) \cdot \tilde{x}_i \exp(w_k^T \tilde{x}_i)}{\left(\sum_{r \in [B]} \exp(w_r^T \tilde{x}_i) \right)^2} \\ &= \tilde{x}_i (a_{ij} \delta_{jk} - a_{ij} a_{ik}) \end{aligned} \quad (47)$$

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

$$\frac{\partial}{\partial w_k} \|\theta\|^2 = \frac{\partial}{\partial w_k} \left(\sum_{r \in [B]} \|w_r\|^2 \right) = 2w_k \quad (48)$$

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

$$\begin{aligned} \frac{\partial U}{\partial w_k} &= \sum_{i=1}^N \sum_{j=1}^B y_{ij} \left(\frac{-\tilde{x}_i}{a_{ij}} (a_{ij} \delta_{jk} - a_{ij} a_{ik}) \right) + \frac{w_k}{\sigma_\theta^2} \\ &= - \left(\sum_{i=1}^N \tilde{x}_i \left(y_{ik} - a_{ik} \sum_{j=1}^B y_{ij} \right) - \frac{w_k}{\sigma_\theta^2} \right) \\ &= - \left(\sum_{i=1}^N \left\{ \tilde{x}_i (y_{ik} - a_{ik}) \right\} - \frac{w_k}{\sigma_\theta^2} \right) \end{aligned} \quad (49)$$

533 This is the required result. This form can be computed efficiently through matrix operations. The only
 534 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 .

535 **C Appendix: Implementation details**

536 **C.1 Algorithms**

Algorithm 1 Block membership sample generation

```

 $b^{(0)} \leftarrow \arg \min_b S(b|A)$  ▷ 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  $\{\theta^{(t)}\}_{t=1}^{T_\theta}$ 

```

537 **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

538 **C.3 Hardware specification**

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

- 542 • **CPU:** Intel(R) Core(TM) i7-1065G7
- 543 • **RAM:** 8GB
- 544 • **GPU:** Intel(R) Iris(R) Plus Graphics

⁵⁴⁵ On this hardware each experiment iteration took the following amount of time to execute:

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