Stochastic block model and extensions

Christos Papachristou

With amounts of information ever increasing, modeling and predicting community structures can be of great value for better understanding users, and providing improved personalised services for them. Stochastic block models are random graph models that exhibit cluster structures. They are being used as base models for understanding community detection and clustering in networks. Here, we explore the stochastic blockmodel and some of its extensions, that make it more suitable for applications in real-world networks. Specifically, we will use techniques from Bayesian nonparametrics and MCMC methods while presenting extensions of the stochastic blockmodel in order to derive models that can be tested in characterizing the underlying true node partition of networks. We will incorporate structured information from node attributes, and provide strategies for estimation, uncertainty quantification, model selection and prediction. Also, we will present a way to test whether proposed partition structures fit a specific network via the Bayes factor of the stochastic blockmodel with known structure and the infinite relational model, which allows clusters to be unknown, random and revealed by the block–connectivity patterns in the network.

stochastic block model | bayesian nonparametrics | node partitions

Clustering and community detection are significant problems in network science, machine learning and data science. The stochastic block model is a random graph model with block structures which is being used as a base model to study both of these notions in networks. Stochastic block models group nodes that link similar nodes within communities, thereby arriving at a clustered group structure. A possible downside of this approach is perhaps being too restrictive and not having the ability to create communities with weaker interaction within themselves, rather than between. In this article, we will be presenting the stochastic block model, some of its extensions and an application in the London gang network.

Consider a binary undirected network with n nodes and let Y be its $n \times n$ symmetric adjacency matrix, with elements $y_{uv} = y_{vu} = 1$ if nodes v and u are connected and $y_{uv} = y_{vu} = 0$ otherwise.

Stochastic block model

The stochastic blockmodel (SBM) (1) can be seen as an extension of the Erdős–Rényi (ER) model (2). In the ER model, edges are drawn independently with probability p, constructing a model of one parameter. This model is rich and has been studied extensively, as it exhibits interesting phase transition phenomena (3) and it can help understand simple rules like small average distances in networks. The downside is that it is too simplistic to model real networks, specifically due to its strong homogeneity and lack of cluster structures, as well as its Poisson degree distribution. In the SBM, nodes in a network connect based on their cluster assignments, meaning that the probability of an edge linking nodes u, v depends on the clusters the nodes belong to. The goal of community detection is to recover the clusters behind an observed graph. For an extensive report on the limits of community detection in the SBM see (4).

The SBM is defined as follows (4). Let n be a positive integer (number of nodes), k a positive integer (number of communities), a probability vector $p = (p_1, \dots, p_k)$ (prior on k communities), and a symmetric matrix W of dimension $k \times k$ with entries in $[0, 1]$ (connectivity probabilities). The pair (Z, G) is drawn according to an SBM(n, p, W) if Z is an n-dimensional vector with i.i.d. components under p, and G is a simple graph of n edges, where nodes i, j are connected with probability W_{X_i, X_j}, independent of other pair of edges. Also, we denote the community sets by $\Omega_i(Z) := \{v \in [n] : X_v = i\}$, $i \in [k]$.

The distribution of (Z, G) where $G = ([n], \mathrm{E}(G))$ is defined as

$$\mathbb{P}(Z = z) := \prod_{u=1}^{n} p_{x_u} = \prod_{i=1}^{k} p_i^{|\Omega_i|}, \qquad [1]$$

$$\mathbb{P}(E(G) = y | Z = z) := \prod_{1 \le u < v \le n} W_{x_u, x_v}^{y_{uv}} (1 - W_{x_u, x_v})^{1 - y_{uv}} = \prod_{1 \le i < j \le k} W_{i,j}^{m_{i,j}} (1 - W_{i,j})^{\overline{m}_{i,j}}, \qquad [2]$$

where $m_{i,j}$ and $\overline{m}_{i,j}$ denote the number of edges and non-edges respectively between communities i and j. The random vector $z = (z_1, \dots, z_n)$ is the node membership vector associated to the node partition $\{Z_1, \dots, Z_k\}$, so that $z_u = h$ if $u \in Z_h$. The classical SBMs setup Eq. (1), Eq. (2) assumes independent B(a, b) prior for the block probabilities $W_{i,j}$. Thus, the joint density of the elements of W is $p(W) = \prod_{1 \le i < j \le k} W_{i,j}^{a-1} (1 - W_{i,j})^{b-1} \mathrm{B}(a, b)^{-1}$, for B$(.,.)$ the Beta function.

The goal of SBMs is to infer the node partition. Due to the beta-binomial conjugacy, we can marginalize over W and obtain

$$p(Y | z) = \prod_{1 \le i < j \le k} \frac{\mathrm{B}(a + m_{i,j}, b + \overline{m}_{i,j})}{\mathrm{B}(a, b)}. \qquad [3]$$

Significance Statement

Uncovering the community structure of node partitions is of great value for everyone from tech-companies (recommendation systems) to governments (uncovering structure of criminal organisations) to hospitals (describing the stage and structure of a disease in the human brain network). To address such a problem, a wide range of flexible cluster-based models that provide inference and prediction techniques are being explored in Bayesian nonparametrics. However, the transition of these models into networks has yet to be fully developed. We introduce the stochastic blockmodel as a base model for describing cluster structures in networks, some extensions for describing real-world networks and a Bayesian framework for cluster inference and testing the fit of node partitions to networks.

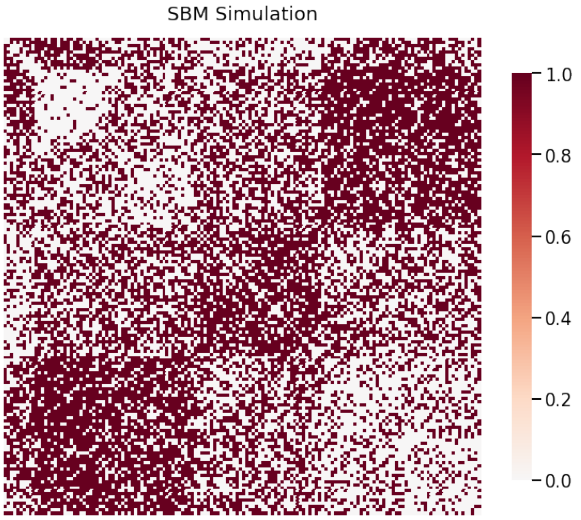

Fig. 1. Simulation of a SBM of 5 blocks, each containing 10, 20, 30, 40 and 50 nodes in each cluster, with corresponding probability matrix in SI 1).

Thresholds for weak and exact recovery. Consider the SBM of 2 equally sized communities with within probabilities p and between probabilities q for the existence of edges between nodes of the same or different clusters respectively. For which values of p and q can the underlying partition (the two clusters) be recovered? By "recovery" we will present necessary and sufficient conditions only in the case of exact and weak recovery, i.e. when the underlying partition of clusters is fully recovered or we have a non-trivial fraction of misclassified vertices and ignore the cases of almost exact recovery and partial recovery for simplicity.

When $p = q$ it is impossible to recover the communities, while the symmetric case of $p > q$ follows (4).

- Exact recovery in the 2-block SBM of n nodes with within probabilities $p = a\log(n)/n$ and between probabilities $q = b\log(n)/n)$ is solvable and efficient if $|\sqrt{a} - \sqrt{b}| > \sqrt{2}$ and unsolvable if $|\sqrt{a} - \sqrt{b}| < \sqrt{2}$.

- Weak recovery in the 2-block SBM of n nodes with within probabilities $p = a/n$ and between probabilities $q = b/n$ is solvable and efficient when $a, b = O(1)$ if and only if $(a - b)^2 > 2(a + b)$.

Conditions that are necessary and sufficient in the recovery of communities, such as the ones provided above, can impact greatly the crafting of algorithms that can solve the problem of community detection, as they show the limits of what is possible.

1. Testing partition structures

It is desirable to have tools by which one can test whether a proposed partition is a good fit for a given stochastic block model. We will compare a fixed partition of nodes with a proposed one where node assignments are unknown, random and modeled through a Chinese restaurant process (CRP) (5), which allows the number of clusters k to be inferred. Modelling node partitions in this way allows for flexibility in learning the inherent underlying structure of the network and is a logical model to test against for the block structure when compared with a pre-specified node partition (6).

Bayes factor. As the block connectivity structures of the SBM follow from Eq. (1), Eq. (2), we can see that $p(Y|z)$ is invariant under relabeling of the cluster indicators using Eq. (3).

To compare a given endogenous partition $\mathcal{M}$ with an exogenous one $\mathcal{M}^*$, the Bayes factor will be presented (7). Assuming that two models are equally likely a priori $p(\mathcal{M}) = p(\mathcal{M}^*)$, the Bayes factor of these models is

$$\mathcal{B}_{\mathcal{M},\mathcal{M}^*} = \frac{p(Y|\mathcal{M})}{p(Y|\mathcal{M}^*)}\frac{p(\mathcal{M})}{p(\mathcal{M}^*)} = \frac{\sum_{z\in\mathcal{Z}} p(Y|z)p(z)}{p(Y|z^*)}, \quad [4]$$

where the numerator and denominator of Eq. (4) depict the marginal likelihoods of Y under models $\mathcal{M}$ and $\mathcal{M}^*$ respectively.

In evaluating Eq. (4), the marginal likelihood of an exogenous model $p(Y|z^*)$ can be pointwise calculated using Eq. (3) at $z = z^*$, while for model $\mathcal{M}$, the calculation of $p(Y|z)$ for every $z \in \mathcal{Z}$ is required. Although the model assumed (CRP) has closed-form solutions for this expression, the cardinality of $\mathcal{Z}$ renders this approach useless. To bypass this problem, one can rely on Monte Carlo techniques to approximate $p(Y|\mathcal{M})$. (7) use the harmonic mean approach (8)

$$\hat{p}(Y|\mathcal{M}) = \left[\frac{1}{R}\sum_{r=1}^{R}\frac{1}{p(Y|z^{(r)})}\right]^{-1}, \quad [5]$$

where $z^{(1)}, \ldots, z^{(R)}$ are samples from the posterior distribution of z and $p(Y|z^{(r)})$ is given by Eq. (1) for $z = z^{(r)}$. Using Eq. (1) and Eq. (5), we derive

$$\hat{\mathcal{B}}_{\mathcal{M},\mathcal{M}^*} = \frac{\left[\frac{1}{R}\sum_{r=1}^{R}\prod_{1\le j<k\le k^{(r)}}\frac{\mathrm{B}(a,b)}{\mathrm{B}\left(a+m_{i,j}^{(r)},b+\bar{m}_{i,j}^{(r)}\right)}\right]^{-1}}{\prod_{1\le j<k\le k^*}\frac{\mathrm{B}\left(a+m_{i,j}^{*},b+\bar{m}_{i,j}^{*}\right)}{\mathrm{B}(a,b)}}, \quad [6]$$

where $m_{i,j}^{(r)}$ and $\bar{m}_{i,j}^{(r)}$ are the counts of edges and non-edges between nodes in groups i and j, induced by the r-th MCMC sample of z, while $m_{i,j}^{*}$ and $\bar{m}_{i,j}^{*}$ are the number of edges and non-edges of nodes between clusters i and j induced by the exogenous assignments z^*. Also, $k^{(r)}$ and k^* are the number of unique labels in $z^{(r)}$ and z^* respectively. (9) provide the thresholds by which the Bayes factor provides weak or strong evidence for using a specific model against another one.

This gives us the ability to test whether a conjectured structure provides a better fit for a network compared to another one.

Inference and uncertainty quantification. It is of interest to characterize the posterior distribution of z using the Gibbs samples $z^{(1)}, \ldots, z^{(R)}$, when there is evidence by the Bayes factor Eq. (6). While other approaches rely on the computation of the posterior co-clustering matrix $\mathcal{C}$ with elements measuring the relative frequency of the Gibbs samples for the sub-diagonal elements being in the same cluster, and then applying classical clustering procedures, (7) propose using the approach of (10). This approach relies on a metric called Variation of Information (VI), which compares information inside two clusters $z, \hat{z}$ with information shared between them, via an entropy function H and a cross-information function I

$$\mathrm{VI}(z, \hat{z}) = H(z) + H(\hat{z}) - 2I(z, \hat{z}). \quad [7]$$

The goal is to minimize this function, which acts as a metric on the space of partitions, so as to obtain an optimal partition

$$z^* = \text{argmin}_{\hat{z}} \mathbb{E}[\text{VI}(z, \hat{z})|Y], \quad [8]$$

with the expectation taken with respect to the posterior distribution of z.

It is possible to construct credible sets around point estimates using the VI distance. First define a $1-\alpha$ credible ball around $\hat{z}$: order the partitions according to their VI distance from $\hat{z}$, and then define the ball to contain partitions less than a threshold distance from $\hat{z}$, such that the size of the ball is minimized and it contains at least $1-\alpha$ probability. For the simulation purposes, we will use the "edge" of the ball and thus get credible bounds (6).

Moreover, estimates for the block probabilities $\hat{W}_{i,j}$, $1 \leq i < j \leq \hat{k}$ can be obtained using Eq. (3)

$$\hat{W}_{i,j} = \mathbb{E}[W_{i,j}|Y, \hat{z}] = \frac{a + \hat{m}_{i,j}}{a + \hat{m}_{i,j} + b + \hat{\bar{m}}_{i,j}}, \quad [9]$$

with $\hat{m}_{i,j}$ and $\hat{\bar{m}}_{i,j}$ denoting the number of edges and non-edges between nodes in groups i, j respectively, from the posterior point estimate $\hat{z}$ of z.

CRP base model. Motivated by wanting to choose a prior that has an unspecified number of clusters and provides thus flexibility in learning the number of clusters, the Chinese restaurant process (CRP) (5) is introduced. Under this model, a node is assigned to an already existing cluster with probability proportional to the size of the cluster, and to a new cluster, with probability proportional to a tuning parameter α. Specifically, the prior over clusters for node u, given the membership vector $z_{-u} = (z_1, \ldots, z_{u-1}, z_{u+1}, \ldots, z_n)^\top$ of other nodes is

$$p(z_u = i|z_{-u}) = \begin{cases} \dfrac{n_{i,-u}}{n-1+\alpha}, & \text{if } i = 1, \ldots, k_{-v}, \\ \dfrac{\alpha}{n-1+\alpha}, & \text{if } i = k_{-u}+1. \end{cases} \quad [10]$$

k_{-u} is the number of non-empty groups in z_{-u}, the integer $n_{i,-u}$ is the total number of nodes in cluster i without node u, and $\alpha > 0$ is a parameter controlling the expected number of nonempty clusters.

Figure 2 provides an example by using the above framework to recover the block structure of Figure 1. Specifically, to generate Figure 2 we derive the posterior distribution distribution on the node partitions, using the CRP prior Eq. (10), the likelihood by Eq. (3) via the Gibbs sampler Algorithm 1 of (7). We use 15000 samples, after a burn-in of 2000, and note that the error in classification is 36%. That is to be expected, due to the true structure of Figure 1 having not distinct "enough" blocks (most between and within probabilities are around 0.5, making the recovery of the true partition harder, as we saw in the subsection regarding exact and weak recovery).

2. Gibbs-type stochastic block models

Despite the existence of Bayesian nonparametric models and their use in mixture models, hidden Markov models and models using the Dirichlet process, the transition of their use in networks has only recently been made (7), (6). Using such approaches in community detection in networks will allow us to get a distribution over partitions of nodes. This approach

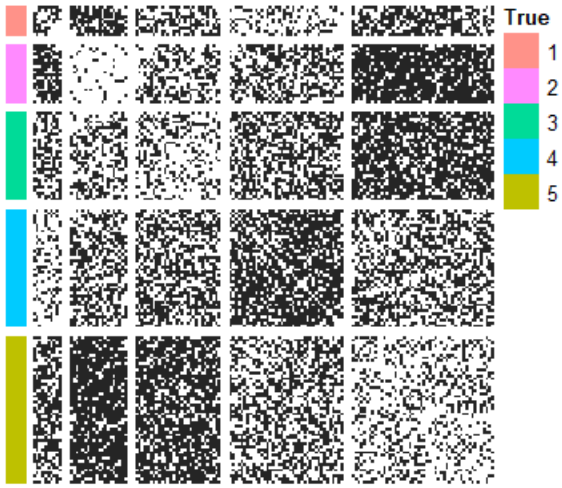

Fig. 2. Simulated adjacency matrix Y of 1 partitioned in blocks, according to the model assignments $\hat{z}$. Black and white pixels represent the existence or not of edges, while the true partition z_0 is the left coloured column.

provides greater flexibility by using different models that can provide custom-made solutions, interpretable results, metrics against which we can test the fit of node partitions to networks and a framework by which we can update the network and predict in which clusters new nodes most likely belong to.

A. Prior specification. It is possible to get rich posterior results through considering different priors on node partitions, and uncovering characteristics and network structures that purely algorithmic approaches wouldn't provide. Several priors have been introduced in the SBM framework, such as the Dirichlet-multinomial (11), the Dirichlet process (12) and mixtures of finite Dirichlet mixtures (13). All the above cases are included in the Gibbs-type class of priors (14).

Unsupervised Gibbs-type priors. (6) introduce Gibbs-type priors for node partitions in networks as follows. A probability mass function $p(z)$ is of Gibbs-type (14) if it has the form

$$p(z) = \mathcal{W}_{n,k} \prod_{i=1}^{k} (1-\sigma)_{n_i-1}, \quad [11]$$

where n_i is the number of nodes in cluster i, $\sigma < 1$ is the discount parameter, $\{\mathcal{W} : 1 \leq k \leq n\}$ is a family of non-negative weights such that $\mathcal{W}_{n,k} = (n - k\sigma)\mathcal{W}_{n+1,k} + \mathcal{W}_{n+1,k+1}$, $W_{1,1} = 1$ and $(\alpha)_n = a(a+1)\ldots(a+n-1)$ is the ascending factorial for $n \geq 1$ with $(\alpha)_0 = 1$. Gibbs-type priors can be viewed as models describing seating mechanisms, where the membership (or seating) indicators z can be obtained sequentially as

$$\mathbb{P}(z_{n+1} = i) \propto \left\{ \begin{array}{ll} \mathcal{W}_{n+1,k}(n_i - \sigma) & i = 1, \ldots, k, \\ \mathcal{W}_{n+1,k+1} & \text{if } i = k+1. \end{array} \right\} \quad [12]$$

Intuitively, a new node is classified to an existing cluster i with probability proportional to the number of nodes already belonging to that cluster, or to a new cluster. Specific choices for the weight parameters lead to different well-studied Gibbs-type priors in the context of Bayesian nonparametric statistics, such as the Dirichlet-multinomial (DM)

$$\mathcal{W}_{n,k} = \frac{\beta^{k-1}}{(\beta\bar{k}+1)_{n-1}} \prod_{i=1}^{k-1} (\bar{k} - k) 1_{k \leq \bar{k}},$$

for some $\beta = -\sigma < 0$ and $\overline{k} \in \{1, 2, \dots\}$, the Dirichlet process (DP)

$$\mathcal{W}_{n,i} = \frac{\alpha^k}{(\alpha)_n}, \text{ for some } \alpha > 0,\ \sigma = 0,$$

the Pitman-Yor process (PY) for $\sigma \in [0, 1)$,

$$\mathcal{W}_{n,k} = \prod_{i=1}^{k} \frac{(\alpha + \sigma k)}{(\alpha + 1)_{n-1}}, \text{ for some } \alpha > -\sigma,$$

and the Gnedin process (GN) for some $\sigma = -1$

$$\mathcal{W}_{j,k} = (\gamma)_{n-k} \frac{\prod_{i=1}^{k-1}(i^2 - \gamma i)}{\prod_{j=1}^{n-1}(j^2 + \gamma j)}, \text{ and } \gamma \in (0, 1).$$

Different choices of the prior will affect our results substancially. There are instances where the formation of clusters of small size is favored (e.g. PY may be better than DP), while in cases where regimes with slower increments are favored DP may be better (6).

Supervised Gibbs-type priors. Incorporating additional known information for making better inference decisions is of great importance in this Bayesian framework in networks. We leverage node attributes $x_n = (x_{n1}, \dots, x_{nd})^\top$ when these are available for each node $n = 1, \dots, n$, to incorporate it into inference on block structures. (7) replace Eq. (11) with

$$p(z|X) \propto \mathcal{W}_{N,k} \prod_{i=1}^{k} p(X_i)(1 - \sigma)_{n_i - 1}, \qquad [13]$$

with $X = (x_1, \dots, x_n)^\top$, where $X_i = \{x_u : z_u = i\}$ are the attributes of nodes in cluster i.

In the case of the node attributes x_u taking categorical values in $\{1, \dots, C\}$ the recommended practice is to use the Dirichlet-multinomial cohesion

$$p(\mathbf{X}_i) \propto \frac{1}{\Gamma(n_i + \alpha_0)} \prod_{c=1}^{C} \Gamma(n_{ic} + \alpha_c) \qquad [14]$$

for n_{ic} being the number of nodes in cluster i of attribute value c, $\alpha_0 = \sum_{c=1}^{C} \alpha_c$, with $\alpha_c > 0$ for $c = 1, \dots, C$ and Γ the Gamma function. One can also here derive an urn scheme using 13, and use it to define a Gibbs sampler to be used for posterior computation and inference (7).

B. Posterior and inference. As the already presented Gibbs-type priors (DM, DP, PY, GN) admit urn schemes (iterative algorithms by which one can sample from the corresponding processes), one can leverage this representation combined with Eq. (3) to derive a collapsed Gibbs samplers and thus characterise the posterior distribution over the space of node partitions of the above processes (6). Such models are called ESBM((7)). In this way, we can for example run the Louvain algorithm (15) or any community detection algorithm we wish to get a first estimate of the number of clusters. Then, we can adjust our prior's model parameters in subsection A, so that we induce approximately the number of clusters found by the previous algorithm. ESBM derives a posterior distribution over the space of node partitions in the London gang network. The variation of information VI Eq. (7) provides a way to quantify the distance of two clusters compared to their individual and joint entropies. Another measure for comparing models is through calculating the logarithm of the marginal likelihood $\log p(Y)$ (specifically in our application using the mean-field approximation as done in 5).

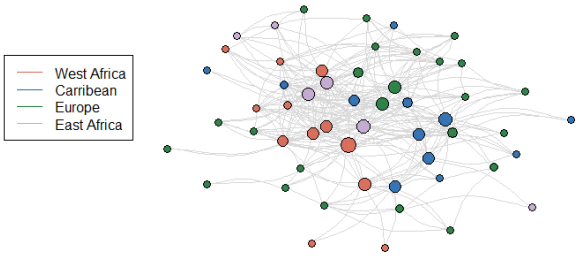

Fig. 3. The 54×54 London gang network with node colorings according to the birthplace of gang members and the size of nodes proportional to their betweenness. The layout used was the Kamada-Kawai layout algorithm.

Application in the London gang network. The network that we will apply the above procedure for characterizing its structure will be the London gang network (16), (17). The data consists of a 54×54 undirected network of interactions of co-offending individuals in a London-based inner-city street gang from 2005 to 2009, operating from a social housing estate. It was collected via anonymised police arrests and convictions for confirmed members of the gang. It consists of the interactions of known members, ranging from weak ones ($1 \equiv$ hang out together) to strong ones ($4 \equiv$ co-offend together, serious crime, kin). This data was processed to produce a 54×54 adjacency matrix of node interactions regardless of their strength, taking values 1 and 0, where node interactions have taken place or not respectively.

The ESBM models are applied to the London gang network. Specifically, we run the Louvain algorithm (15) on the network and get an estimated number of 5 clusters. Then the parameters of the priors DP, DM, PY, GN unsupervised and GN supervised are selected by setting the expected number of clusters induced by the priors to be the number of clusters inferred by the Louvain algorithm, specifically 5. Based on these model parameter selections, we calculate the posterior for the models without node attributes in the cases of DM, DP, PY, GN [unsup] and we incorporate the birthplace of the gang members to derive the posterior GN [sup] for 50000 samples. In particular, we have set the model parameters: $\alpha = 1.2$ for the DP, $\sigma = 0.52$ for the PY, $\overline{k} = 5$, $\beta = 12/5$ for the DM and $\gamma = 0.55$ for the GN (both for the supervised and unsupervised version), which induce 5 clusters in the prior. Furthermore, we calculate the logarithm of the marginal likelihood under the above priors and perform a burn-in of 10000 samples to see satisfactory mixing and convergence of the Gibbs algorithm in Figure 8.

Table 1 depicts the performance of the proposed models. In particular, we see that the supervised version of the GN process produces the best results, both in terms of minimizing the deviance and the VI distance. We see that incorporating the birth place information shows that GN [sup] is the best model over the unsupervised GN[unsup] and all other priors, as it produces the best results. Also, note that the cluster initialization seems to not be significant, as experiments were run for a number of 3 and 5 induced clusters via the priors,

Prior	$\log p(Y)$	$\mathrm{VI}(\hat{z}, z_{cr})$	Clusters k
DM	529.2	1.12	5 [5,5]
DP	508.9	0.87	8 [8,9]
PY	515.7	0.81	9 [9,10]
GN[unsup]	512.6	1.01	9 [8,9]
GN[sup]	506.8	0.68	9 [8,9]

Table 1. Performance of the ESBM when the number of cluster induced by the priors is 5. The measures used are the minus logarithm of the marginal likelihood $\log p(Y)$ (first column) and the VI distance between the calculated partition $\hat{z}$ and the 95% credible bound z_{cr}, both of which we want to minimize. Also, the posterior median number of the non empty clusters k is shown in the third column (first and third quantiles in brackets).

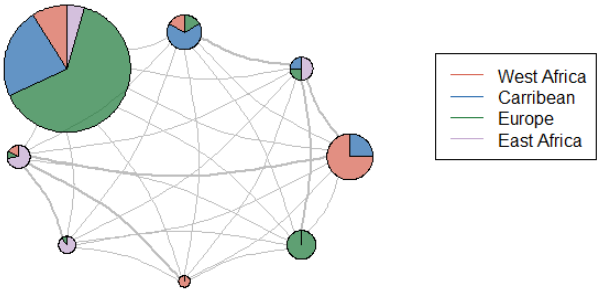

Fig. 4. Network representation of the inferred clusters in the London gang network using a circle layout. The supervised GN prior inducing an expected number 7 clusters is used. Each node denotes one group and edges are weighted by the estimated block probabilities. Node sizes are proportional to cluster cardinalities, while pie charts represent compositions with respect to birth locations.

where similar results were produced (8 to 10 clusters inferred, disregarding the worst performing DM process).

When comparing the supervised ($\mathcal{M}$) versus the unsupervised GN process ($\mathcal{M}^*$) in this application, we get the Bayes value of $\mathcal{B}_{\mathcal{M},\mathcal{M}^*} = 11.6$, which provides very strong evidence (using the thresholds of (9)) for the use of the supervised GN prior over the unsupervised one. This indication is made even clearer when comparing their VI distance, as seen in 1. The cluster representation of the network clusters inferred by the supervised GN prior in table 1 can be seen in Figure 4. Finally, the corresponding block structure and block probability matrix are presented in Figure 5. A comparison of the block structure of the clusters inferred by the Louvain algorithm, spectral clustering and the GN[sup] are depicted in Figure 7.

Model discussion. Depending on the choice of the prior and likelihood, a certain number of clusters and specific type of posteriors are introduced. Depending on the application, specific tailoring has to be done in order to ensure that the prior introduced will provide a sufficiently rich and appropriate structure via the posterior to describe the underlying true structure of the network. For example, if we know that the cluster sizes may be approximately equal, the DP isn't a good model, as it has the CRP as an underlying de Finnetti distribution, which is by its definition it is a rich get richer model and will favor the creation of big clusters and power-law degree distributions. Also, the MCMC samplers (7), (6) that are being used, can be computationally costly and inefficient for large networks and more efficient algorithms need to be developed. Finally, a more dynamical approach to networks can be taken using this framework, as one can assign new nodes to clusters via prediction formulas that are available in the mentioned Gibbs-type models.

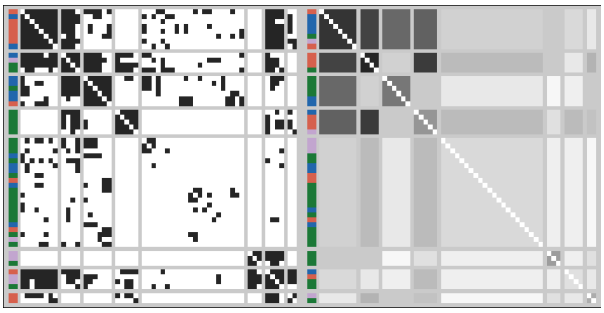

Fig. 5. Adjacency matrix (left) and estimated edge-probability matrices (right) of the London gang network with block structures inferred using the supervised GN process prior. Stronger shades of grey in the latter show a higher probability of connections within the block. To the left of the two blocks lie the colors corresponding to the birthplaces of the corresponding nodes, as can be seen from Figure 4.

3. Degree-corrected SBM

(18) introduced the degree-corrected stochastic block model (DCSBM) in undirected graphs with self-loops. Its formulation without self loops (for simplicity) is as follows (19).

First, we define the Poisson SBM (PSBM). For $Y_{uv} \sim$ Poisson to be the number of edges for the dyad of nodes (u, v), and W_{ij} the probability of a connection between a node in cluster i and a node in cluster j (expected number of edges of a node in cluster i to a node in cluster j), while the density of Y_{uv} is

$$p(Y_{uv}|Z, W) = (Y_{uv}!)^{-1}\exp(-Z_u^\top W Z_v)(Z_u^\top W Z_v)^{Y_{uv}}. \quad [15]$$

As in problems that had risen in the choice of the null model in the definition of modularity in regards to the degree sequence of the nodes, which were circumvented by the Newman-modularity definition (18)

$$Q = \frac{1}{2N}\sum_{ij}[Y_{ij} - \frac{k_i k_j}{2N}], \quad [16]$$

similar problems arise in the degree sequence in the SBM (18). Using the SBM produces degree distributions that are either Bernoulli or Poisson, which isn't realistic for real-world networks (20).

The DCSBM is defined by modifying the PSBM Eq. (15) network models of any degree distributions. In the DCSBM, the parameter ϕ_u is introduced for each node, such that $\sum_{u=1}^{n} \phi_u 1_{Z_{ui}=1} = 1$ for every cluster i, and that the expected number of edges for the dyad (u, v) is $\phi_u\phi_v Z_u^T W Z_v$. The density of Y_{uv} now is

$$p(Y_{uv}|Z, W, \phi) = \exp(-\phi_u\phi_v Z_u^\top W Z_v)\frac{(\phi_u\phi_v Z_u^\top W Z_v)^{Y_{uv}}}{Y_{uv}!}, \quad [17]$$

for $\phi = (\phi_1, \ldots, \phi_n)^T$. Note that the parameters ϕ_u are interpretable, as their maximum likelihood estimate are the ratio of u's degree to the sum of degrees in u's cluster.

Moreover, the DCBSM can be thought of as a generalization of the modularity optimization approach, through the planted

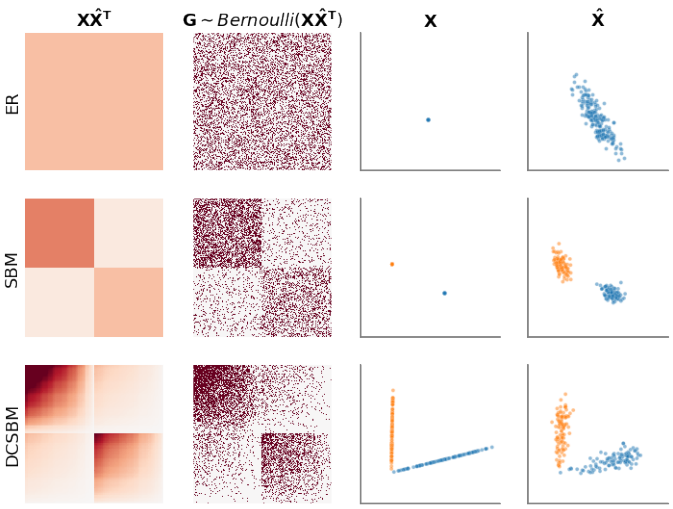

Fig. 6. Simulation of the ER, SBM and DCSBM models with additional representation via their RDPG form and the corresponding latent positions $X, \hat{X}$. Figure from (23).

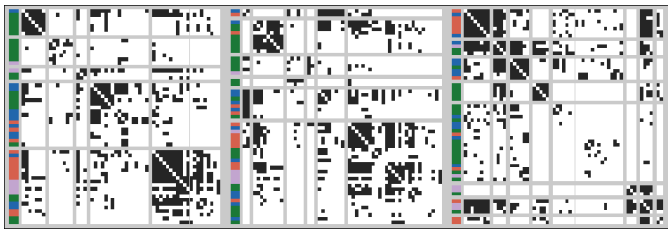

Fig. 7. Comparison of inferred block structures and clusters of the Louvain algorithm (left) (15), spectral methods (24) (middle) and supervised GN prior (right) for the London gang network. To the left of the three blocks lie the colors corresponding to the birthplaces of the corresponding nodes, as can be seen from Figure 4.

partition model (20). Specifically, (20) show that maximizing the probability of an observed network being generated according to the DCSBM for a specific choice of edge probabilities coincides with modularity optimization Eq. (16).

4. Random Dot Product Graph

Random dot product graphs can be thought of as generalisations of the stochastic blockmodel (SBM). Let F be a d-dimensional inner product distribution, i.e. for F a p.d.f. whose support $\text{supp}F = \mathcal{X}_d \subset \mathbb{R}^d$, we have for all $x, y \in \mathcal{X}_d$ that $x^\top y \in [0, 1]$. We define a random dot product graph (RDPG) (21) with d-dimensional inner product distribution F in the following way (22). Suppose we have $X_1, \dots, X_n \overset{i.i.d.}{\sim} F$ the rows of the matrix

$$X = (X_1, X_2, \dots, X_n)^\top \in \mathbb{R}^{n \times d},$$

and a random adjacency matrix A given by

$$\mathbb{P}(A|X) = \prod_{i<j} (X_i^\top X_j)^{A_{ij}} (1 - X_i^\top X_j)^{1-A_{ij}}. \qquad [18]$$

Then, we say that $(A, X) \sim \text{RPDG}(F, n)$ and denote A as the adjacency matrix of a random dot product graph of dimension at most d and with latent positions given by the rows of X. If $XX^\top$ is of rank d, then A is the adjacency matrix of a rand d random dot product graph.

Their similarity with the SBM can be seen directly from Eq. (2).

Acknowledgements. The author would like to thank (6), (7) for publishing code along with their papers, as for Figure 2 and the application in the London gang network that code was relied upon (but heavily customized) for the generation of the related figures. The corresponding code that can generate Figures 2, 3, 4, 5, 7 and 8 can be found in the code notebook files attached, with explanations and instructions within.

1. Holland PW, Laskey KB, Leinhardt S (1983) Stochastic blockmodels: First steps. *Social networks* 5(2):109–137.
2. Erdős P, Rényi A (1960) On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* 5(1):17–60.
3. Krivelevich M, Sudakov B (2013) The phase transition in random graphs: A simple proof. *Random Structures & Algorithms* 43(2):131–138.
4. Abbe E (2017) Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research* 18(1):6446–6531.
5. Aldous DJ (1985) Exchangeability and related topics in *École d'Été de Probabilités de Saint-Flour XIII—1983*. (Springer), pp. 1–198.
6. Legramanti S, Rigon T, Durante D, Dunson DB (2020) Extended stochastic block models. *arXiv preprint arXiv:2007.08569*.
7. Legramanti S, Rigon T, Durante D (2020) Bayesian testing for exogenous partition structures in stochastic block models. *Sankhya A* pp. 1–19.
8. Newton MA, Raftery AE (1994) Approximate bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society: Series B (Methodological)* 56(1):3–26.
9. Kass RE, Raftery AE (1995) Bayes factors. *Journal of the american statistical association* 90(430):773–795.
10. Wade S, Ghahramani Z, , et al. (2018) Bayesian cluster analysis: Point estimation and credible balls (with discussion). *Bayesian Analysis* 13(2):559–626.
11. Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association* 96(455):1077–1087.
12. Kemp C, Tenenbaum JB, Griffiths TL, Yamada T, Ueda N (2006) Learning systems of concepts with an infinite relational model in *AAAI*. Vol. 3, p. 5.
13. Geng J, Bhattacharya A, Pati D (2019) Probabilistic community detection with unknown number of communities. *Journal of the American Statistical Association* 114(526):893–905.
14. Gnedin A, Pitman J (2006) Exchangeable gibbs partitions and stirling triangles. *Journal of Mathematical sciences* 138(3):5674–5685.
15. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008(10):P10008.
16. Grund TU, Densley JA (2012) Ethnic heterogeneity in the activity and structure of a black street gang. *European Journal of Criminology* 9(4):388–406.
17. Grund TU, Densley JA (2015) Ethnic homophily and triad closure: Mapping internal gang structure using exponential random graph models. *Journal of Contemporary Criminal Justice* 31(3):354–370.
18. Karrer B, Newman ME (2011) Stochastic blockmodels and community structure in networks. *Physical review E* 83(1):016107.
19. Lee C, Wilkinson DJ (2019) A review of stochastic block models and extensions for graph clustering. *Applied Network Science* 4(1):1–50.
20. Newman ME (2016) Equivalence between modularity optimization and maximum likelihood methods for community detection. *Physical Review E* 94(5):052315.
21. Young SJ, Scheinerman ER (2007) Random dot product graph models for social networks in *International Workshop on Algorithms and Models for the Web-Graph*. (Springer), pp. 138–149.
22. Athreya A, et al. (2017) Statistical inference on random dot product graphs: a survey. *The Journal of Machine Learning Research* 18(1):8393–8484.
23. Pedigo B (2020) Neurodata.io notebooks.
24. Von Luxburg U (2007) A tutorial on spectral clustering. *Statistics and computing* 17(4):395–416.

z^*	z_0(True)	z_1(Random)	z_2(Refined)	z_3(Coarse)
$2\log\hat{\mathcal{B}}_{\mathcal{M},\mathcal{M}^*}$	47.1	1291.9	-13.1	720.3
$\mathrm{VI}(\hat{z}, z^*)$	0.1	4.2	0.38	1.23

Table 2. Comparison of four different partitions z^* with Chinese restaurant process $\mathcal{M}$ for describing the block structure of the SBM Y in Figure 1.

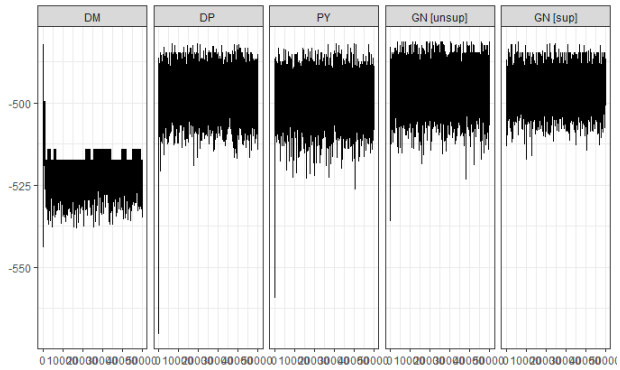

Fig. 8. Mixing and convergence of the Gibbs sampler in the ESBM model using 50000 samples and a burn-in of 10000, when we induce 5 cluster via the parameters of the priors.

5. Supporting Information (SI)

```
!pip install graspologic
from graspologic.simulations import sbm
from graspologic.plot import heatmap
n = [10, 20, 30, 40, 50]
p = [[0.5, 0.7, 0.5, 0.3, 0.6],
     [0.7, 0.1, 0.4, 0.5, 0.8],
     [0.5, 0.4, 0.3, 0.55, 0.7],
     [0.3, 0.5, 0.55, 0.7, 0.5],
     [0.6, 0.8, 0.7, 0.5, 0.29]]

np.random.seed(1)
G = sbm(n=n, p=p)
_ = heatmap(G, title ='SBM Simulation')
```

Listing 1. Python code for Figure 1.

```
degree_corrections = np.random.beta(2, 2, size=n)
for label in np.unique(labels):
    mask = labels == label
    degree_corrections[mask] = np.sort(
 degree_corrections[mask])[::-1]
    comm_sum = np.sum(degree_corrections[mask])
    degree_corrections[mask] = degree_corrections[
 mask] / comm_sum

dcsbm_graph = sbm(community_sizes, B, dc=
 degree_corrections, loops=True)
# get DCSBM P matrix
dcsbm_model = DCSBMEstimator(directed=False, loops=
 True)
dcsbm_model.fit(dcsbm_graph, y=labels)
dcsbm_model.block_p_ = B * (n // 2) ** 2  #
 block_p_ has a different meaning here
degree_corrections = degree_corrections.reshape(-1,
  1)
dcsbm_model.degree_corrections_ =
 degree_corrections
p_mat = _block_to_full(dcsbm_model.block_p_,
 _block_inv, dcsbm_graph.shape)
p_mat = p_mat * np.outer(degree_corrections[:, 0],
 degree_corrections[:, -1])
dcsbm_model.p_mat_ = p_mat

n_models = len(model_names)
n_cols = 4
scale = 3

fig, axs = plt.subplots(n_models, n_cols, figsize=(
 scale * n_cols, scale * n_models))


def simple_heatmap(mat, ax):
    sns.heatmap(
        mat,
        ax=ax,
        xticklabels=False,
        yticklabels=False,
        cbar=False,
        cmap="RdBu_r",
        center=0,
        square=True,
        vmin=0,
        vmax=1,
    )


def simple_scatter(X, ax, y=None):
    sns.scatterplot(
        x=X[:, 0],
        y=X[:, 1],
        hue=y,
        s=15,
        linewidth=0.25,
        alpha=0.5,
        ax=ax,
        legend=False,
    )
    ax.set(
        xticks=[], yticks=[], xlabel="", ylabel="",
  xlim=(-0.4, 1.4), ylim=(-0.4, 1.4)
    )


y = None
for i, model_name in enumerate(model_names):
    graph = graphs[i]
    p_mat = p_mats[i]
    graph_latent = graph_latents[i]
    p_mat_latent = p_mat_latents[i]
    if i > 0:
        y = labels

    ax = axs[i, 0]
    simple_heatmap(p_mat, ax)
    ax.set_ylabel(model_name)

    ax = axs[i, 1]
    simple_heatmap(graph, ax)

    ax = axs[i, 2]
    simple_scatter(p_mat_latent, ax, y=y)

    ax = axs[i, 3]
    simple_scatter(graph_latent, ax, y=y)

axs[0, 0].set_title(r"$\mathbf{P} = \mathbf{X \hat{
 X}^T}$")
axs[0, 1].set_title(r"$\mathbf{G} \sim Bernoulli(\
 mathbf{X \hat{X}^T})$")
axs[0, 2].set_title(r"$\mathbf{X}$")
axs[0, 3].set_title(r"$\mathbf{\hat{X}}$")
```

Listing 2. Python code for Figure 6.